LAB07B: Bypassing a strong preg\_replace(pattern, ‘ ‘, )

[Simon X. Camilo. Cybersecurity Student]

[BHCC CIT-275-WB Professor Philip Kazanjian]

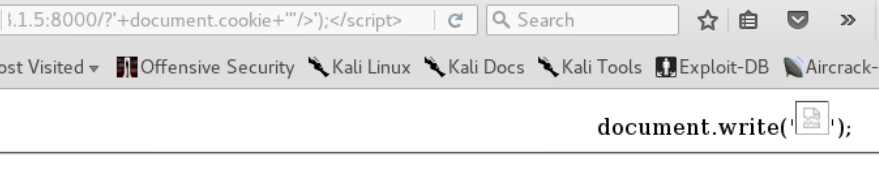
Introduction

The preg\_replace(pattern, ‘ ‘, ) function we are trying to bypass this time is stronger than the last one, this is preventing us from running script on the address bar. This document tries to finds ways to bypass that function and steal session ids.

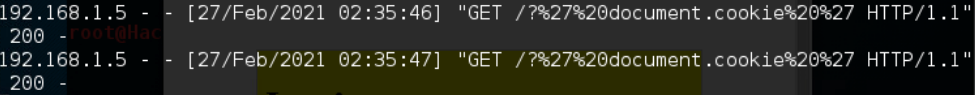
**What is preg\_replace?**

preg\_replace is a function that replaces a string when it follows a pattern. In this case it’s being used to replace any script and + sign that is located in any element that appears on the address bar. It also takes in consideration a lot of patterns. This makes it so attackers can’t use the methods of attack used on the previous function.

This stronger preg\_replace is effective as shown in Image 1



**Image 1** *Demonstration of stronger preg\_replace(pattern, ‘ ‘, )*



*Demonstration of stronger preg\_replace(pattern, null, ) on a server, we should expect to get a session id instead of our function.*

A demonstration of how this function works can be seen on the screenshot above. In this case, the function is:

preg\_replace('/<script[^\>]\*>|<\/script>|onabort|onblur

|onchange|onclick|ondbclick|onmousedown|onmousemove|onmouseout|

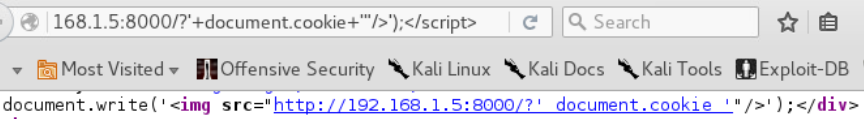
onmouseover|onerror|onfocus|onkeydown|onkeypress|onkeyup|onload|

onmouseup|onreset|onresize|onselect|onsubmit|onunload\s\*=\s\*"[^"]

+"/i', '', $\_REQUEST['myusername']);.

This make it so when the url above is entered, we can’t track the urls

The page source also shows that it’s not there anymore, it has been replaced with nothing. This is how preg\_replace works.

  
*Demonstration of stronger preg\_replace(pattern, null, ) in html source*

# Analysis

## Steps

1. Start the server
2. Bypass <script> by inserting “<script>” in the middle
3. Bypass + by encoding it
4. Result

### **Start the server**

First of all, to be able to see the session id’s of the victims, we should have a php server open that will track the ids. In this case, we are using a python server called “SimpleHTTPServer”. This module comes preinstalled with python and it can be opened with the command “python -m SimpleHTTPServer”

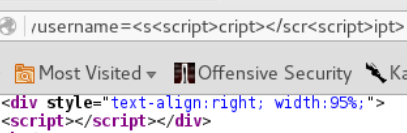


*Starting a SimpleHTTPServer on Python*

after doing this, we will be able to direct any information we get from our code to our server.

### Bypass <script> by inserting “<script>” in the middle

This is a very simple method, since the word “script” is blocked but not each letter or the result of the html. To be able to enter a script, you simple need to put the function between the letters. Doing something such as “<scri<script>pt>” will give us the output <script>. A html demonstration of this working can be shown below

  
*stronger preg\_replace bypass script result in html source*

An example outside of html can’t be shown without first being able to bypass how the + sign is blocked.

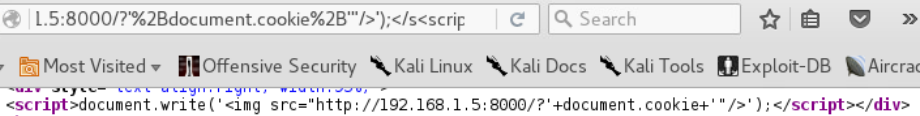
### Bypass + by encoding it

Every sign has can be translated to code, that browsers and websites automatically translate into letters, this is simply because of how programming works. It also helps in solving compatibility issues. We can take advantage of this since we can type something that’s the equivalent of a “+” sign, without being a “+” sign. For this, you will need a tool that is able to encode the sign, you can make your own tool, but there are also some tools online that can be used. In this case, we are going to use a command-line tool.

1. Translate the + sign

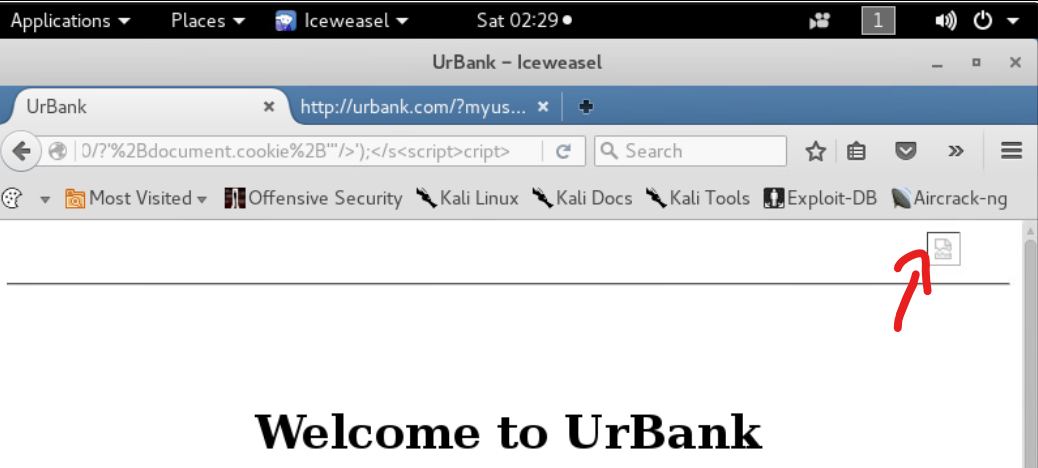
  
*Our result shows that our encoded + sign is %2B*

1. Enter it on the search bar and see our result

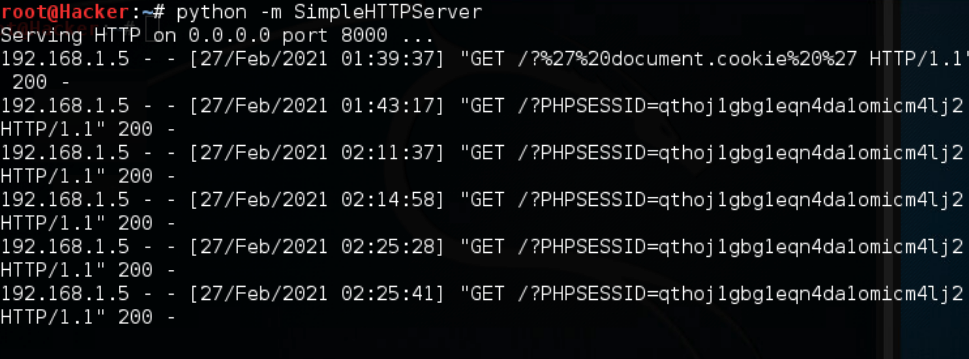
  
*Final result by combining our bypass for both script and the + sign in html source*

### Result

Now that we know how to bypass both the script block and the + sign block, we are able to combine them both to be able to steal session id’s for the site “urbank.com”

  
*Final result by combining our bypass for both script and the + sign*

We know it works because we have a broken icon on the top right of the page. We can also look at our server and see we now have stolen a session id



*Succesfully stolen session ids after sucessfulling running the script*

# Conclusion

This stronger preg\_replace(patter, ‘ ’, ) is a more thoughtout method of mitigation, and it can prevent hackers that don’t have any knowledge of text encoding. Despite of this, it is still not enough since someone with the knowledge can exploit this vulnerability. It’s recommended to use a better method to prevent these kind of attacks, instead of solely relying on this function

References

Preg\_replace. (n.d.). Retrieved February 26, 2021, from <https://www.php.net/manual/en/function.preg-replace.php>

PHP preg\_replace() Function. (n.d.). Retrieved February 26, 2021, from <https://www.w3schools.com/php/func_regex_preg_replace.asp>

Schurtz, S. (2013, December 12). Bypass a 'preg\_replace' Cross-site Scripting filter. Retrieved February 26, 2021, from <https://www.youtube.com/watch?v=XzXz0g196qg>