LAB08B: Bypassing addslashes

[Simon X. Camilo. Cybersecurity Student]

[Redacted]

note: I messed up and deleted my previous intro, the intro here is for 9b, the rest is from 7b

Introduction

There is a filter that prevents attackers from running malicious code that will allow them to run additional malicious code. This document shows how obfuscating code is effective in bypassing some filters.

**What is obfuscation?**

Obfuscation is the technique of making a message difficult to understand, most of the time the code will look like a bunch of random gibberish. Obfuscation is often used to hide information such as confidential documents or private messages. In technology, it is often used to hide malicious code, or with the focus of providing privacy and security to the users of the product. Obfuscation is famously used in the tor network, making it one of the main components that allows every user in the network to remain anonymous. It’s also used by some vpns, and many revolutionary privacy and security apps that had started being developed over the past decade.

# Analysis

As

## 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”

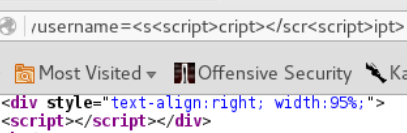


*Starting a SimpleHTTPServer on Python*

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.

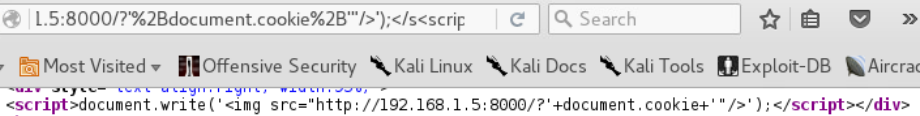
### Bypass + by encoding it

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

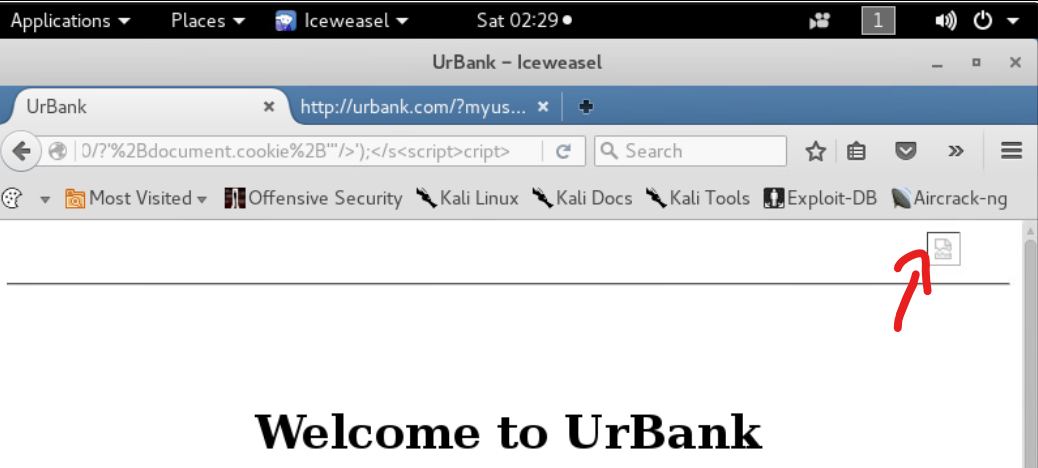
  
*Our result shows that our encoded + sign is %2B*

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

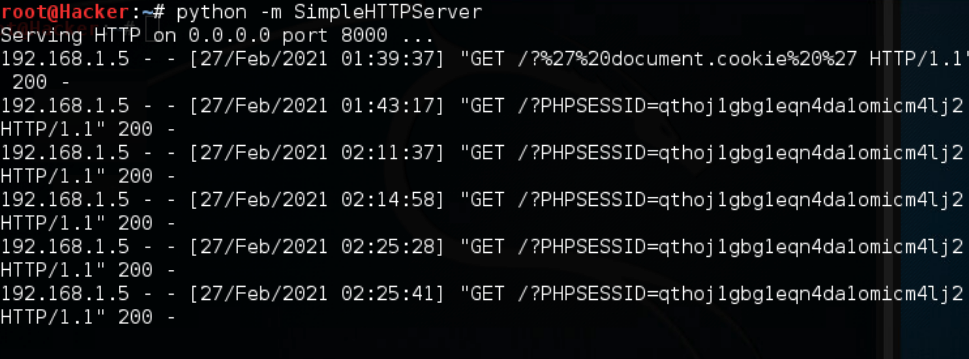
  
*Final result by combining our bypass for both script and the + sign in html source*

### 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>