LAB04B: Bypassing str\_replace(input, null, )

[Simon X. Camilo. Cybersecurity Student]

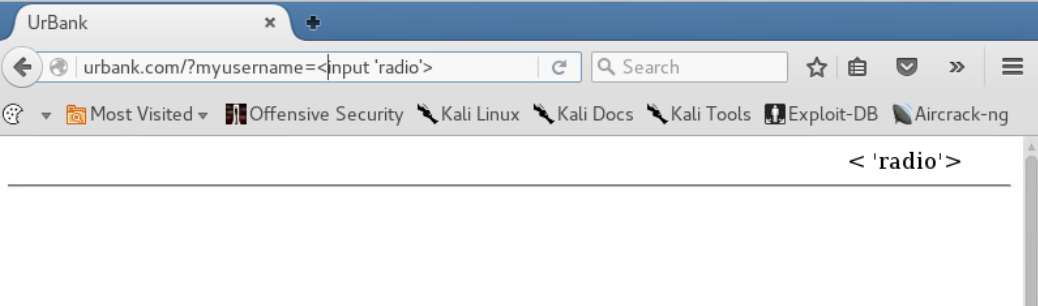
[Redacted]

Introduction

str\_replace(input, null, ) is a function that is disallowing the HTML INPUT from displaying an input field on the webpage. This document tries to finds ways to force the html input element into showing the input field

**What is str\_replace?**

str\_replace is a simple function that replaces a string with another string. In this case it’s being used to replace any code that is entered that says input, with nothing (null), this makes it so attackers can’t use the common method of attack, which is to add an input field for victims to use.



*Demonstration of str\_replace(input, null, )*

A demonstration of how this function works can be seen on the screenshot above. In this case, the function is str\_replace(input, null, $\_REQUEST[‘myusername’]). This make it so when the url above is entered, instead of showing an input field, the only thing that’s shown is the name of the variable.

The page source also shows that it’s not there anymore, it has been replaced with null (which means nothing), it has been erased. This is how str\_replace works.

  
*Demonstration of str\_replace(input, null, ) in html source*

# Analysis

## Ways to bypass str\_replace function

1. Changing upper/lower cases
2. Partially writing input before and after input
3. Extra: Try other commands instead of input (such as alert).

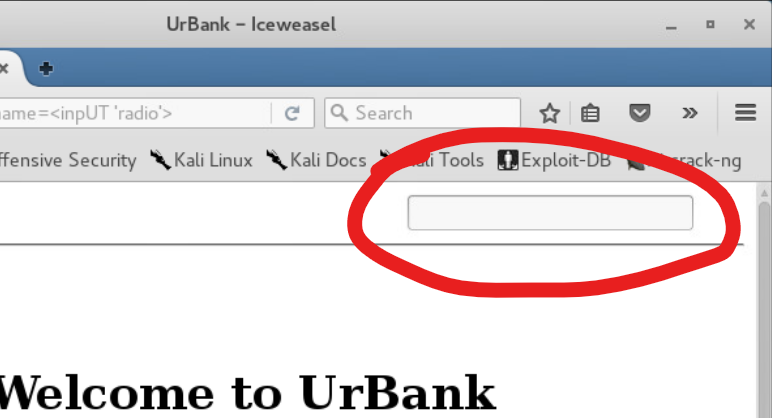
### Changing upper/lower cases

This is the easiest method and it will usually work if the web server owner is trying to bypass a word or a command, such as input, people who try to prevent these kind of attacks using str\_replace may not know that it’s case sensitive. They blocked the word “input” but not the word “Input” and many other variations. An attacker can take advantage of this to easily bypass the function.

1. The attacker can type this or a variation of this in the url urbank.com/?myusername=<inpUT’radio’>

  
*Bypassing str\_replace with cases*

2. And now you have another input field

  
*str\_replace bypass case result*

Here is how it looks in the page source, as you can see, it’s the same command but with different cases, and it works. str\_replace was not able to change it.

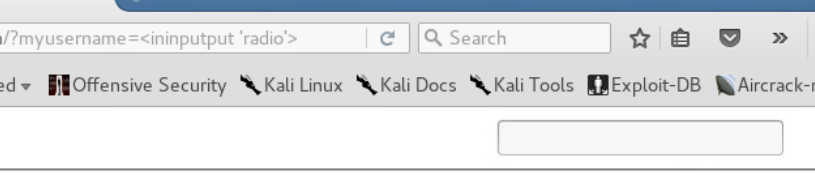


*Successful str\_replace(input, null, ) bypass in html source*

### Writting input before and after input

The wording is confusing. A better way to explain this, is that we should type ininputput instead of input. The word input that’s in the middle will be erased, making the other letters besides it combine and show the word input. This method is a lot more likely to work, since it has many variations that can be used to bypass that (even combine the method above).

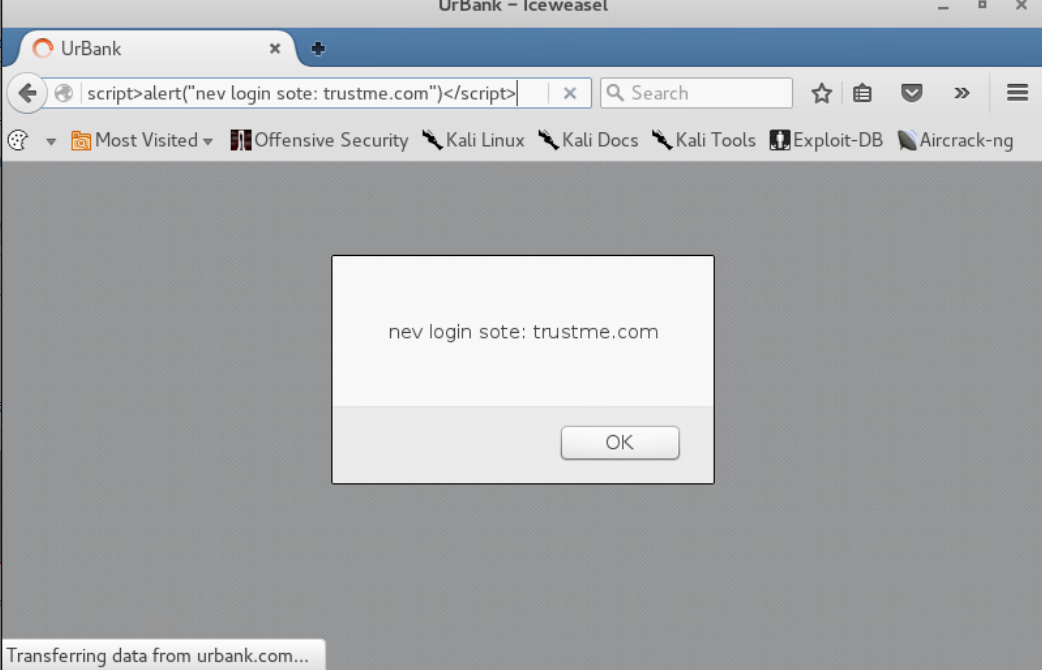
A demonstration of it is shown below. Typing urbank.com/?myusername=<ininputput ‘radio’> in the url will show our input field



*Bypassing str\_replace(input, null, ) with in the middle code*

### Trying something else (Extra)

If input or the command is really hard to bypass, and their workarounds don’t work. An attacker can instead try other ways to compromise the website. For example, you can have users have to look at a banner that tells them that the site they usually use to login is no longer going to be used by the company, and to go to their site that poses as a legitimate site by the company.



*Making changes on the website by working around str\_replace*

There are many other things an attacker can do if they are not able to use the input command.

# Conclusion

str\_replace is one of the most basic and most ineffective methods of mitigation, and it can be easily worked around by most hackers with a basic understanding of sql injection. It’s important to use more efficient methods to protect against these types of attacks, instead of relying on this function.

References

Str\_replace. (n.d.). Retrieved February 18, 2021, from https://www.php.net/manual/en/function.str-replace.php

Bypassing str\_replace. (n.d.). Retrieved February 18, 2021, from https://www.sevenlayers.com/index.php/328-bypassing-str-replace

A. (2016, January 26). Bypassing basic protections. Retrieved February 18, 2021, from <https://acrobertssec.wordpress.com/2016/01/26/bypassing-basic-protections/>