Austin Strobel

Homework 3 COSC 366

1.

This can prevent some XSS attacks. This type of attack can occur when the attacker injects JavaScript code into a webpage. The developer aims to filter out JavaScript code on the browser side in order to prevent malicious attacks. One scenario in which an XSS attack may not be able to be prevented is through false positives. It's quite difficult to create perfect JavaScript code that is able to entirely filter malicious code, so the attacker's code may still get through. Another scenario in which an XSS attack still may occur is if the attacker sends data through an API to bypass validation on the client side. Filtering on the browser side would then be useless.

2.

The JavaScript code that is allowed to execute is the following:

```
<script type="text/javascript" nonce="1rA2345">
  ... JavaScript Code ...
</script>
```

```
<script src="script.js"> </script>

<script src="https://example.com/script2.js"> </script>
```

3.

An additional layer of security is added the browser includes the token in the data and header fields. It is required that the token in the data field be manually added, while the domain automatically attaches the token. These tokens are then compared to confirm if they are from the same website, this preventing CSRF attacks. The bottom line is that the server compares the token in the data field of the HTTP request with the token in the header (that is automatically included in the browser) to make sure that they match.

4.

```
<!DOCTYPE html>
<html>
<head>
<script src="https://code.jquery.com/jquery-1.12.4.min.js"></script>
</head>
<body>
<script>
   function sendGETRequest() {
     $.ajax({
       url: 'http://www.example.com/delete.php?pageid=5',
       type: 'get',
       success: function(data) alert("success");
     });
   }

   $(document).ready(function(){
     sendGETRequest();
   });
 </script>
</body>
</html>
```