Daniel Oliveros

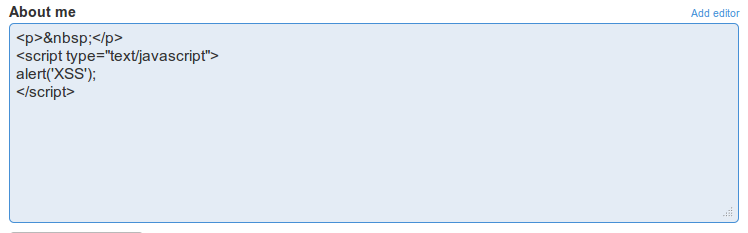
Garrett Bogart

Spring 2018 – Independent Study

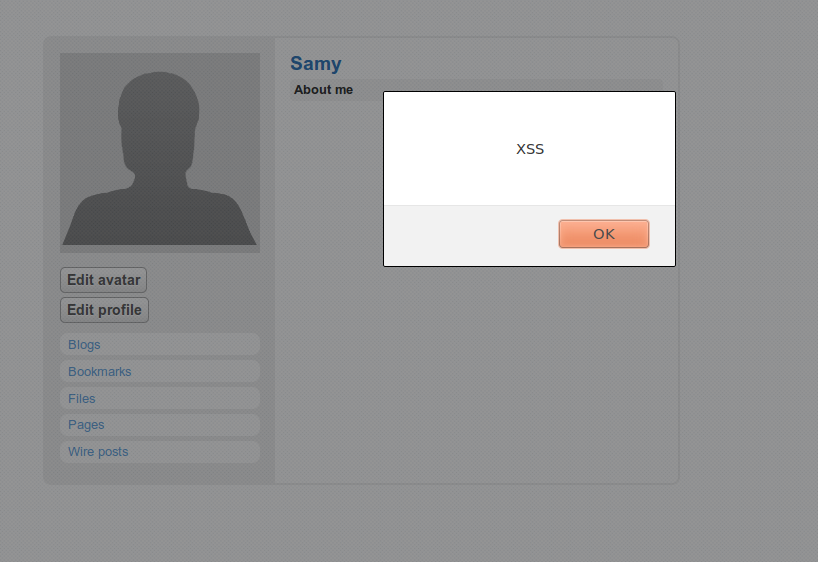
Cross Site Scripting Lab

**Task 1:**

After fixing the copy and paste issue that happens when you copy the code from the PDF (it pastes the wrong type of ‘). I pasted the code onto the **About me** page since that one allows you to remove the text editor and work with multiple lines for code editing. This is the code I pasted onto that section



And this is the popup the code caused:

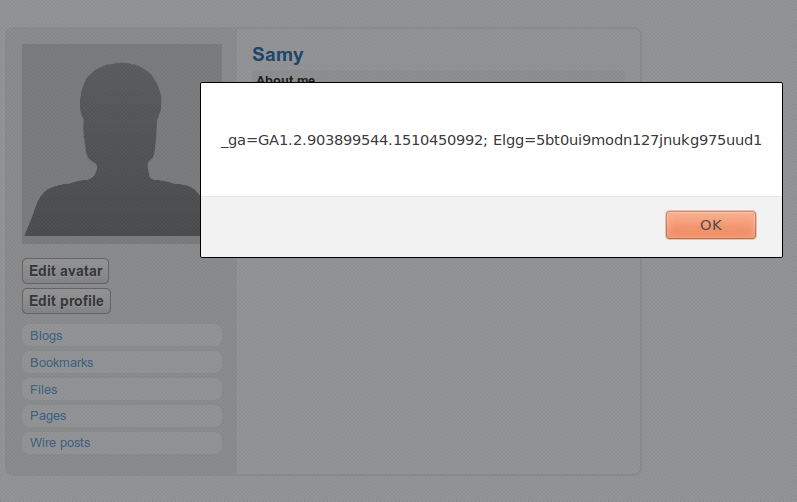
****

**Task 2:**

Changing the code to

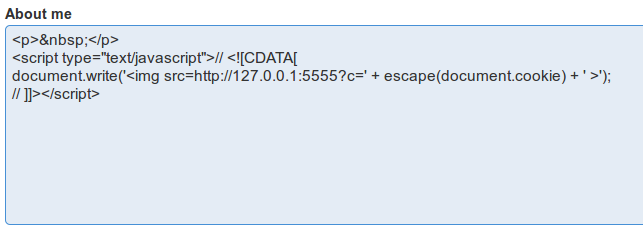


Caused the following popup, which contains the cookie data from the current session



**Task 3:**

We can route the cookie to a given ip address and port using the following code:

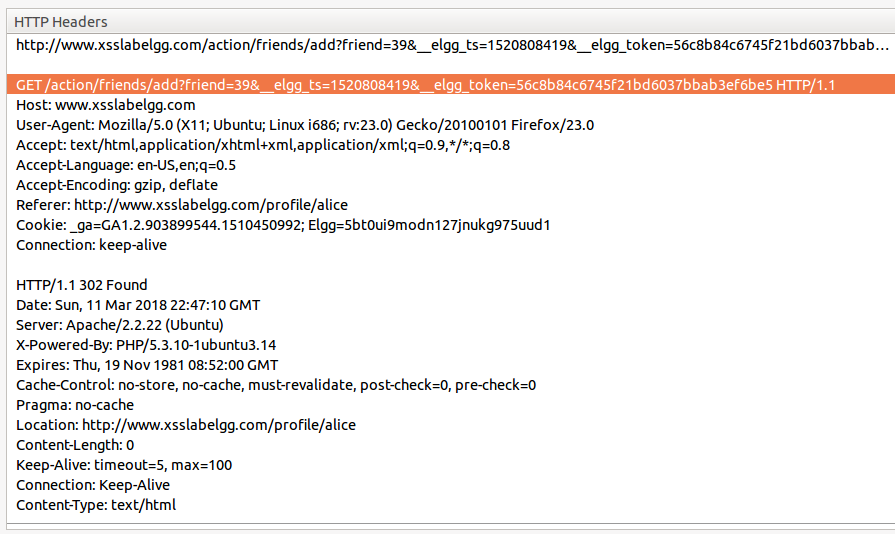


Listening into that port allows us to catch the cookie

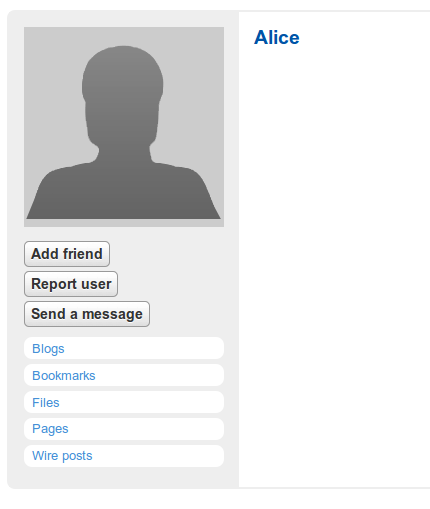


**Task 4:**

By using LiveHTTPHeaders, we can see the structure of a friend request. We can use this to write our HTTPSimpleForge.java program so it matches that request.

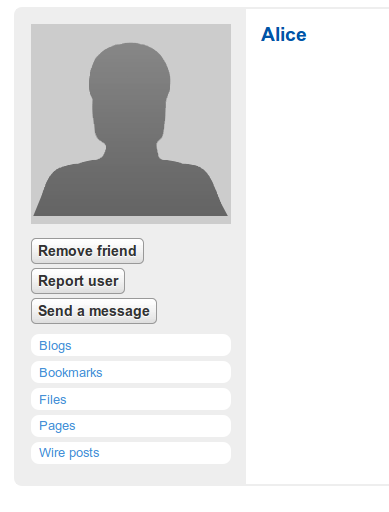


This is what Alice’s profile looks like before running the program



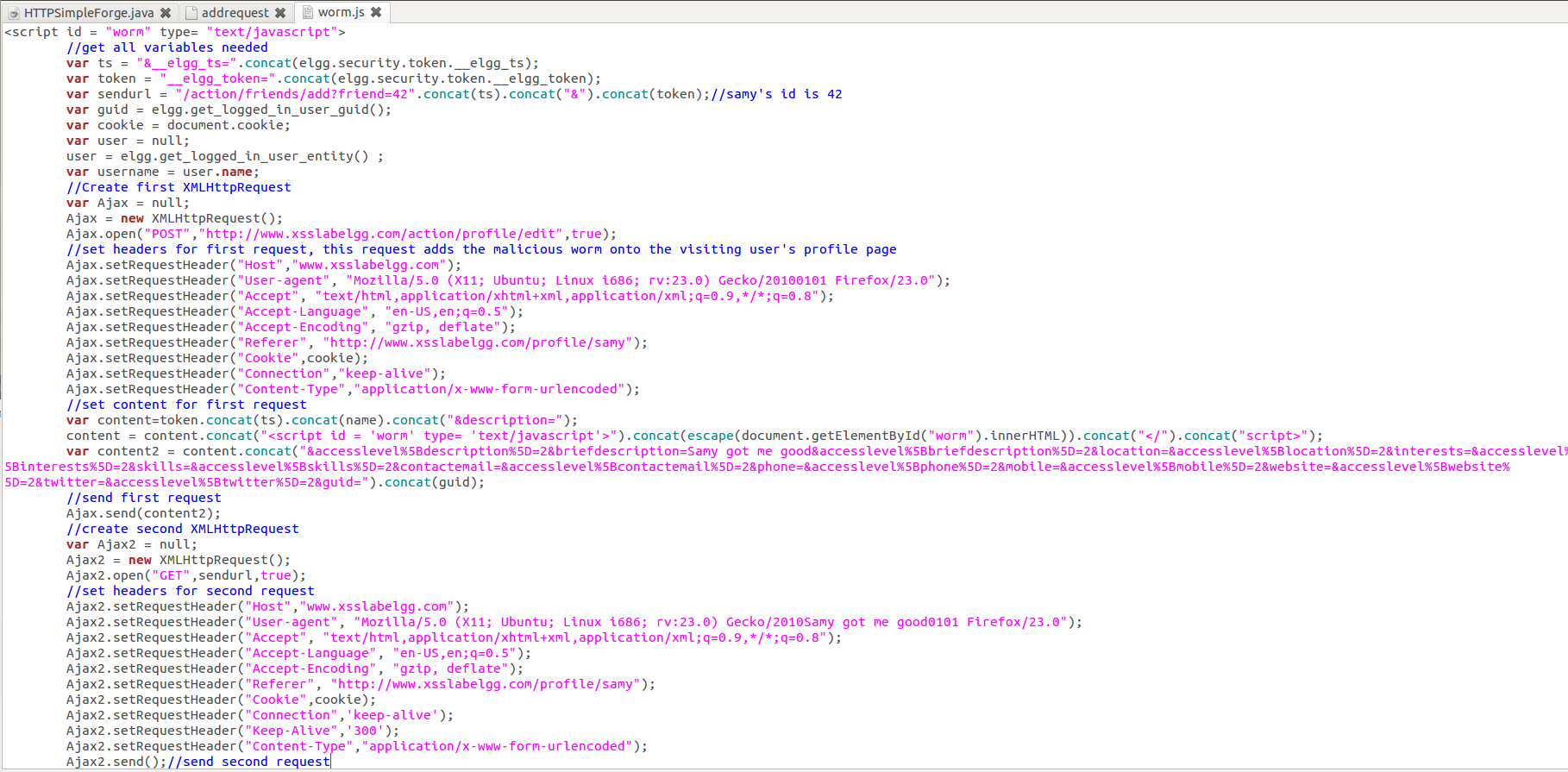


And this is what it looks like after running it, notice she has now been added as a friend.

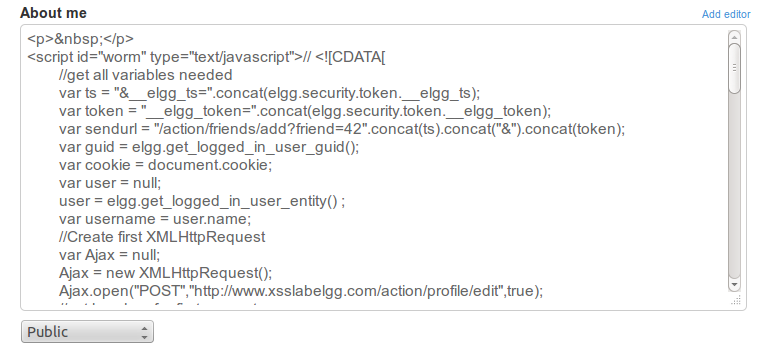


**Task 5 and 6:**

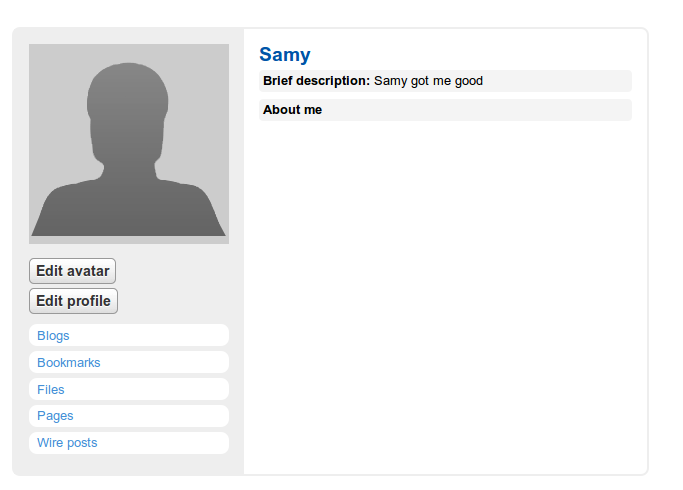
These tasks are best done together, as task 6 is pretty much a strictly better version of task 5. The javascript code we wrote for these tasks uses the id tag in order to reference itself when we need to post it into a victim’s profile.

****

We pasted this code in the About Me section, as it doesn’t have a character limit. This is what that section looks like with the javascript added to it.

****

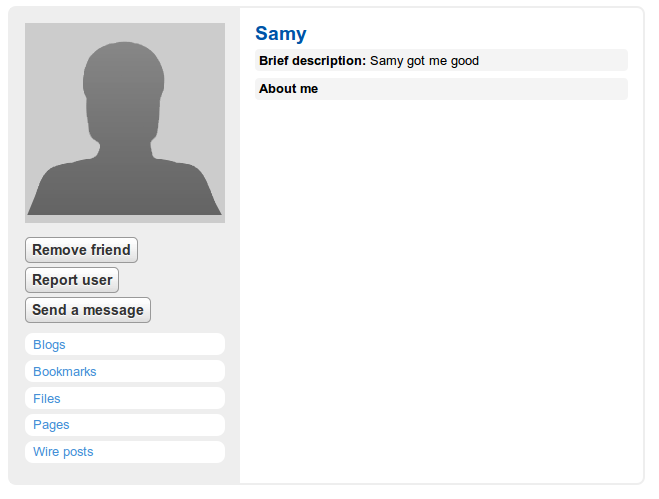
For this example, we’ve set the program up so that those affected by it will add Samy as a friend and add the malicious piece of code to their own profile. This is what Samy’s profile looks like to people who visit it.

****

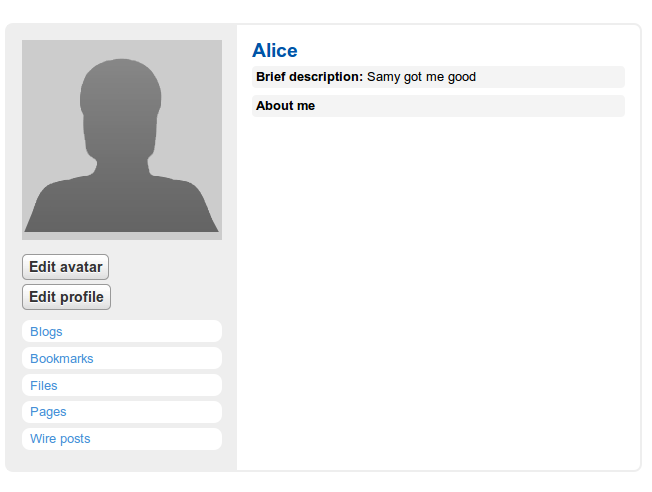
This is what Alice’s profile normally looks like from her perspective

****

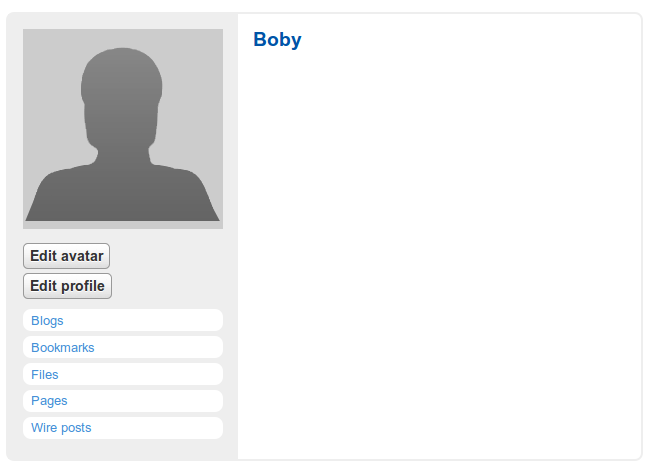
Once affected by Samy’s attack, this is what Samy’s profile will look like to Alice.

****

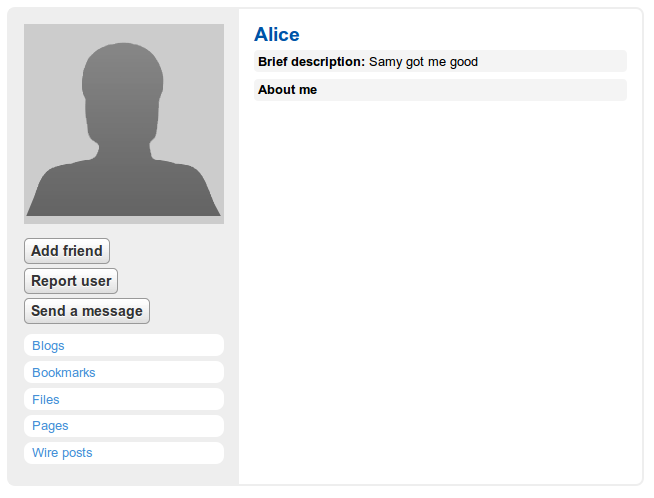
And this is what her own profile will look like, notice she now has the same code in her profile as Samy does.

****

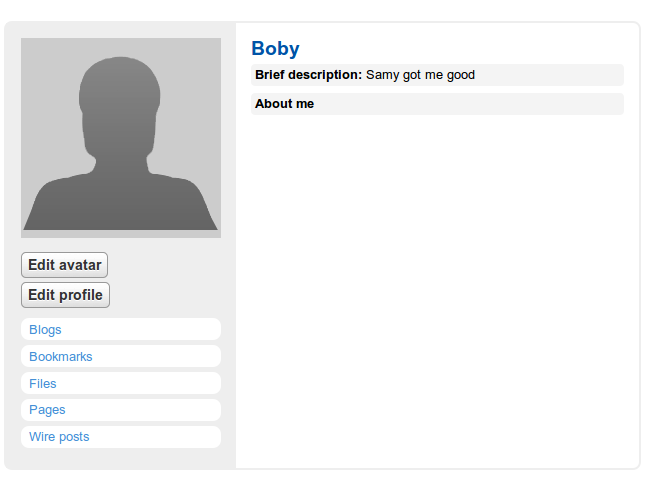
Logged in as Boby, this is how his own profile normally looks like

****

When he visits Alice’s profile, this is what he sees

****

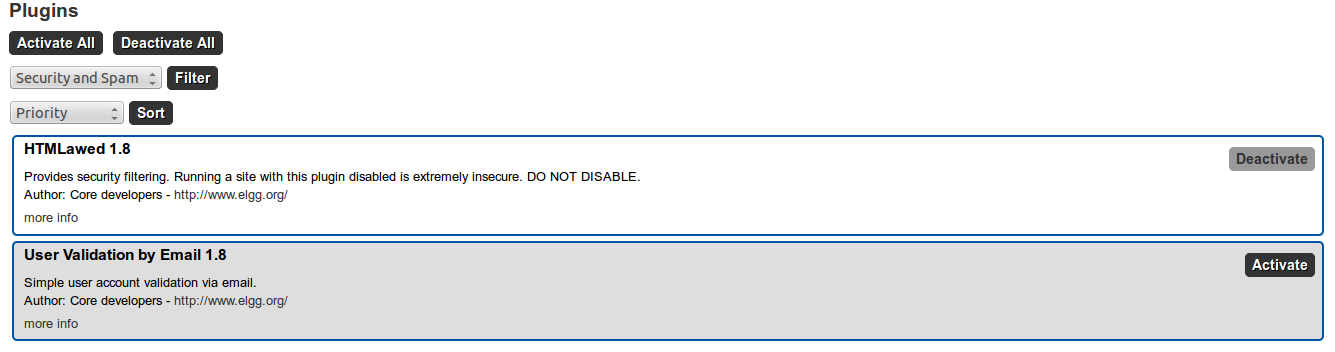
Visiting her profile executed the code stored within it, so now Boby’s profile has been infected as well.

****

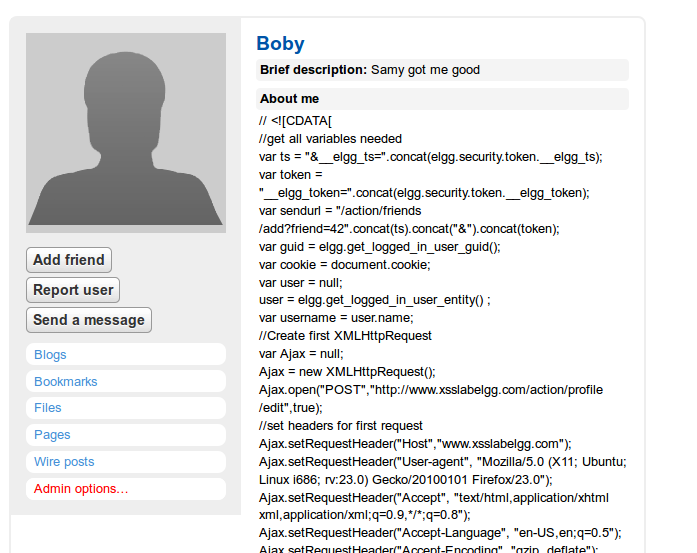
**Task 7:**

**1.**

Activating the countermeasure

****

With the countermeasure activated, this is what an infected profile looks like. This code is not executed, and is instead only displayed.

****

**2.** Turning on both countermeasures doesn’t really change much in the way things are displayed. Infected profiles would look the same as they did with the previous countermeasure turned on.

**Ideas for Improvement:**

1. This lab is a lot of fun, it was much easier to complete it the second time since we didn’t end up spending a long time trying to figure out how to write self-replicating code. Giving students a hint on how to do this would be a great way of avoiding frustration
2. There’s not really a need for tasks 5 and 6 to be separated, we’d recommend sticking to one method or making it so if someone solves task 6 they don’t have to do task 5.