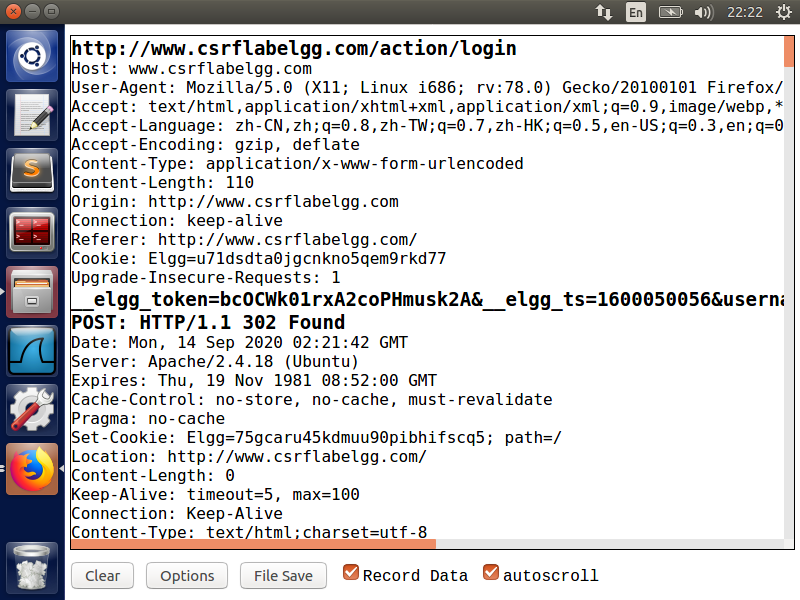
**Cross-Site Request Forgery (CSRF) Attack Lab**

姓名：严佳豪，学号：57118136

**Task 1: Observing HTTP Request**

使用admin 账号登陆 User: Admin; Username: admin; Password: seedelgg;

使用HTTP Header Live插件显示如下：



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：

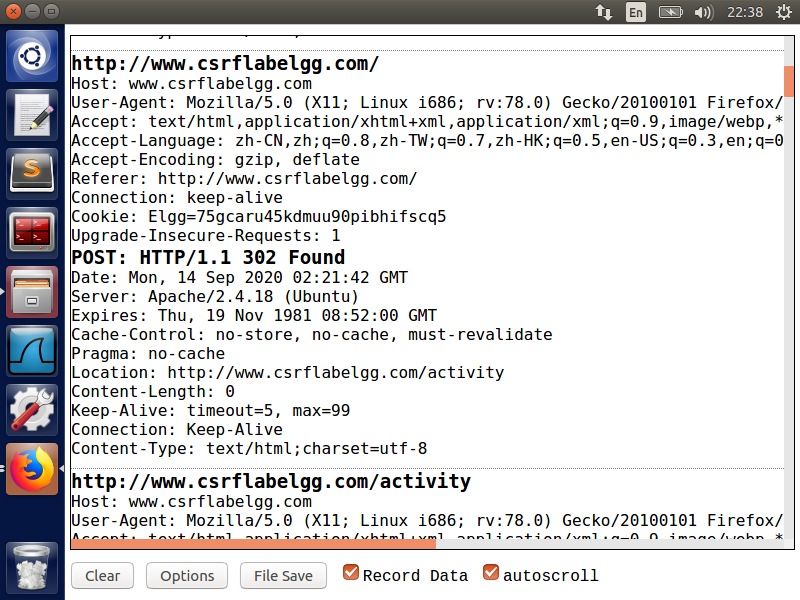
\_\_elgg\_token=bcOCWk01rxA2coPHmusk2A

\_\_elgg\_ts=1600050056

username=admin

password=seedelgg

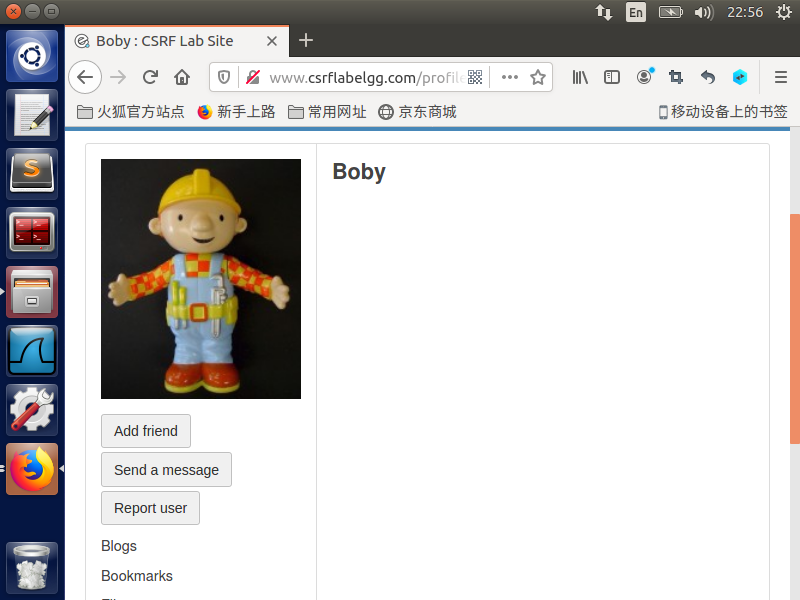
returntoreferer=true

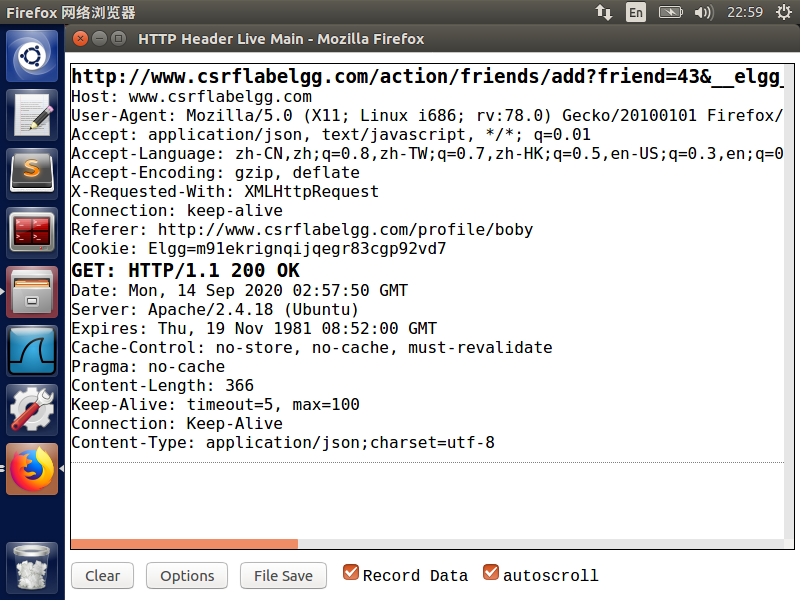


F12调用开发者工具下看此处实际应为GET的请求，请求根目录然后302重定向到activity，此次get请求无参数。

其后为返回200状态码，均为GET请求且无参数

**Task 2: CSRF Attack using GET Request**

首先以alice登录并搜索boby，点击添加好友查看请求

插件中显示如下 为GET请求，返回200

参数为：

\_\_elgg\_ts=1600052173

\_\_elgg\_token=sTcwhLJajCcEwpjykMKShA

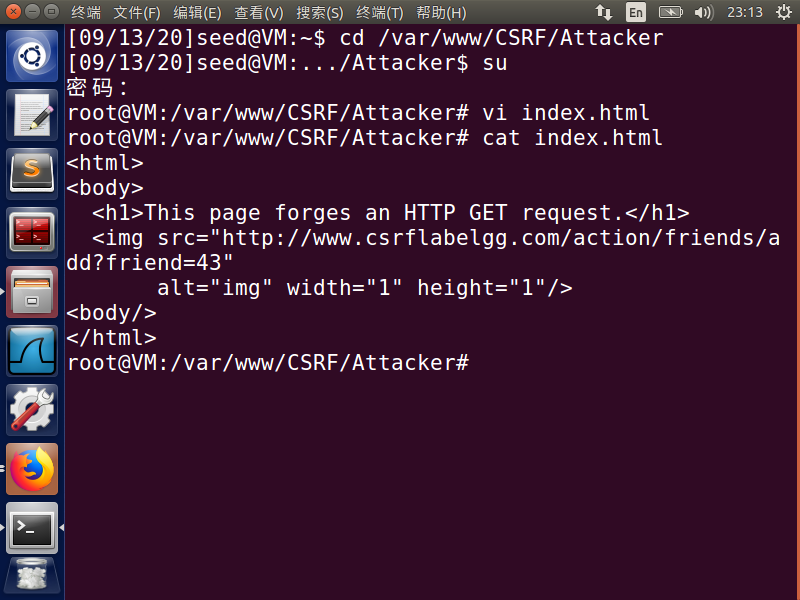
\_\_elgg\_ts=1600052173

\_\_elgg\_token=sTcwhLJajCcEwpjykMKShA

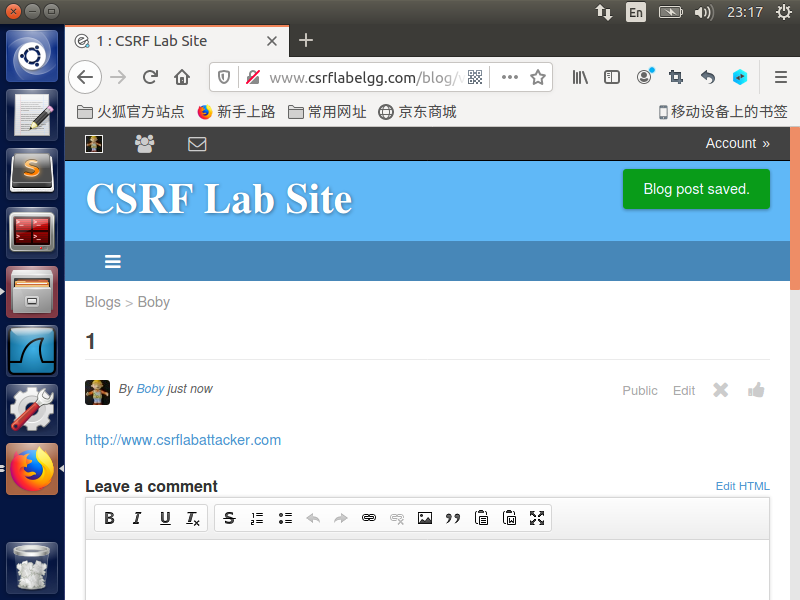
故只需伪造上图中的url即可，

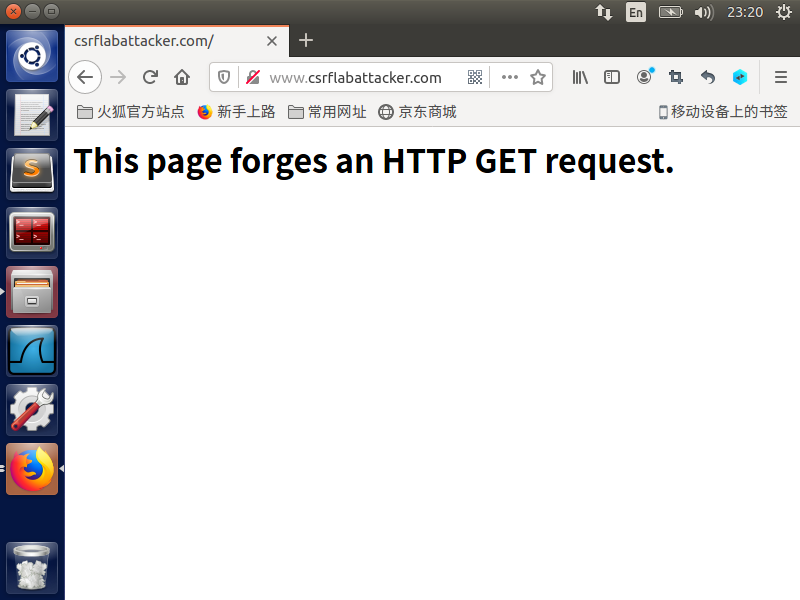
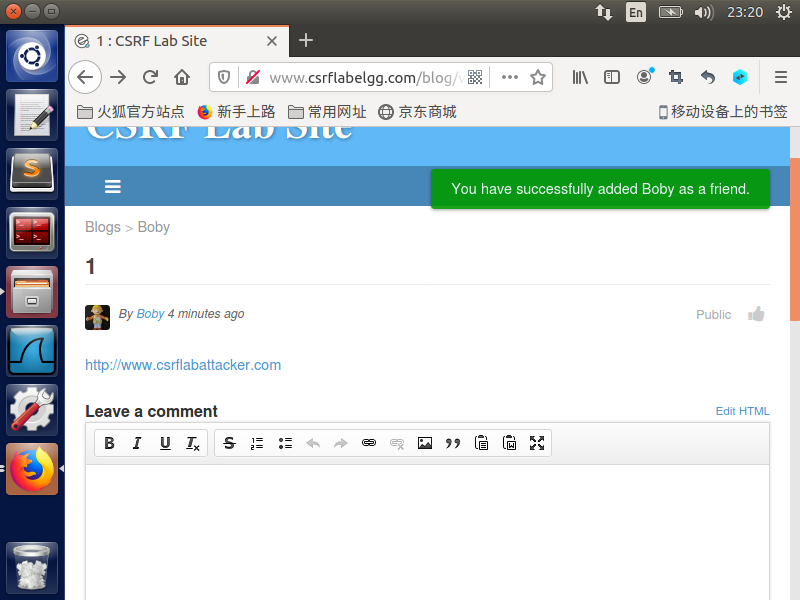
<http://www.csrflabelgg.com/action/friends/add?friend=43&__elgg_ts=1600052173&__elgg_token=sTcwhLJajCcEwpjykMKShA&__elgg_ts=1600052173&__elgg_token=sTcwhLJajCcEwpjykMKShA>

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。

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。

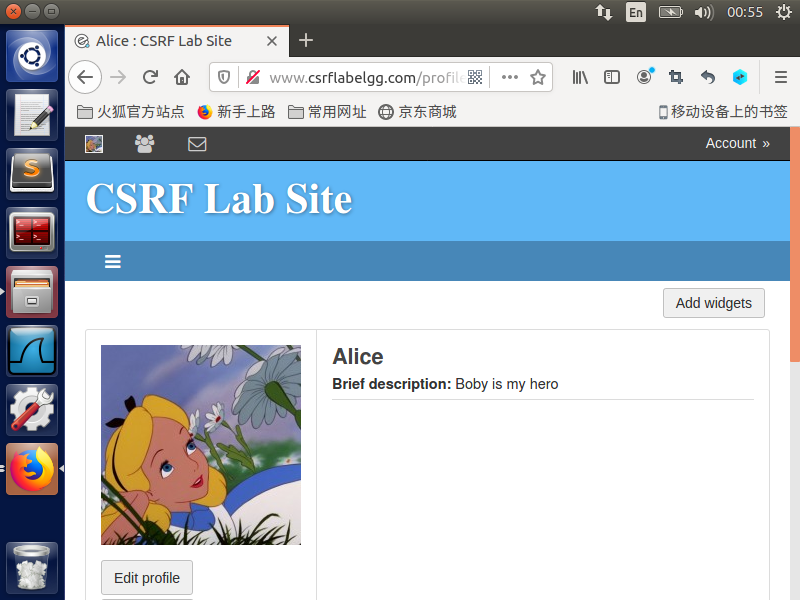
使用Boby登录，发布包含http://www.csrflabattacker.com链接的blog。



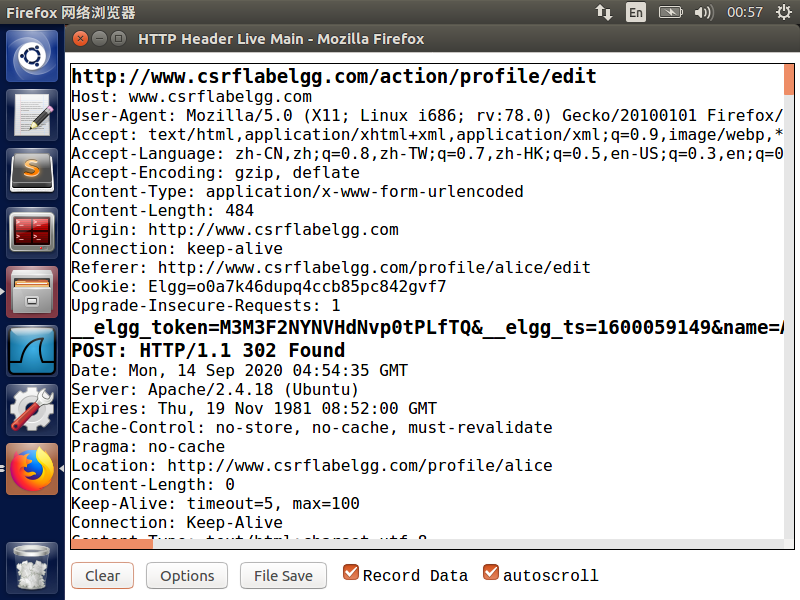
使用alice登录，访问该链接即可实现攻击

**Task 3: CSRF Attack using POST Request**

点击Edit profile 修改其brief description

点击save保存

发送的POST请求如下



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

参数如下：

\_\_elgg\_token=M3M3F2NYNVHdNvp0tPLfTQ

\_\_elgg\_ts=1600059149

name=Alice

description=

accesslevel[description]=2

briefdescription=Boby is my hero

accesslevel[briefdescription]=2

location=

accesslevel[location]=2

interests=

accesslevel[interests]=2

skills=

accesslevel[skills]=2

contactemail=

accesslevel[contactemail]=2

phone=

accesslevel[phone]=2

mobile=

accesslevel[mobile]=2

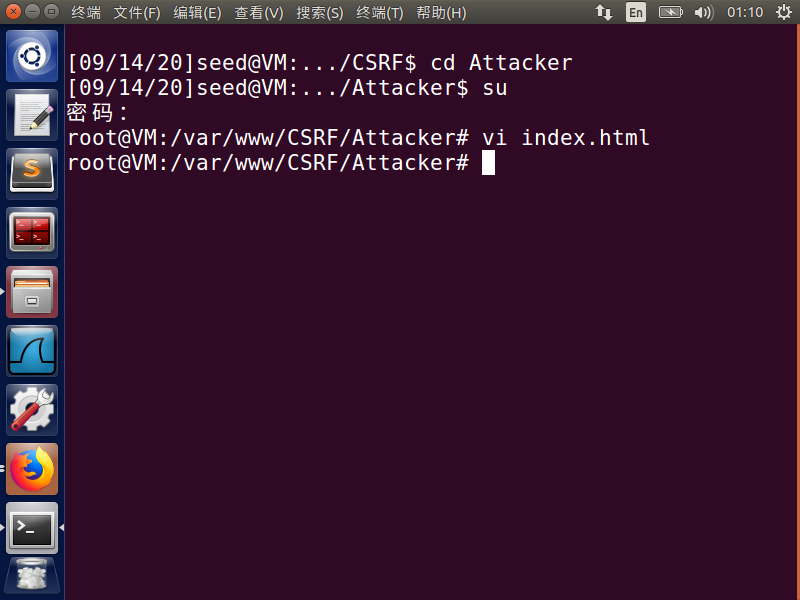
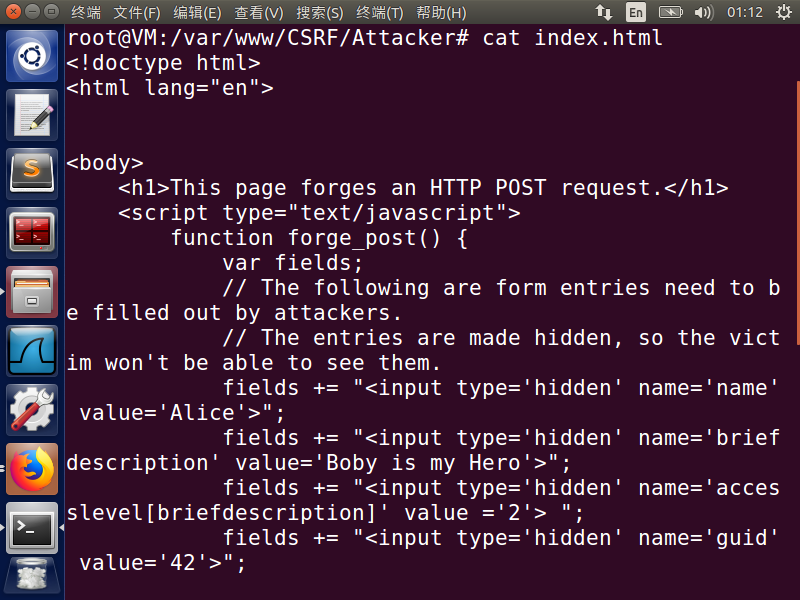
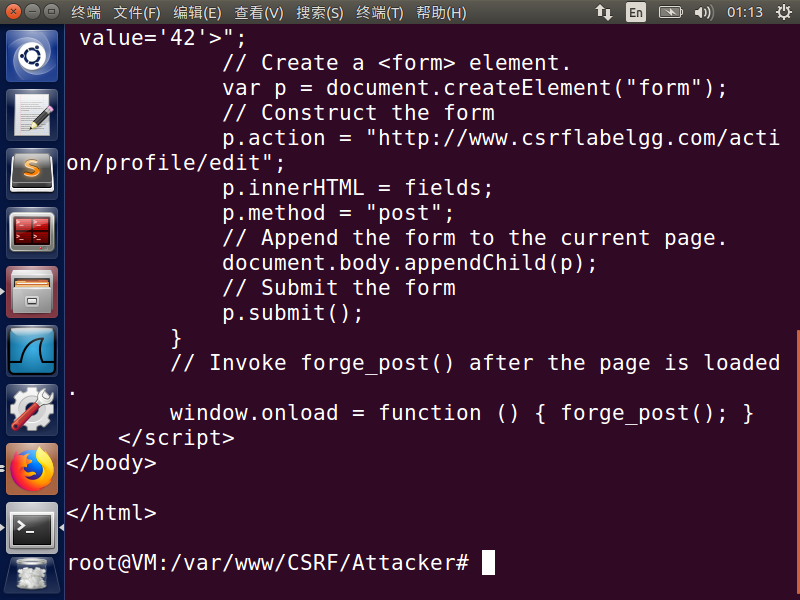
website=

accesslevel[website]=2

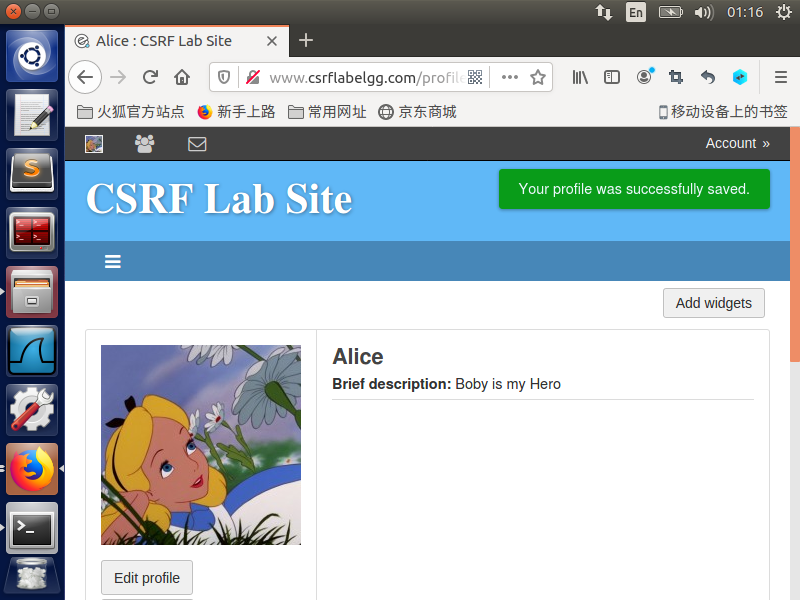
twitter=

accesslevel[twitter]=2

guid=42

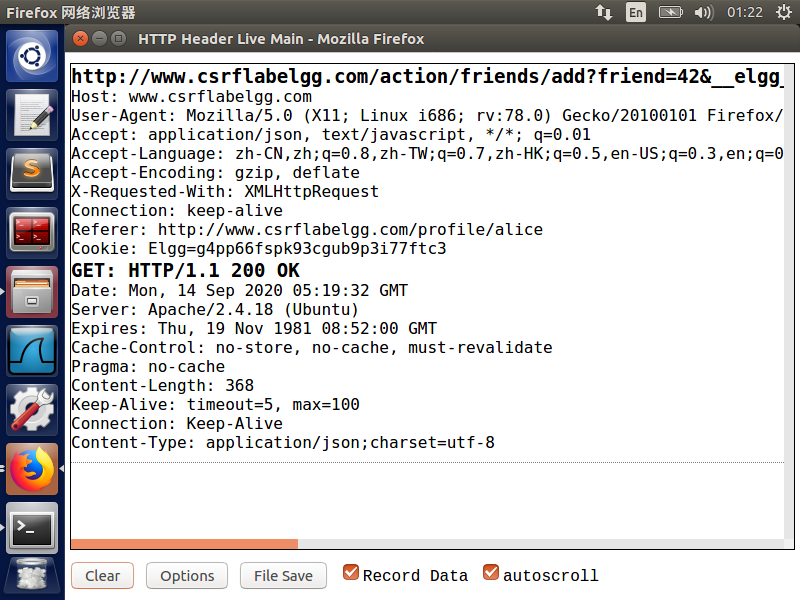
改写原来的index.html文件

点击链接，攻击生效



问题答案：

问题1：

点击添加Alice的好友，返回的GET请求中的add?friend = 42,即为Alice的guid

问题2：不能。因为CSRF攻击是预先写好的，其所构造的参数限制了该攻击只能针对特定的人。

**Task 4: Implementing a countermeasure for Elgg**

Elgg通过使用secret-token approach来抵御CSRF攻击，其利用elgg\_token和elgg\_ts两个参数来作为CSRF攻击的防御对策。在之前的几项任务，该项措施被禁用而使得CSRF攻击能成功实现。现取消禁用来观察攻击是否仍然有效。

开启防御措施：

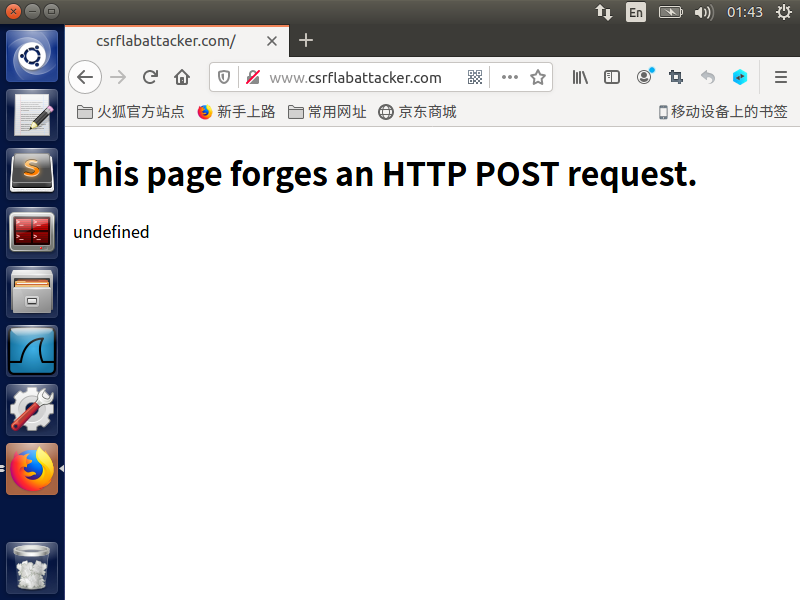
进入/var/www/CSRF/Elgg/vendor/elgg/elgg/engine/classes/Elgg目录

找到ActionsService.php文件

修改函数gatekeeper()，注释掉return true;

然后再观察攻击是否有效



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elgg\_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elgg\_token参数进行检验，那么csrf将无法完成攻击

由其生成代码可以看出

$ts = time();

$token = generate\_action\_token($ts);

echo elgg\_view(’input/hidden’, array(’name’ => ’\_\_elgg\_token’, ’value’ =>

$token));

echo elgg\_view(’input/hidden’, array(’name’ => ’\_\_elgg\_ts’, ’value’ => $ts));

function generate\_action\_token($timestamp)

{

$site\_secret = get\_site\_secret();

$session\_id = session\_id();

// Session token

$st = $\_SESSION[’\_\_elgg\_session’];

if (($site\_secret) && ($session\_id))

{

return md5($site\_secret . $timestamp . $session\_id . $st);

}

return FALSE;

}