**Cross-Site Scripting (XSS) Attack Lab**

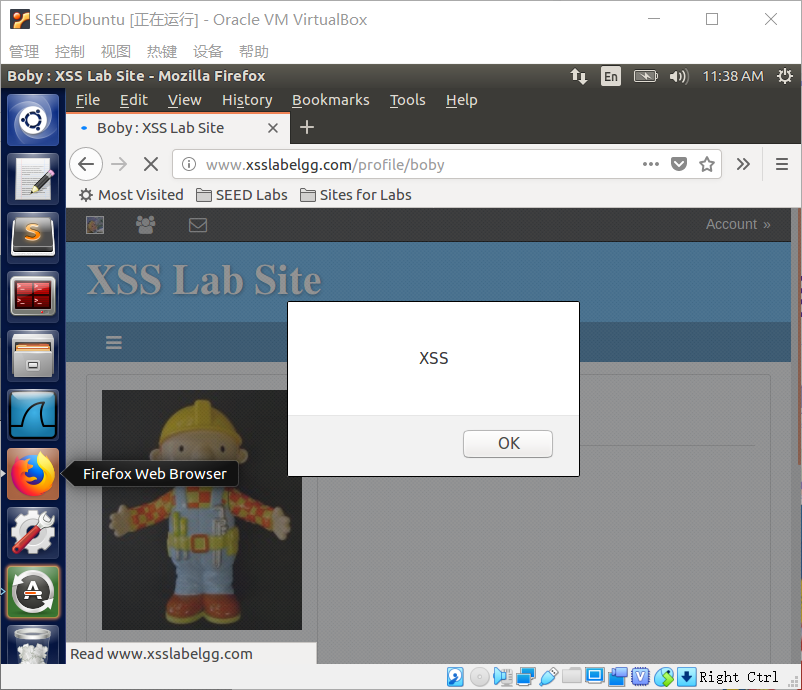
57118135 许皓钦

**Task 1: Posting a Malicious Message to Display an Alert Window**

登录boby的账号，在Brief description域中添加JavaScript代码并保存

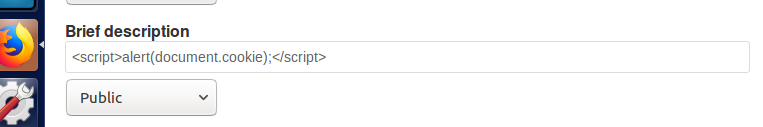


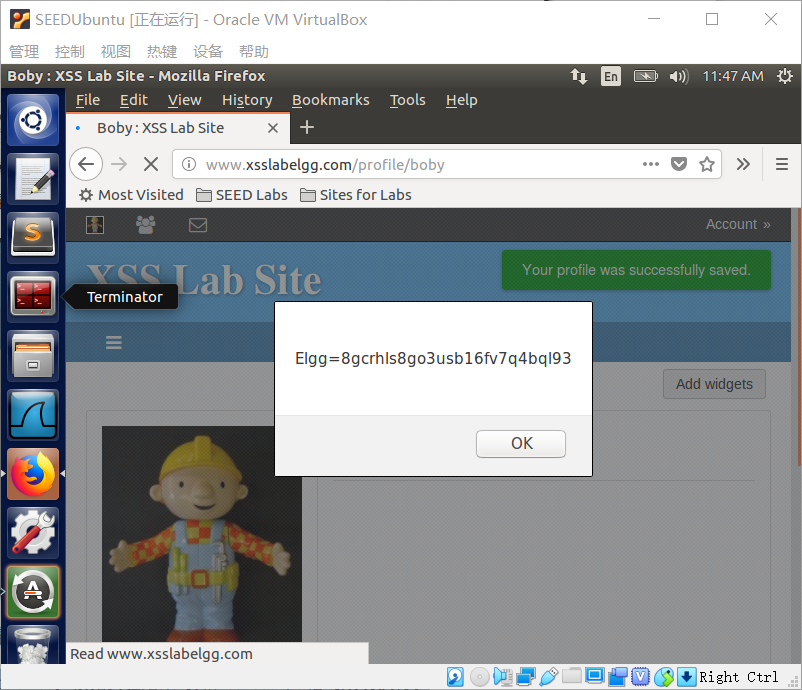
Alice查看Boby的profile，显示警告窗口



**Task 2: Posting a Malicious Message to Display Cookies**

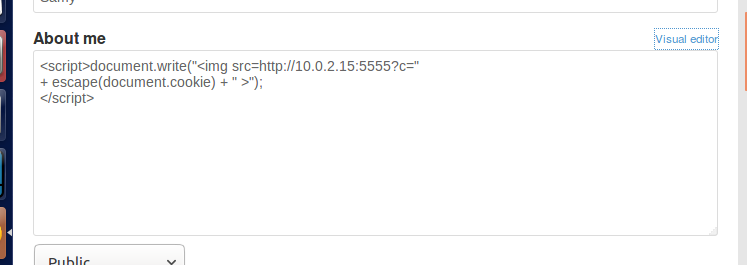
登录 Boby 的账户，修改签名如上图所示。

当Alice查看Boby的profile时，将会在警告框内显示自己的cookies信息。

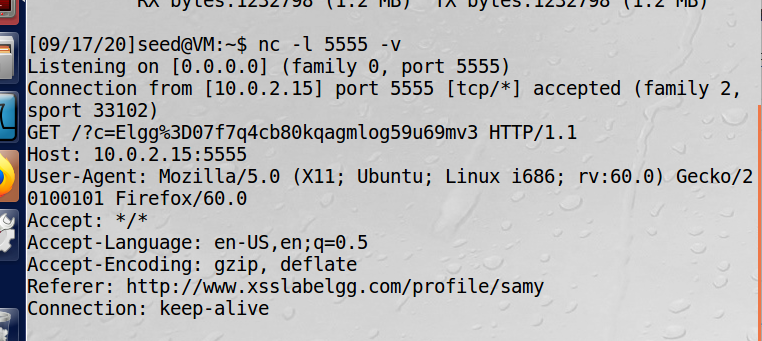


**Task 3: Stealing Cookies from the Victim’s Machine**

登录 Samy 的账户，进入 Edit HTML 界面修改签名如上图所示。

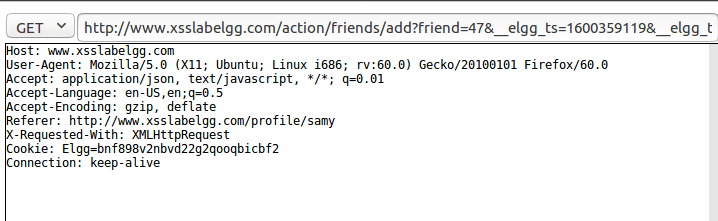


监听5555端口。当用户访问Alice profile时，打印出当前用户的cookies。

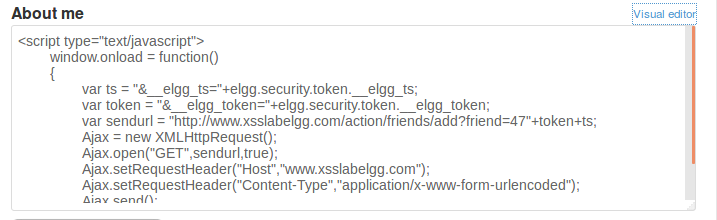


**Task 4: Becoming the Victim’s Friend**

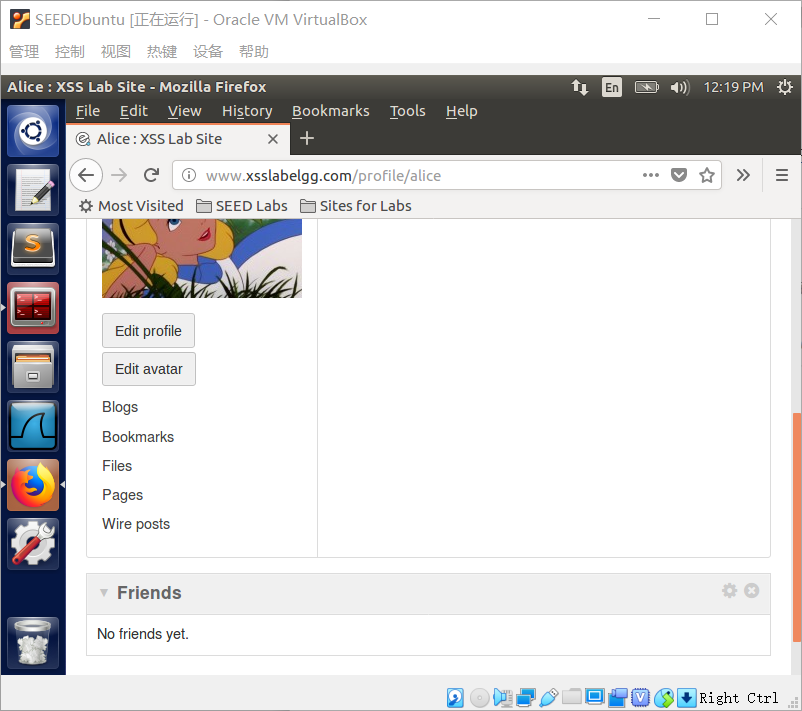
首先使用admin账号登录，并添加samy为好友，抓取的添加好友的 HTTP GET 请求如上图所示，可知 Samy 的用户 id 为 47。



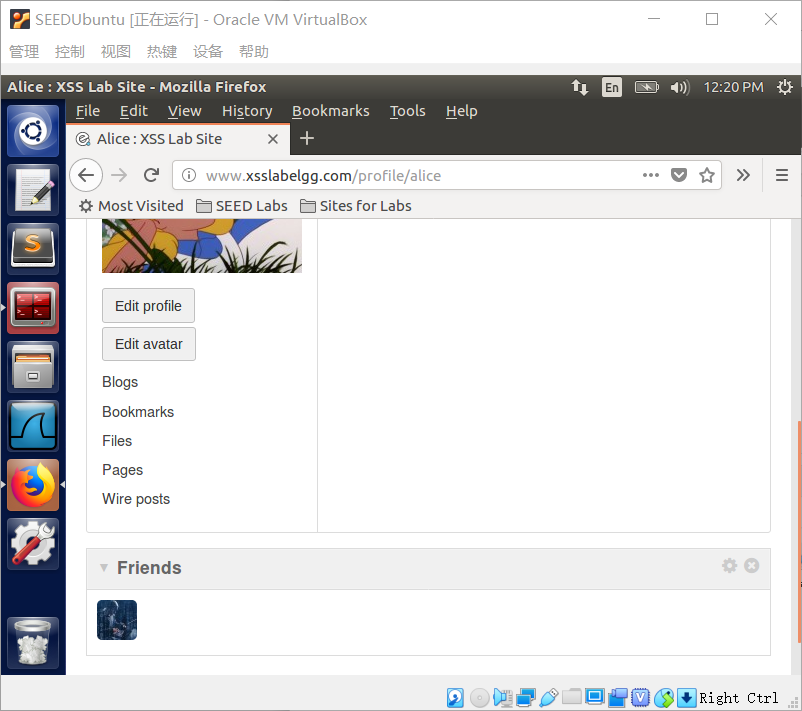
登录 Samy 的账户，进入 Edit HTML 界面修改签名如上图所示。



此时，登录Alice账号，发现好友列表里并无Samy



访问Samy主页后，再查看好友列表，发现已添加Samy为好友



根据实验结果可知，攻击已经生效

问题1

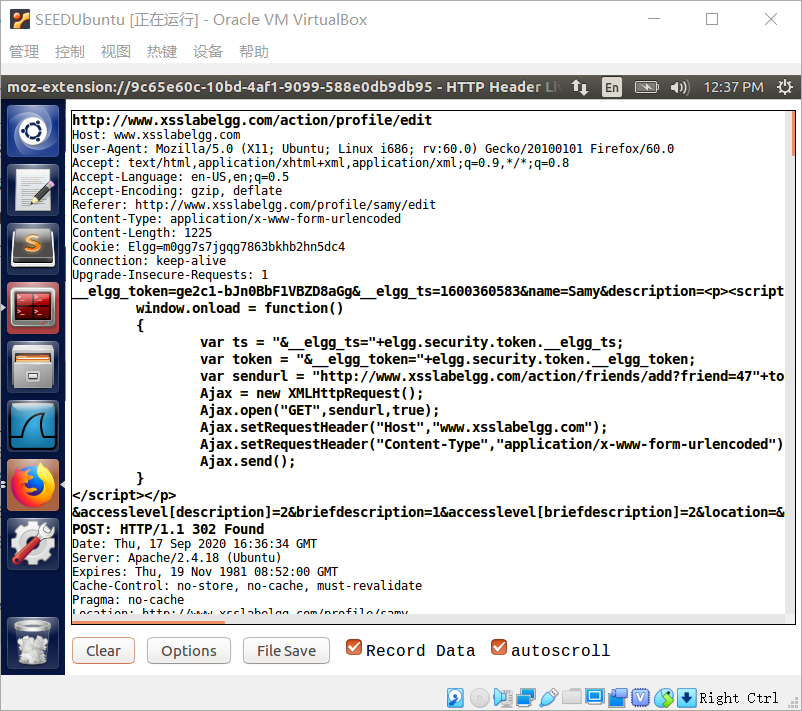
答：Lines ➀ and ➁是为了获取参数\_elgg\_ts 和\_elgg\_token，必须正确设置这两个值，否则它会被丢弃。

问题2

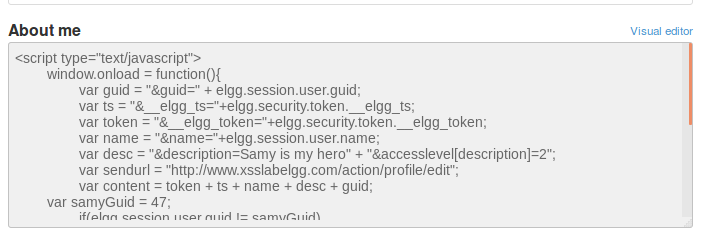
答：能；可以使用浏览器扩展来删除 HTTP 请求中的格式化数据。

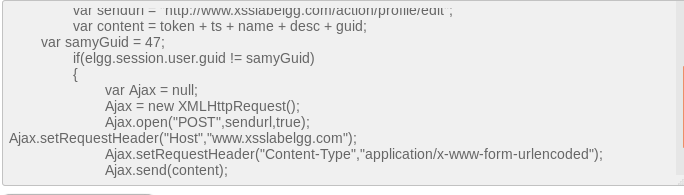
**Task 5: Modifying the Victim’s Profifile**

登录samy账户，点击修改个人资料并保存，使用HTTP Header Live插件获取请求

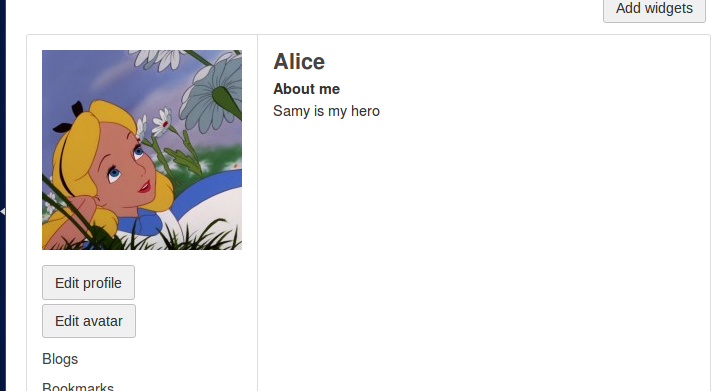


登录 Samy 的账户，进入 Edit HTML 界面修改签名如上图所示。





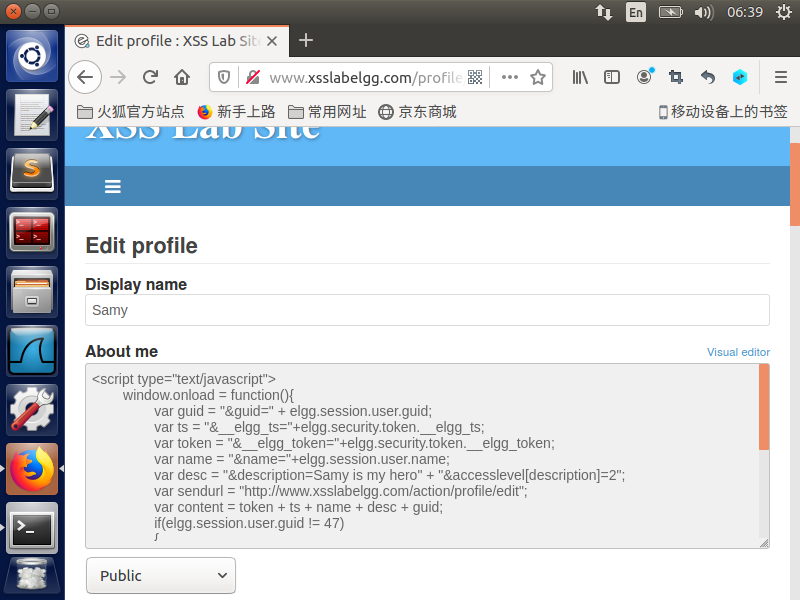
登录 Alice 的账户，点击查看 Samy 的个人信息界面，然后返回自己的个人信息界面，发现签名已经被修改为 Samy is my hero。根据实验结果可知，XSS 攻击成功。



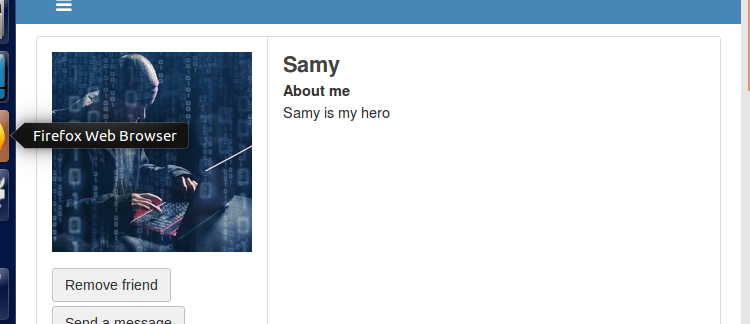
问题3

答：

登录samy账号，进入 Edit HTML 界面修改签名如上图所示。

返回 Samy 的个人信息界面发现自己签名已经被修改为 Samy is my hero，而其

他用户点击查看 Samy 的个人信息界面后签名不会被修改。

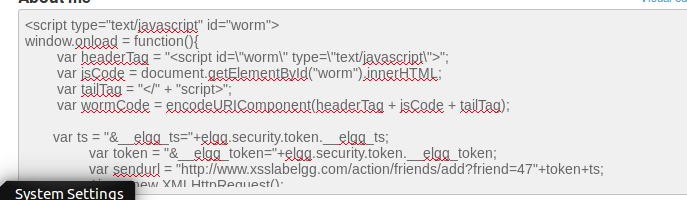


原因是 Line1 的代码负责判断用户是否为 Samy 本人，如果不对攻击者本身的guid进行排除，那么攻击首先会生效在攻击者本人身上。

**Task 6: Writing a Self-Propagating XSS Worm**

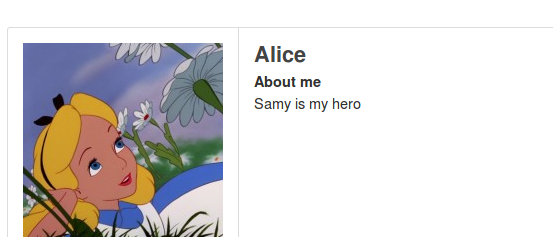
**DOM Approach**

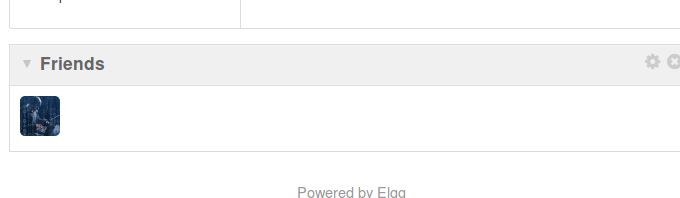
登录 Samy 的账户，进入 Edit HTML 界面修改签名如上图所示。



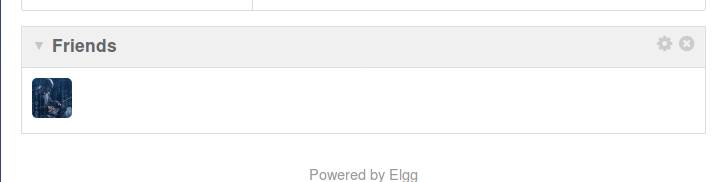
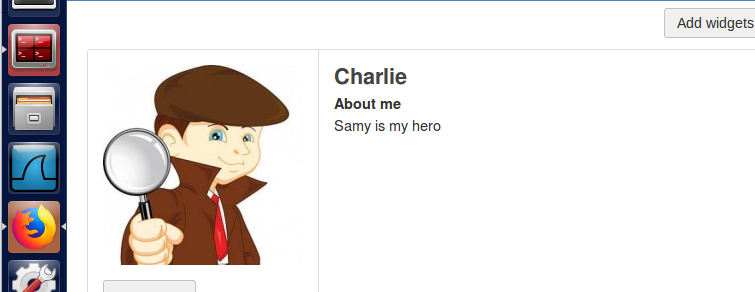


登录Alice账号，点击查看samy的个人资料，然后返回查看自己的资料



可以看到个人资料已被修改，同时添加了Samy为好友.

登录Charlie的账号，查看Alice的个人资料，然后返回查看自己的个人资料

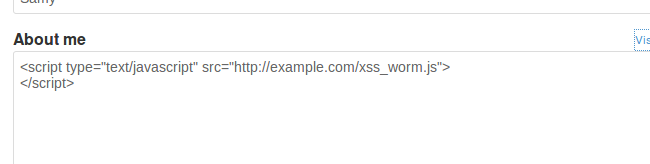


其个人资料也被修改了，同时添加了samy为好友

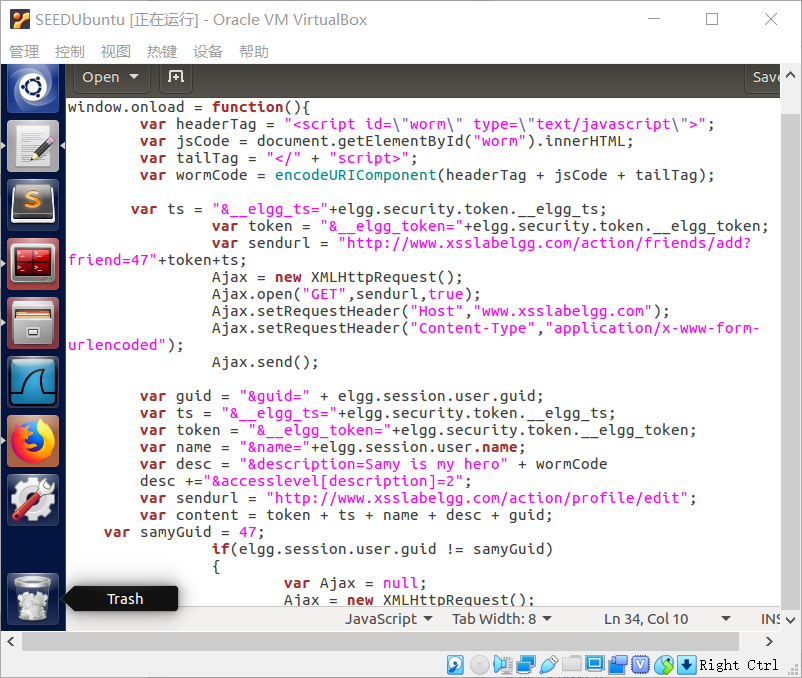
根据实验结果可知，通过 DOM方法，XSS蠕虫攻击成功并且具有自我传播能力。

**Link Approach**

登录 Samy 的账户，进入 Edit HTML 界面修改签名如下图所示

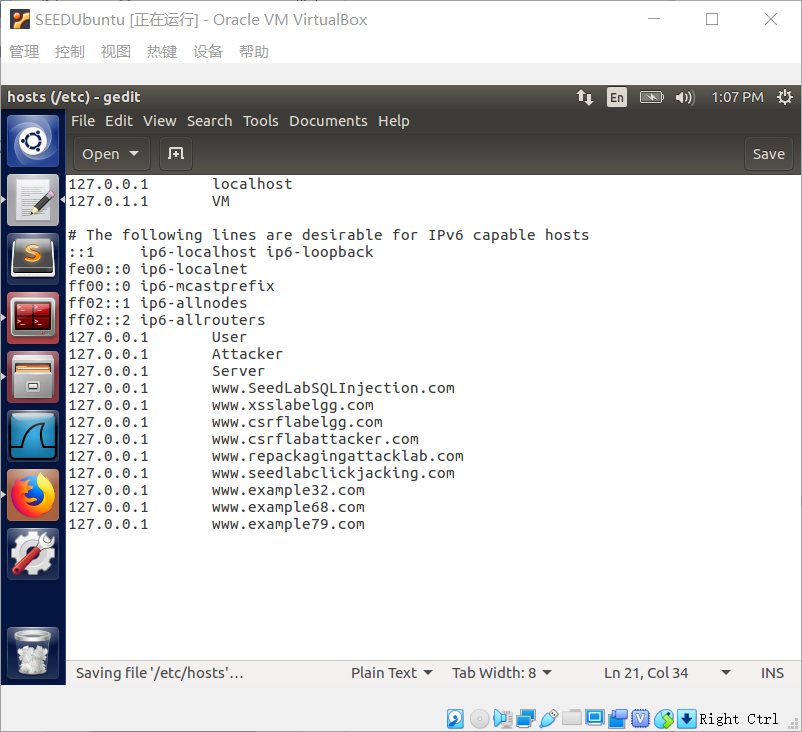


在/var/www/XSS/Elgg 下创建文件 xss\_worm.js，并且添加代码如下图所示。



再次执行与DOM APPROACH相同的操作，发现通过 Link方法，XSS 蠕虫攻击同样成功并且具有自我传播能力

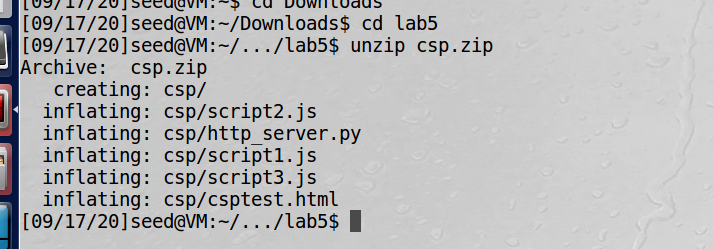
**Task 7: Defeating XSS Attacks Using CSP**



修改/etc/hosts 文件，添加语句如上图所示。

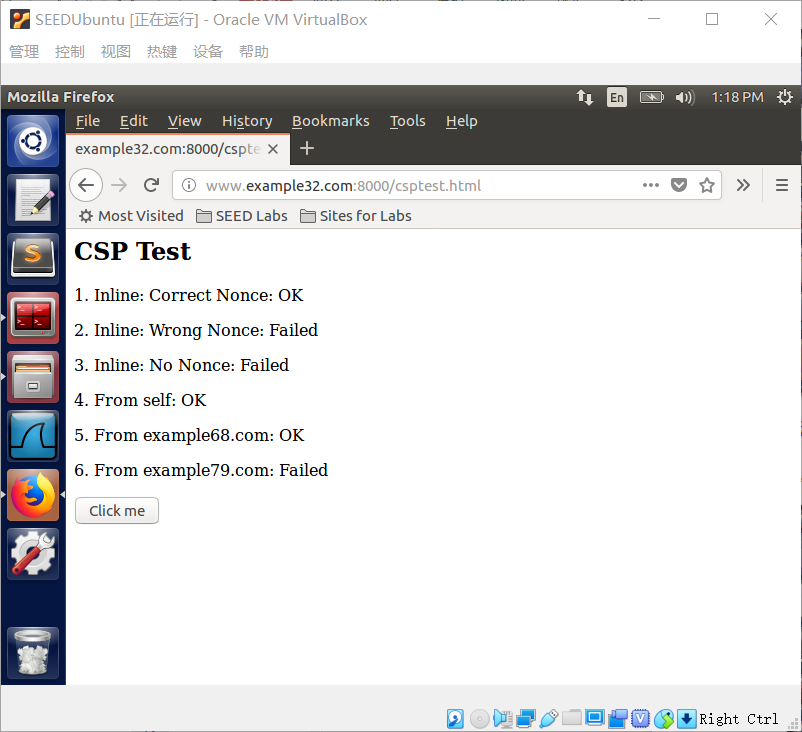
在 https://seedsecuritylabs.org/Labs\_16.04/Web/Web\_XSS\_Elgg/中下载 csp.zip

文件，然后进行解压，再运行其中的 http\_server.py 文件。



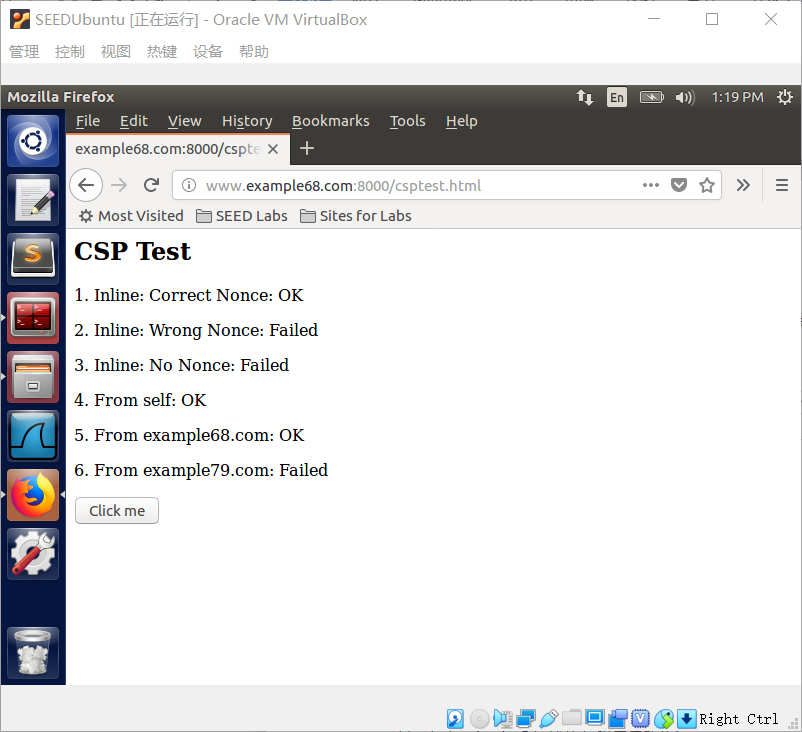
在浏览器中输入 http://www.example32.com:8000/csptest.html，页面如上图所

示，Fields 1、4、5 显示为 OK。



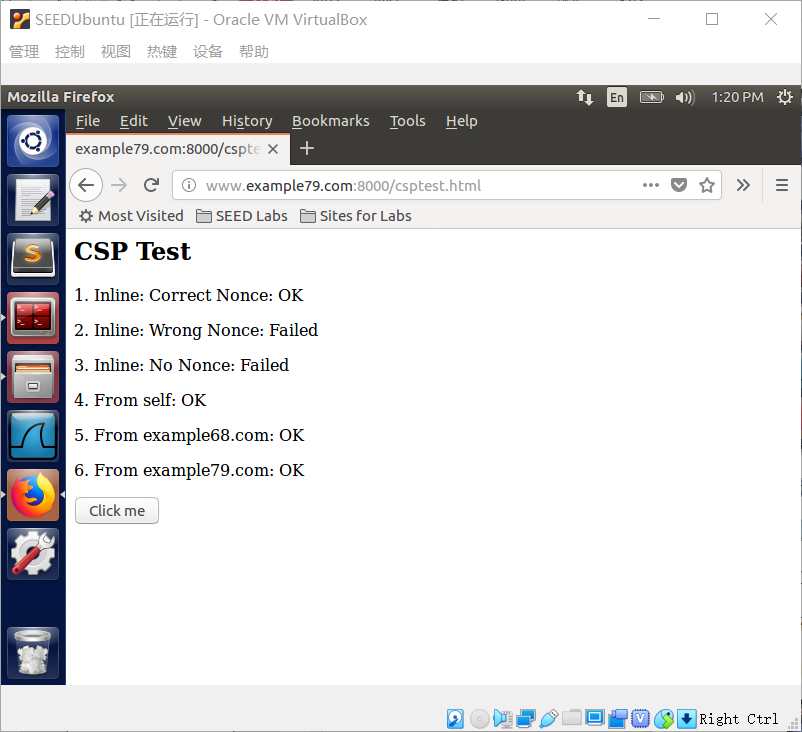
在浏览器中输入 http://www.example68.com:8000/csptest.html，页面如上图所

示，Fields 1、4、5 显示为 OK。

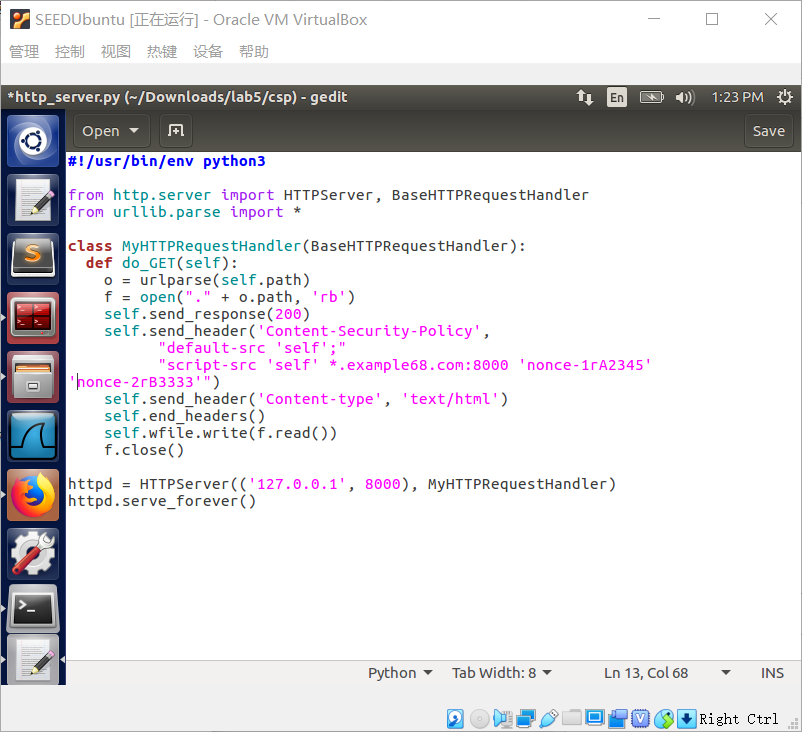


在浏览器中输入 http://www.example79.com:8000/csptest.html，页面如上图所

示，Fields 1、4、5、6 显示为 OK。



修改 http\_server.py 文件，在 script-src 后添加 nonce-2rB3333。



重新编译运行修改后的 http\_server.py，再次在浏览器中输入

<http://www.example79.com:8000/csptest.html>

页面如下图所示，Fields 1、2、4、5、6 显示为 OK。

