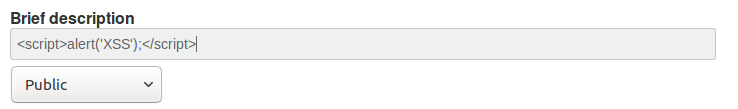
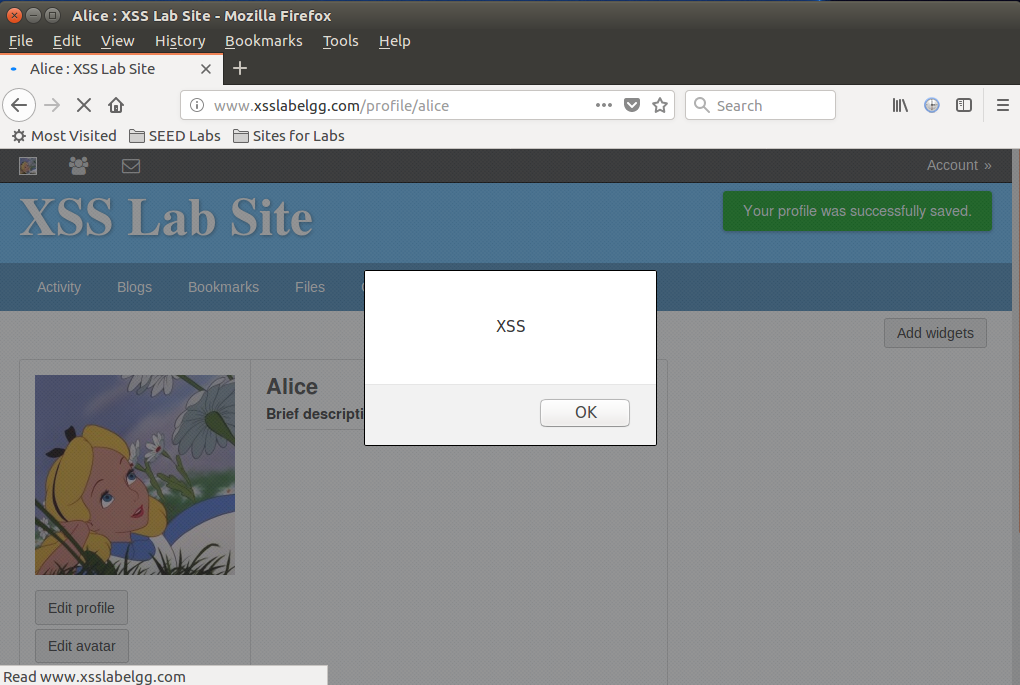
Task1

在[www.xssblgg.com](http://www.xssblgg.com)网站中登陆账号，在brief description中插入script脚本



保存后浏览器返回之前界面并弹出弹窗

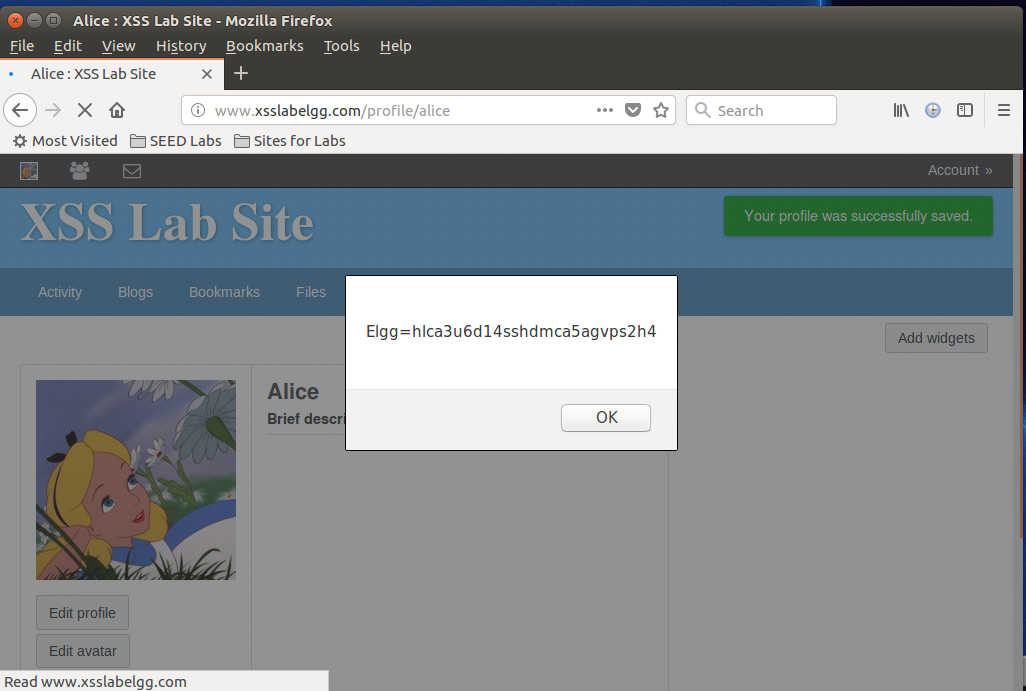


Task2

获得查看配置文件用户的cookie，在brief description中输入如下代码

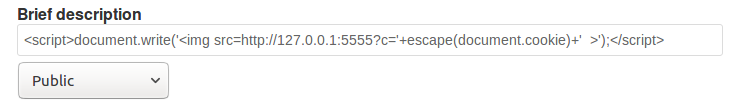


保存后即能看见查看用户的cookie



Task3

如果将cookie发送给攻击者，则需要输入如下代码

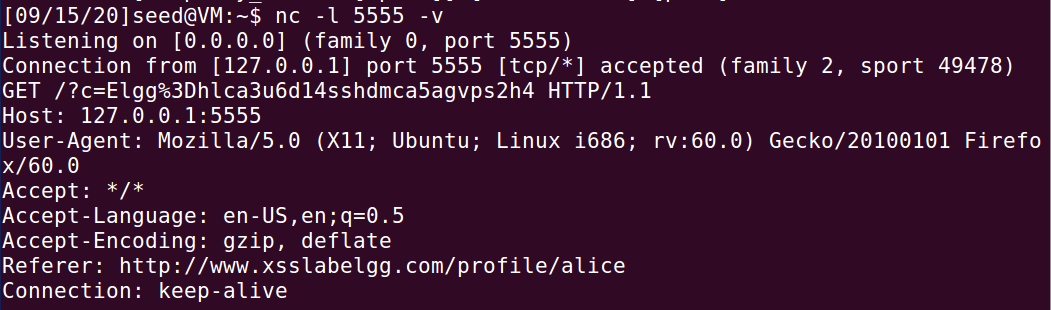


其中一台虚拟机时IP地址为127.0.0.1，两台虚拟机时IP地址为10.1.2.5

在terminal中输入如下监听命令



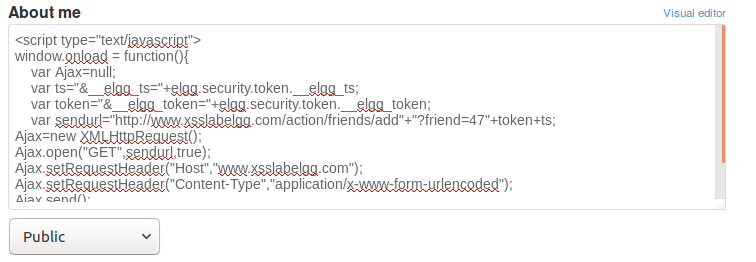
之后保存profile中的内容后terminal中监听到包含cookie的get报文信息



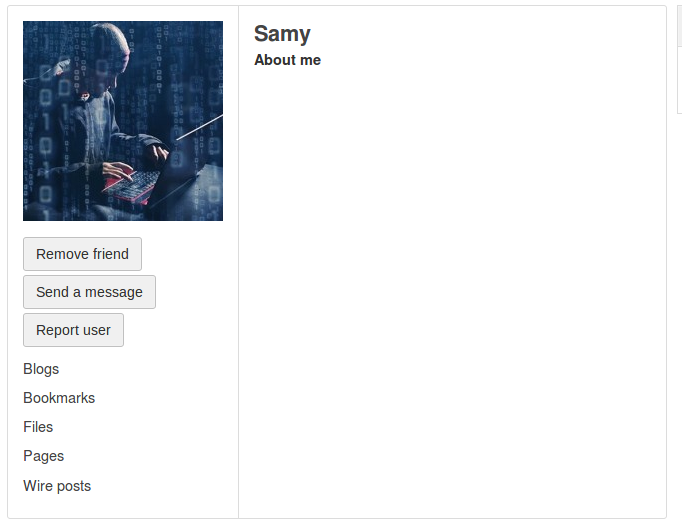
Task4

先添加samy为好友，用HTTP Header Live工具捕获报文，获得Samy的id为47.

登陆Samy账号，在About me的Edit HTML中(由于不会添加额外文本)输入自动添加好友的JavaScript代码，其中需将url部分补充完整



在登陆任意用户的账号，点进Samy界面后刷新，发现已经添加为好友

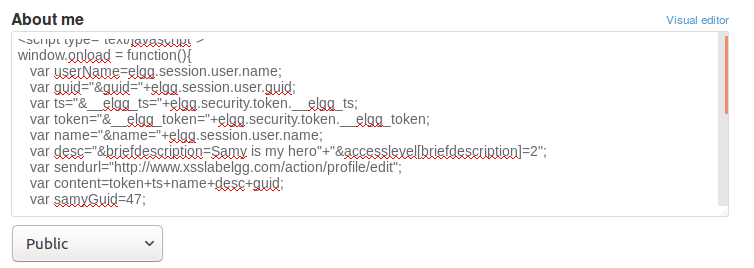


**Q1**：Line1和Line2是为了获取令牌ts和token，因为只有发送的请求中含有正确的令牌才会被接受

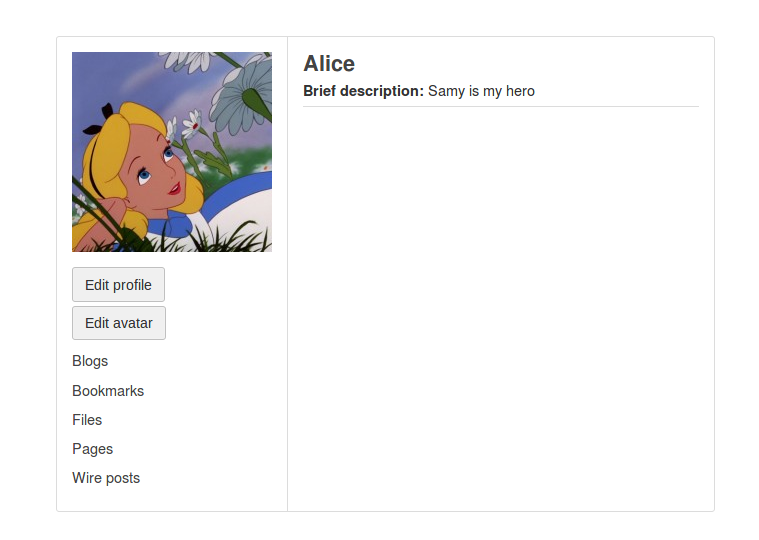
**Q2**：可以通过使用浏览器扩展来删除请求中的格式化数据

Task5：

登陆Samy账号，在About me的Edit HTML中(由于不会添加额外文本)输入修改受害者briefdescription的javascript代码



登入受害者账号，进入Samy主页后刷新，再次回到自己主页后发现brief description被修改，证明攻击成功

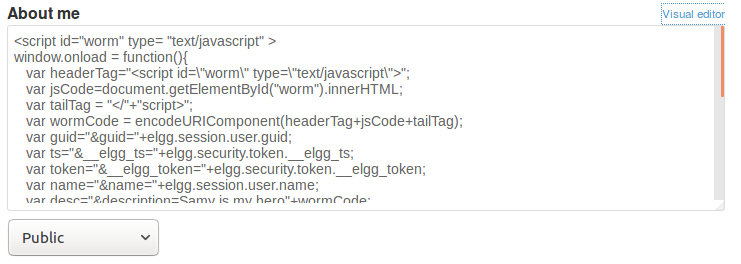


**Q3**：Line1的作用是保证不会修改掉攻击者自己的主页，如果没有line1则会覆盖掉原有的

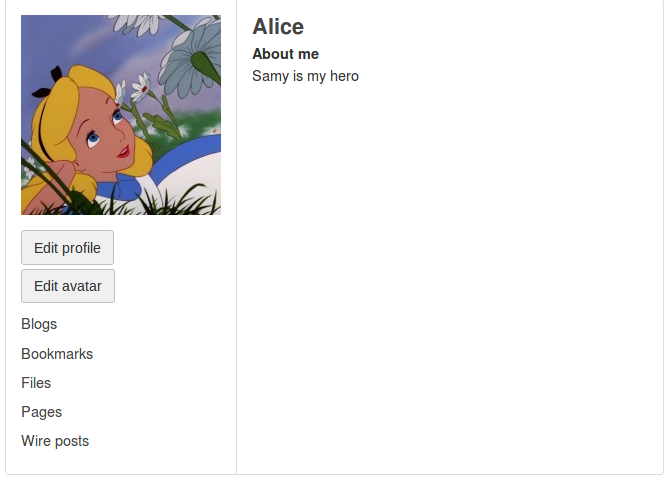
代码，自己的签名会被修改，也无法对用户实现XSS攻击

Task6：

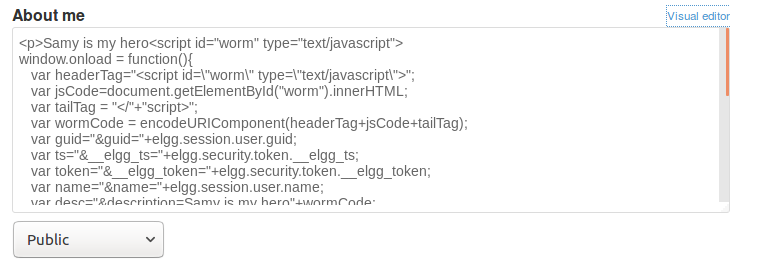
**DOM Approach**：在Samy的About me中输入如下代码



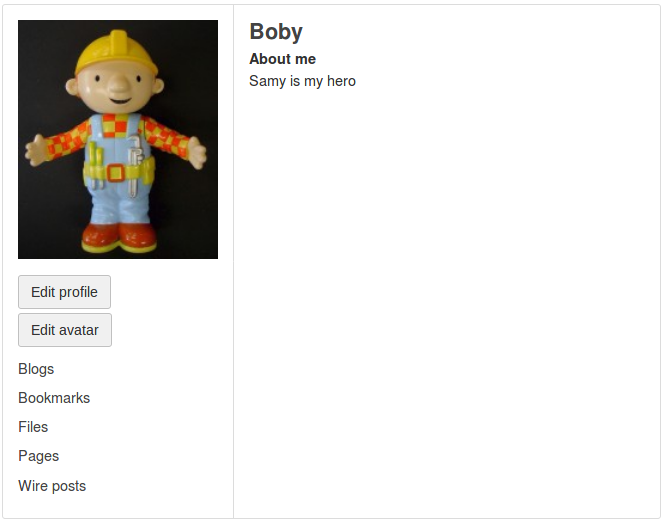
登陆Alice的账号访问Samy的主页后返回自己主页，发现description已被篡改



进入编辑界面发现代码也被成功复制，证明该蠕虫可以进行自繁殖



在登入Boby的账号进入Alice主页，返回自己主页后发现自己主页信息也被篡改



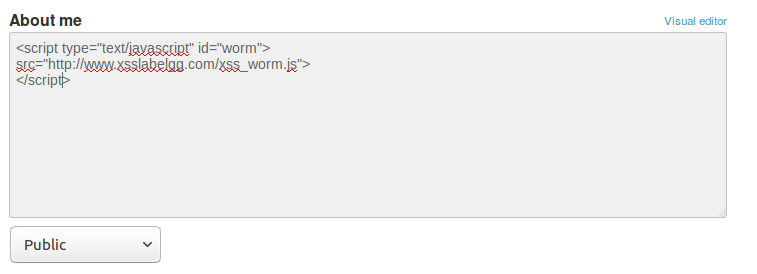
验证成功

**Link Approach**：

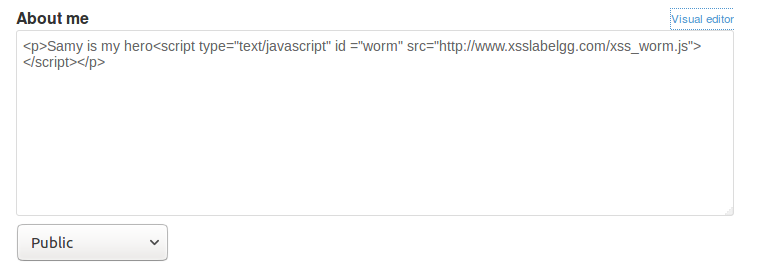
先在var/www/XSS/Elgg文件中创建worm\_worm.js文件，并输入如下代码



然后在Samy的描述界面输入如下脚本link xss\_worm.js文件

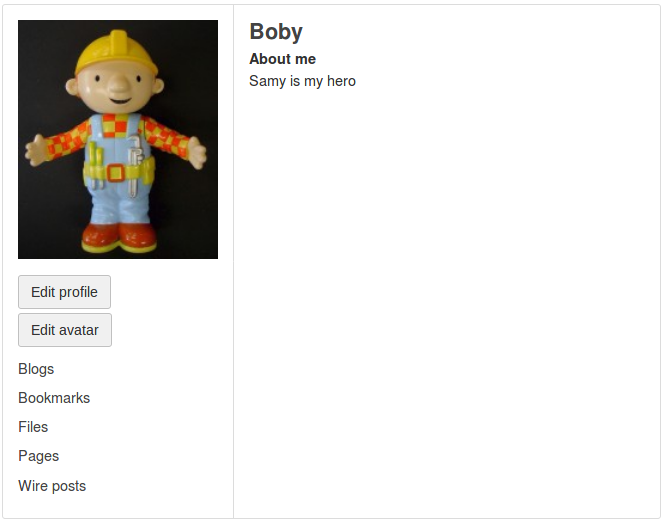


保存后登入Alice账号，进入Samy主页后返回自己主页，发现页面已经被篡改，且编辑界面被嵌入如下代码



证明该蠕虫确实有自我繁殖能力

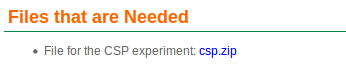
再登入Boby账号，进入Alice主页，返回自己主页后发现自己的主页也被修改



验证成功

Task7

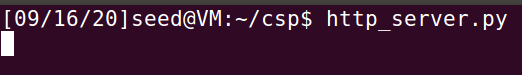
先在网站<https://seedsecuritylabs.org/Labs_16.04/Web/Web_XSS_Elgg/> 中下载csp.zip



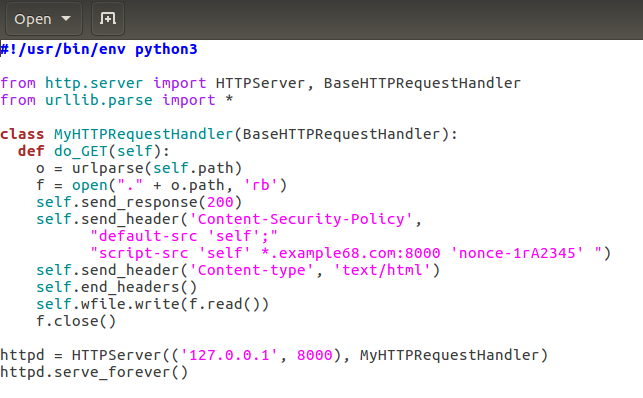
修改hosts文件，添加如下语句



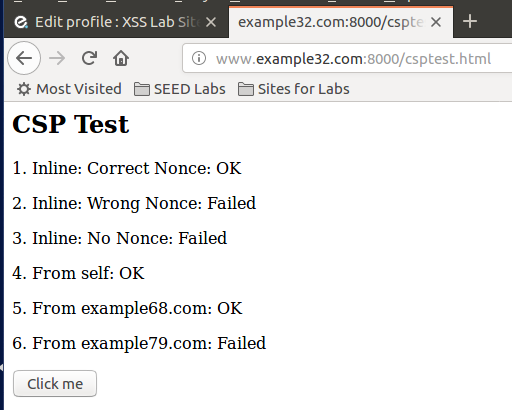
解压后编译并运行http\_server.py文件



http\_server.py代码如下

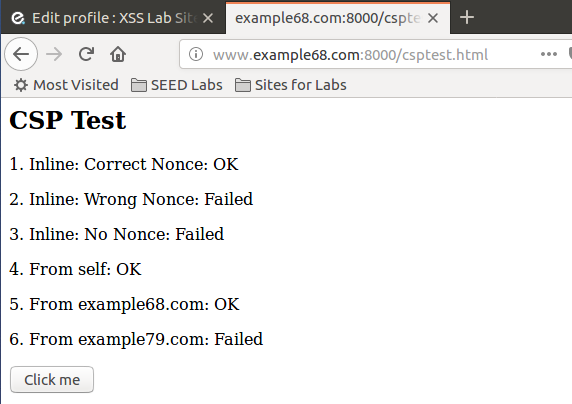


浏览器中输入<http://www.example32.com:8000/csptest.html>，结果如下



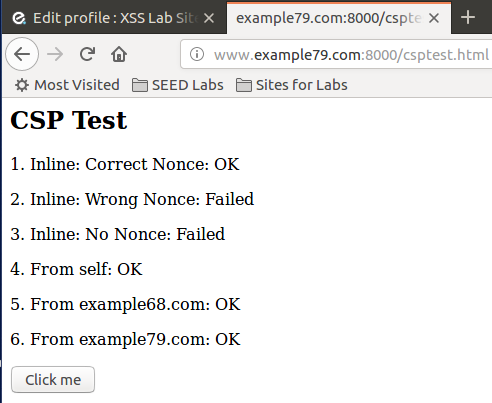
1，4，5，显示为OK

输入<http://www.example68.com:8000/csptest.html>，结果如下



1，4，5显示为OK

输入<http://www.example79.com:8000/csptest.html>，结果如下



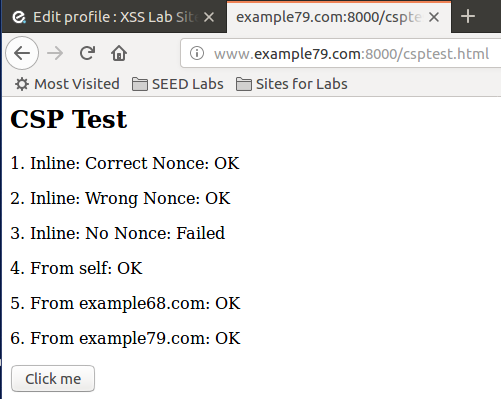
1，4，5，6显示为OK

1显示为OK是因为http\_server.py允许含nonce=1rA2345的代码被执行；4OK是http\_server允许自身代码被执行；5和6显示OK原因分别是允许、来自example68.com:8000的代码和来自example79.com:8000的代码被执行。

修改http\_server文件，添加nonce-2rB3333



重新编译并运行http\_server.py并运行，在浏览器中输入网站<http://www.example79.com:8000/csptest.html>，结果如下



1，2，3，4，5，6均显示为OK