xss-labs 通关笔记

原创

托马斯回旋 于 2020-06-30 10:51:52 发布 154 收藏 1

文章标签： 安全 web xss

本文链接：https://blog.csdn.net/Ewige/article/details/107014185

靶场地址：传送门

概述：

XSS（跨站脚本）概述

Cross-Site Scripting 简称为“CSS”，为避免与前端叠成样式表的缩写"CSS"冲突，故又称XSS。一般XSS可以分为如下几种常见类型：

1.反射性XSS;

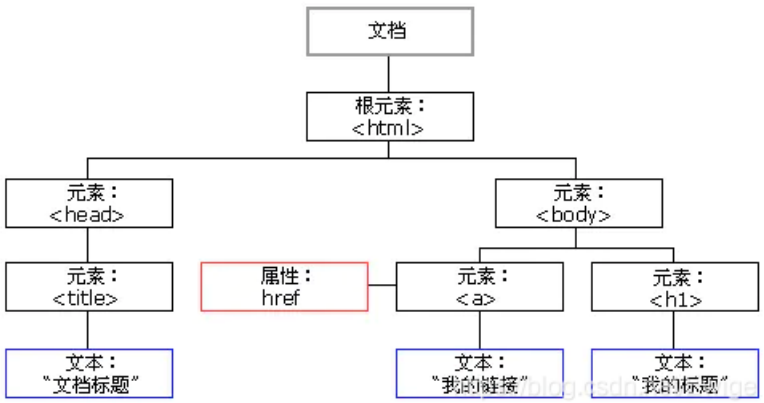
> 反射型XXS是一种非持久性的攻击，它指的是恶意攻击者往Web页面里插入恶意代码，当用户浏览该页之时，嵌入其中Web里面的html代码会被执行，从而达到恶意攻击用户的目的。这里插入的恶意代码并没有保存在目标网站，需要引诱用户点击一个链接到目标网站的恶意链接来实施攻击。
> 原文链接：https://www.jianshu.com/p/a15c411e4c92

2.存储型XSS;

> 什么是存储型XSS：
> 攻击者事先将恶意代码上传或储存到漏洞服务器中，只要受害者浏览包含此恶意代码的页面就会执行恶意代码。这就意味着只要访问了这个页面的访客，都有可能会执行这段恶意脚本，因此储存型XSS的危害会更大。因为存储型XSS的代码存在于网页的代码中，可以说是永久型的。
> 原文链接：https://blog.csdn.net/weixin_44720762/article/details/89736508

3.DOM（Document）型XSS;

先以一张w3c的图来说明，到底什么是dom：



> dom就是一个树状的模型，你可以编写Javascript代码根据dom一层一层的节点，去遍历/获取/修改对应的节点，对象，值。了解了这么一个知识点，你就会发现，其实dom xss并不复杂，他也属于反射型xss的一种(domxss取决于输出位置，并不取决于输出环境，因此domxss既有可能是反射型的，也有可能是存储型的)，简单去理解就是因为他输出点在DOM。dom - xss是通过url传入参数去控制触发的）
> 链接：https://www.jianshu.com/p/190dedd585f2

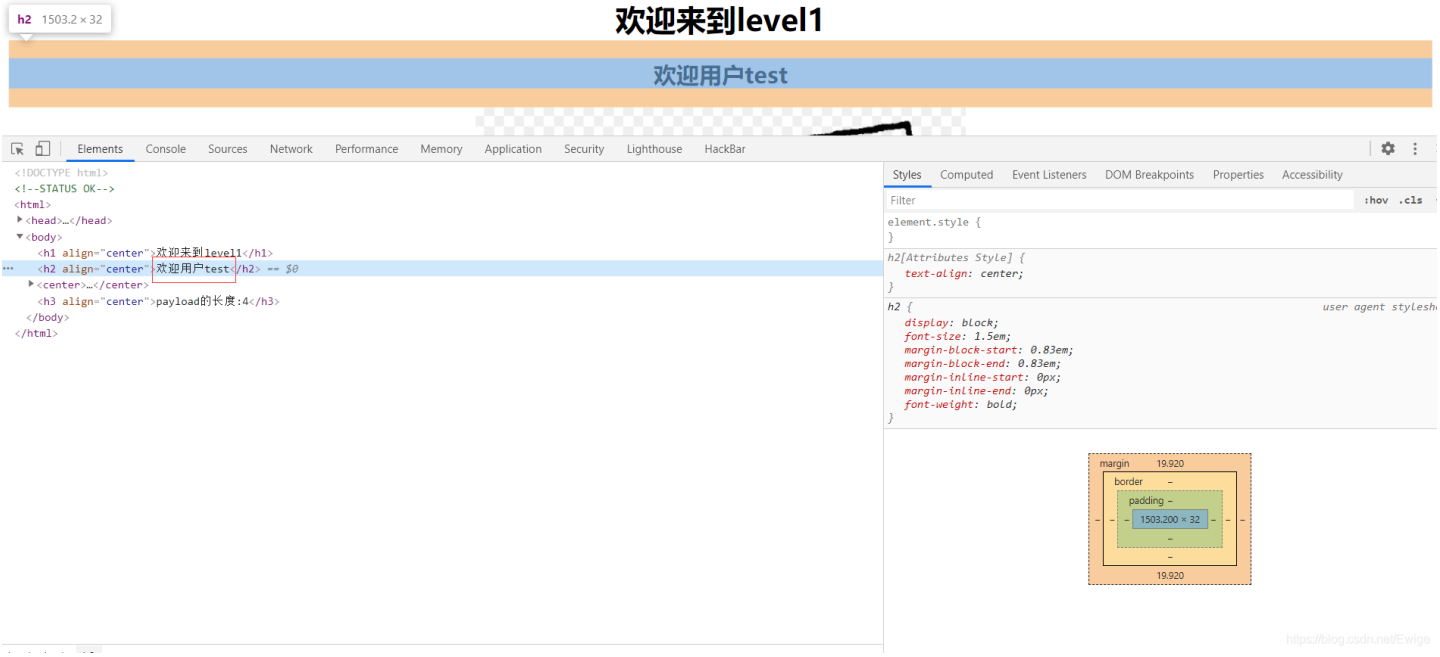
总之：就是向目标注入恶意的javascript代码。目前只能做出这样浅薄的思考。

XSS漏洞一直被评估为web漏洞中危害较大的漏洞，在OWASP TOP10的排名中一直属于前三的江湖地位。
XSS是一种发生在前端浏览器端的漏洞，所以其危害的对象也是前端用户。
形成XSS漏洞的主要原因是程序对输入和输出没有做合适的处理，导致“精心构造”的字符输出在前端时被浏览器当作有效代码解析执行从而产生危害。
因此在XSS漏洞的防范上，一般会采用“对输入进行过滤”和“输出进行转义”的方式进行处理:
输入过滤：对输入进行过滤，不允许可能导致XSS攻击的字符输入;
输出转义：根据输出点的位置对输出到前端的内容进行适当转义;

这里写目录标题

level 1

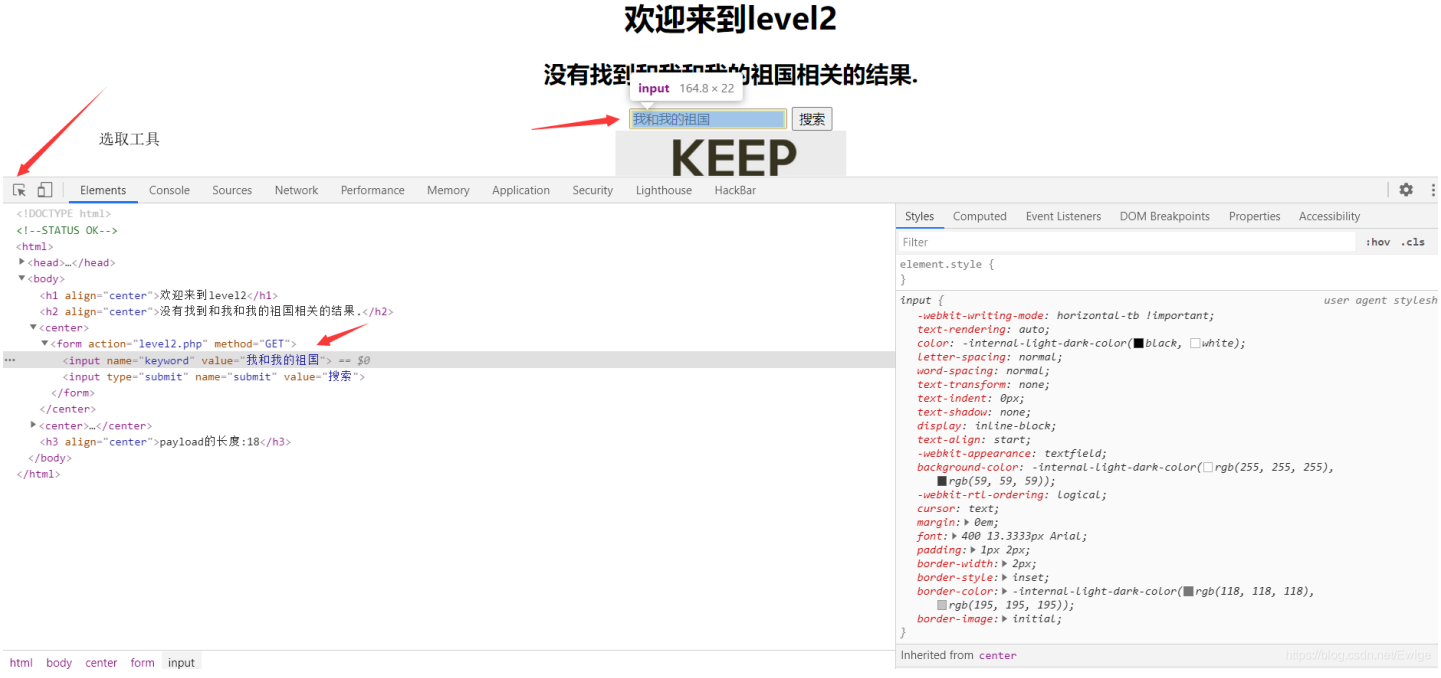
注意观察我们可以发现这几个点



这是一个GET交互的xss漏洞
URL的name传参就是我们的注入点，构造payload
将test 替换为 达到注入的目的进入下一关。

level2

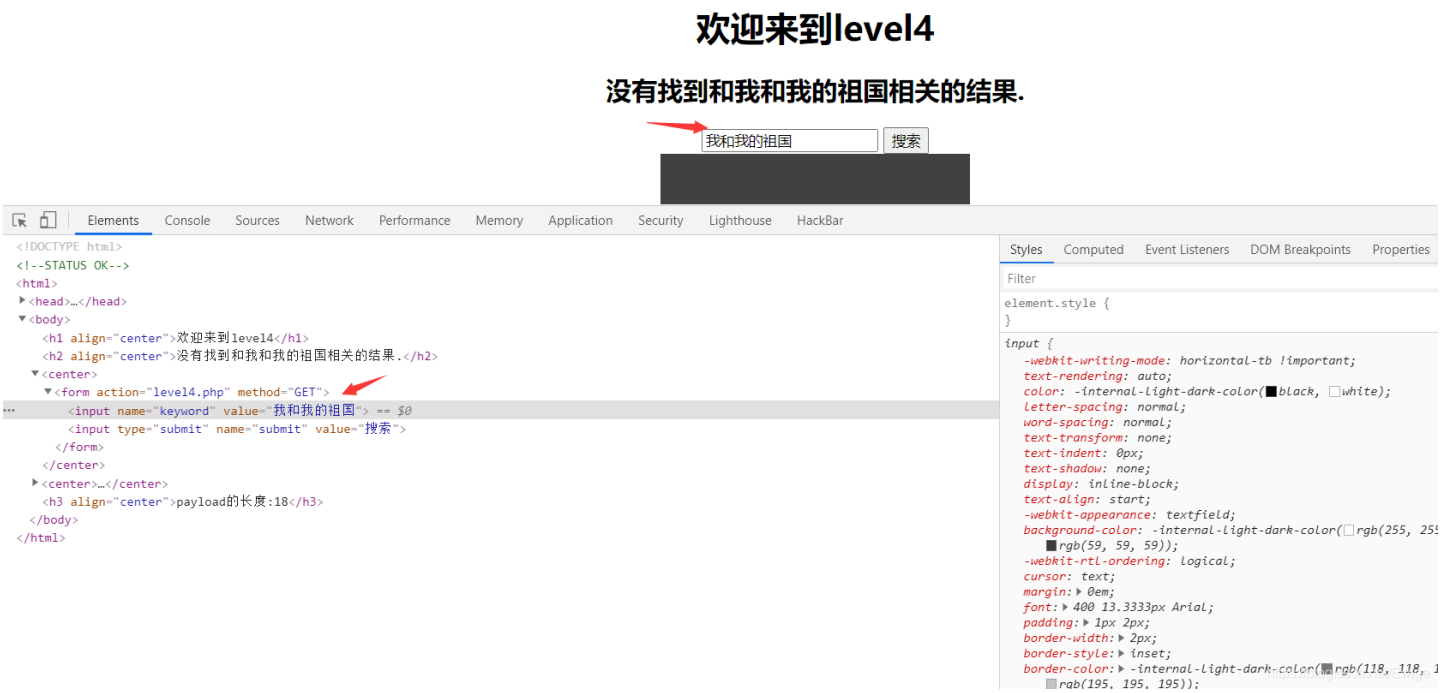
有输的地方就有可能存在注入，
输入一些值然后用选取工具观察一下，发现被填入了
input 的value值里面



对其进行闭合绕过
payload：">

level3

和level2 一样需要构造闭合



但是一顿操作之后发现搞不定，查看一下源码

发现唯一显示$str的地方都做了html实体化，也就是过滤了<>

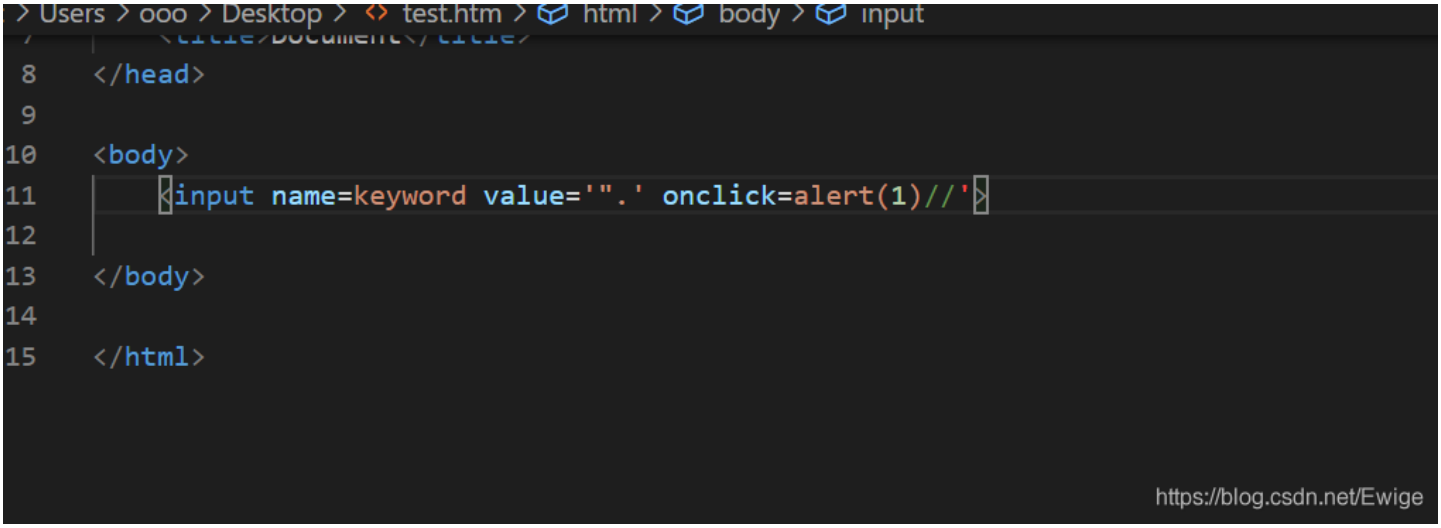
这个时候我们可以借助点击事件onclick

onclick事件会在元素被点击时发生

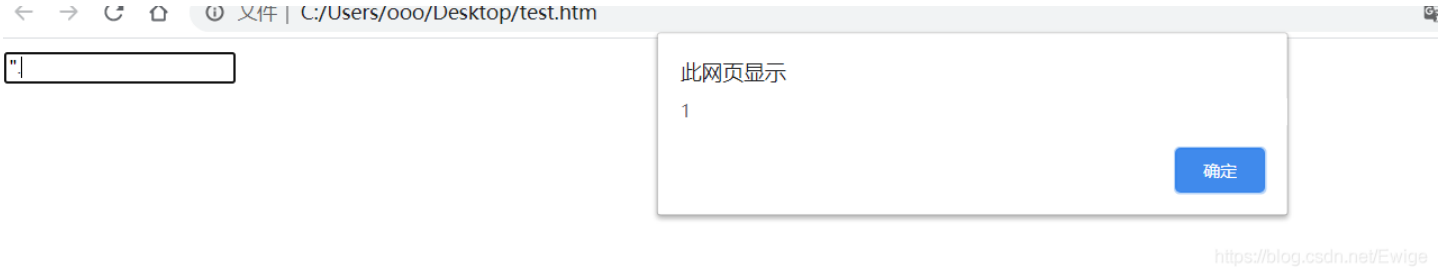
鼠标其他事件

用 ' 闭合前面的单引号 用 // 注释后面的代码

构造payload：** ' onclick=alert(1) //**
经过处理后，在代码中的情况：

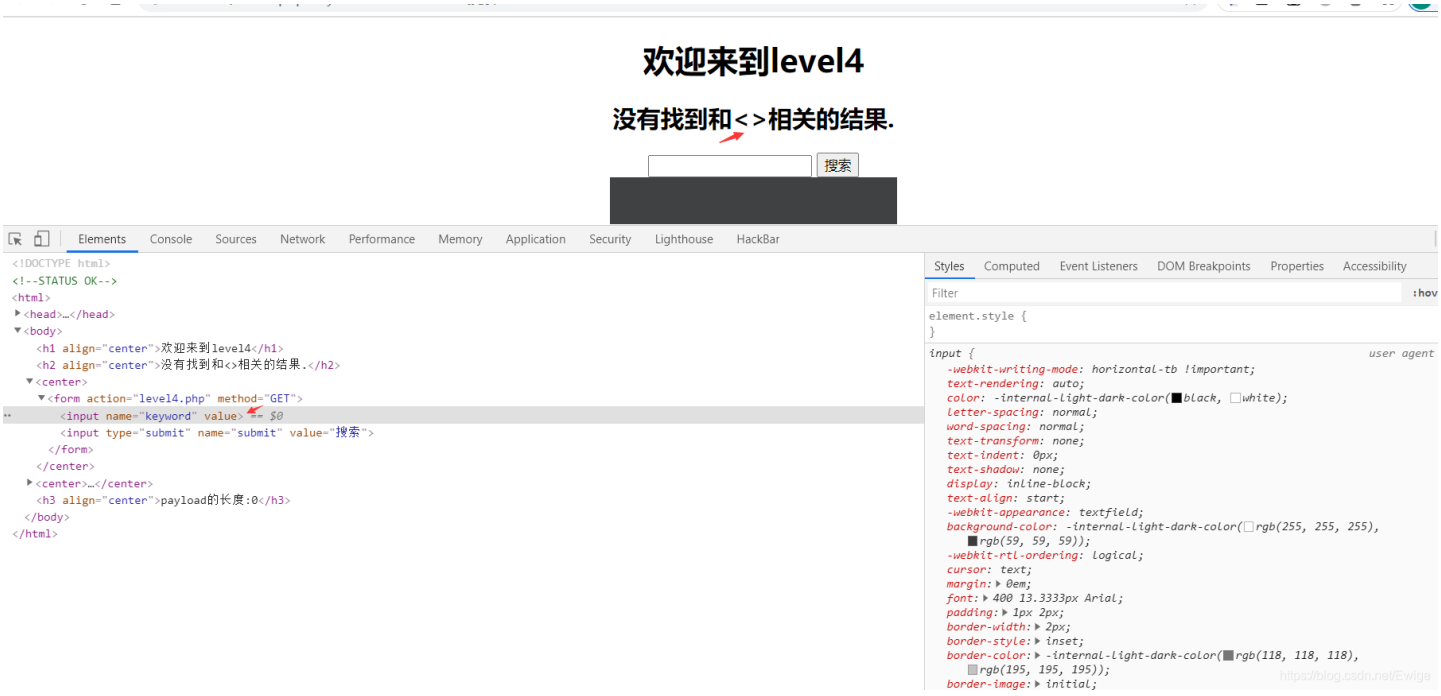


因为此点击事件是包含在input输入框中的，所以我们在输入框中点击鼠标就会触发事件



level4

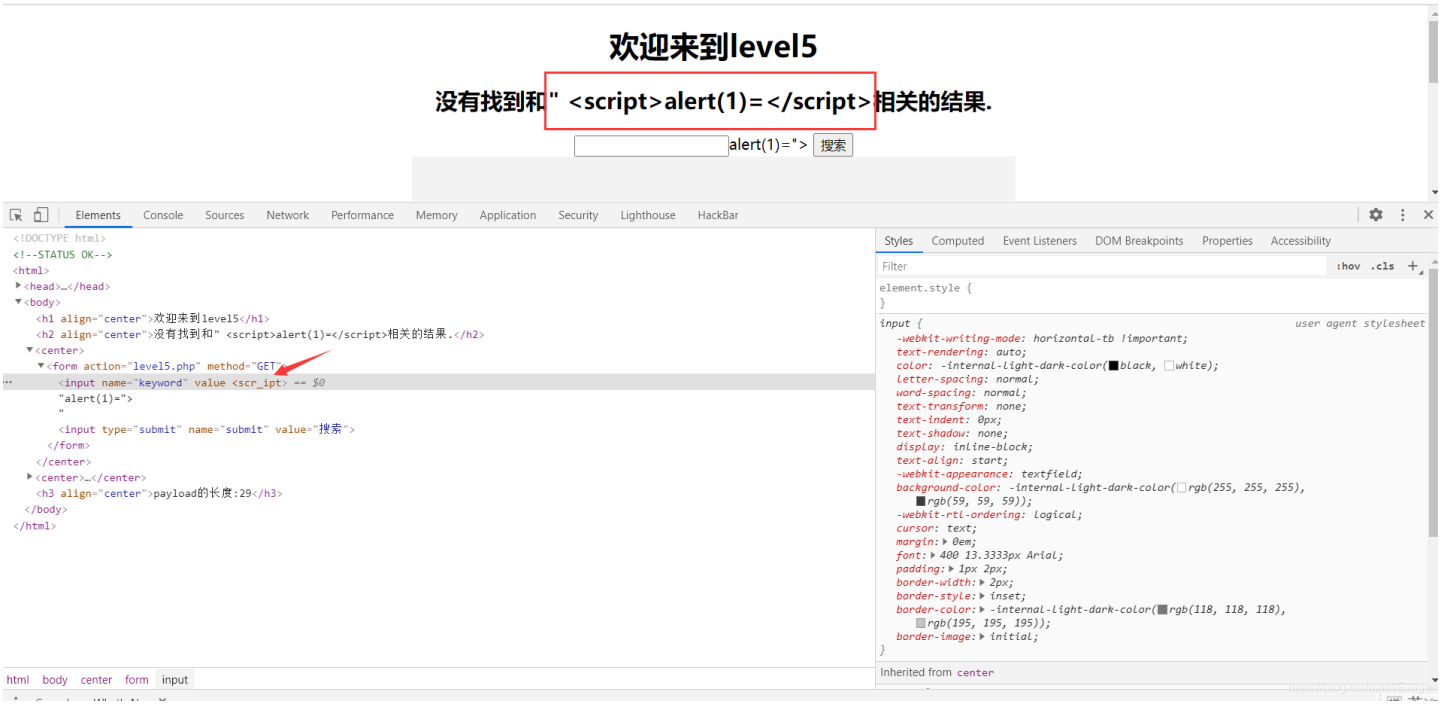
输入<>测试发现被过滤了



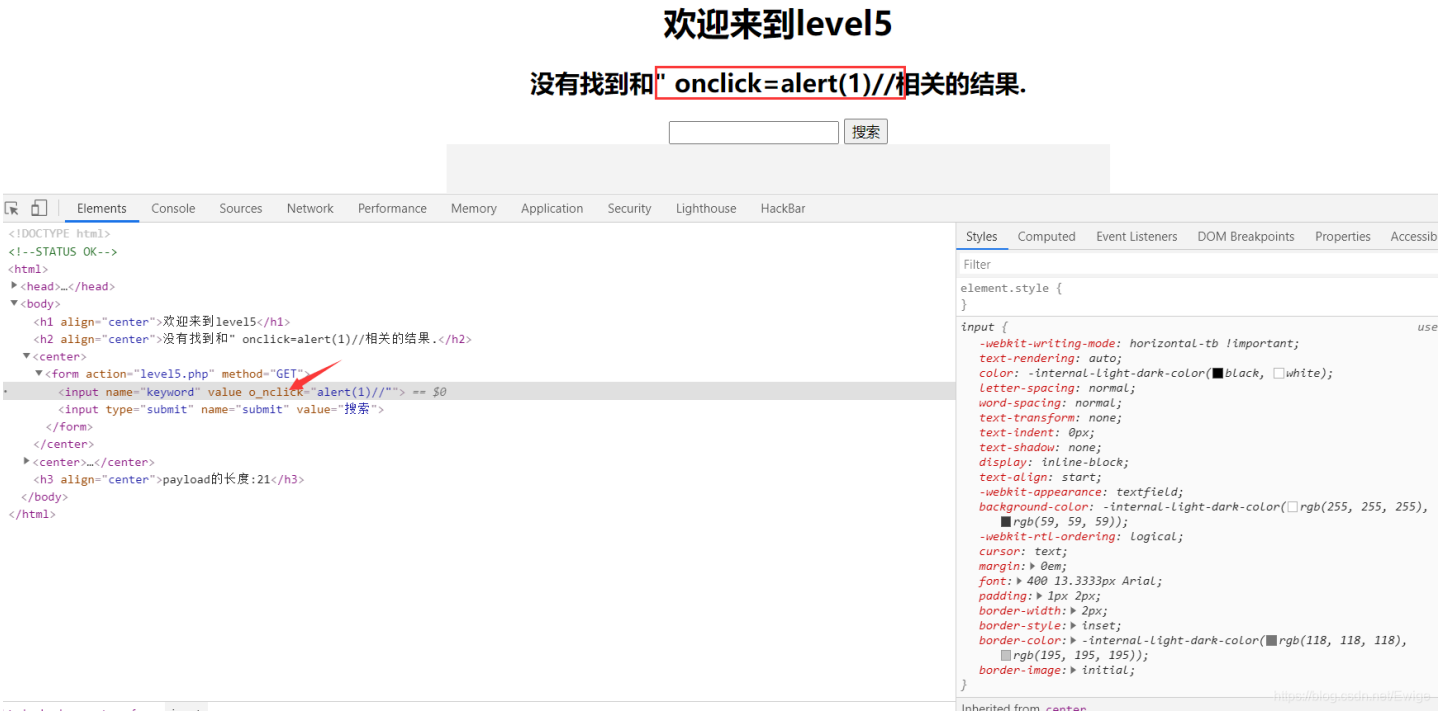
但是点击事件不需要大于小于号，这里需要闭合value
构造payload： ” onclick = alert(1) // ,然后点击输入框触发点击事件

level 5

尝试闭合注入



行不通script关键字被替换了



也被替换

尝试大小写饶过

" OnCliCk=alert(1)//

" 均被过滤

进行HTML编码：" onclick=alert(1) //

没有解析

查看下源码

```
</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($str)."相关的结果.</h2>".'<cen
<form action=level5.php method=GET>
<input name=keyword  value="'.$str3.'">
<input type=submit name=submit value=搜索 />
</form>
</center>';
?>
<center><img src=level5.png></center>
<?php
```


大于小

于号没有被过滤，

这里可以构造a标签的伪协议">

点击链接即可促发

level6

欢迎来到level6

没有找到和" onclick=alert(1) //相关的结果.

搜索

payload的长度:26

这道题不仅对上一道题的内容进行了过滤，还过滤了href关键字
但是没有对大小写进行过滤
构造payload：">

level7

对关键字进行测试
script ：直接全部白给
onclick： on也不知所踪□
大小写也不管用
复写试试：:>alert(1)

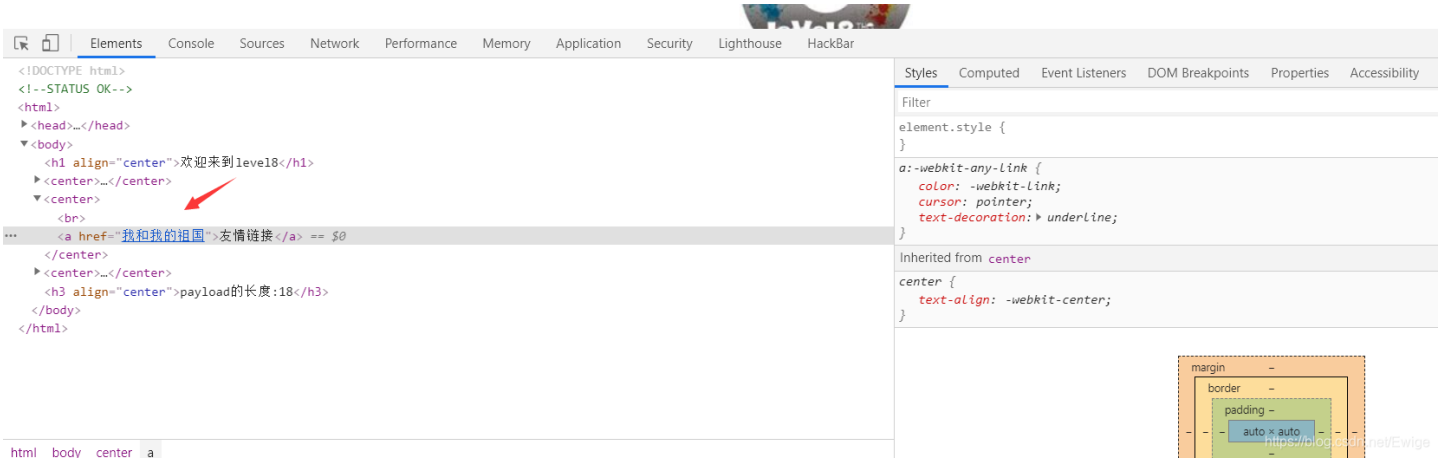
level8

发现添加后会传到此位置

欢迎来到level8

我和我的祖国 添加友情链接

友情链接



这里我们尝试构造a标签

javascript:alert(1)

行不通，

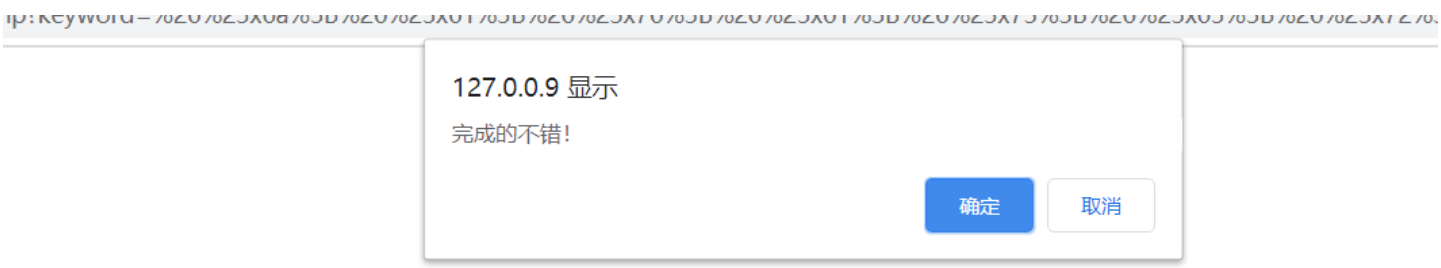
尝试大小写双写，也不行

尝试将javascript进行**HTML编码**

payload：

javascript :alert(1)

搞定



> 注：对函数名进行编码后，解析出来的只是html的普通字符；

如：对，l，a标签进行HTML编码

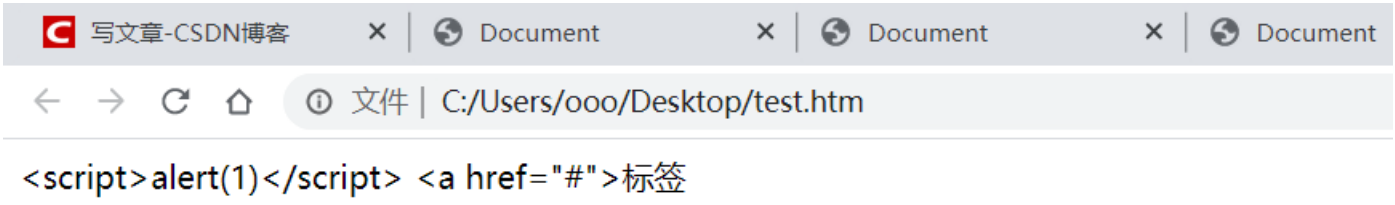
```
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    &#x3c;&#x73;&#x63;&#x72;&#x69;&#x70;&#x74;&#x3e;&#x61;&#x6c;&#x65;&#x72;&
    <&#x61; href="#">标签</&#x61;>

</body>

</html>
```


运行结果



因为这些函数方法，是底层构造好的，二，script，a…是这些方法，函数名，进行解析，时是不能对应哪些编码后的字符的

level9

添加链接：https://www.baidu.com
居然失败了

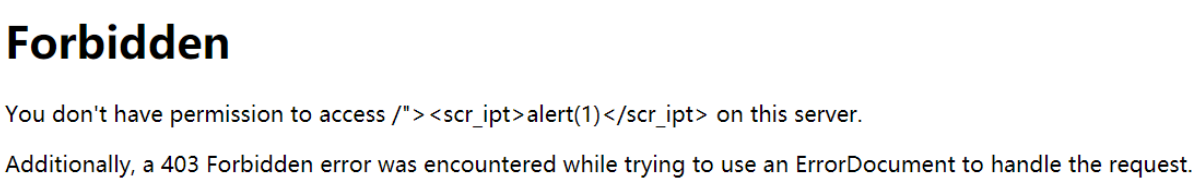
Not Found

The requested URL /æ, ¨çš„é "¾æŽ¥ä¸□å□ˆæ³•ï¼Ÿæœ‰æ²¡æœ‰ï¼□ was not found on this server.

Additionally, a 404 Not Found error was encountered while trying to use an ErrorDocument to handle the request.

换http://www.baidu.com试试
可以跳转

经过测试应该有判断输入是否含有http，否则就跳转



看下源码：

```php
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("script","scr_ipt",$str);
$str3=str_replace("on","o_n",$str2);
$str4=str_replace("src","sr_c",$str3);
$str5=str_replace("data","da_ta",$str4);
$str6=str_replace("href","hr_ef",$str5);
$str7=str_replace('"','&quot',$str6);
echo '<center>
```


这里对输入进行了一连串的过滤

```php
<?php
if(false===strpos($str7,'http://'))
{
```

然后通过strops函数查询输入的有没有http://这个字符串，
没有的话直接跳转的恶心人的网页
对javascript的t进行HTML编码；
构造payload：javascript:alert(1)//http://
搞定

level10

看来这是一个get方式的注入
输入

修改页面元素然后点击按钮触发弹窗

> 11，12，13题都可用此方法解答，就不赘述了

level14

这关需要请求远程站点资源，但好像出问题了，好吧，脚步不能停。

点击也没用，直接修改url

level15

观察源码：

ng-include 指令用于包含外部的 HTML 文件。包含的内容将作为指定元素的子节点。

ng-include 属性的值可以是一个表达式，返回一个文件名。默认情况下，包含的文件

需要包含在同一个域名下。

```
<script>
window.alert = function()
{
confirm("完成的不错！");
 window.location.href="level16.php?keyword=test";
}
</script>
<title>欢迎来到level15</title>
</head>
<h1 align=center>欢迎来到第15关，自己想个办法走出去吧！</h1>
<p align=center><img src=level15.png></p>
<?php
ini_set("display_errors", 0);
$str = $_GET["src"];
echo '<body><span class="ng-include:'.htmlspecialchars($str).'"></span></body>';
?>
```

使其包含leve 1 文件并构造如下payload ， 利用src异常报错 。

onerror 事件在加载外部文件（文档或图像）发生错误时触发。

payload:?name=test ’

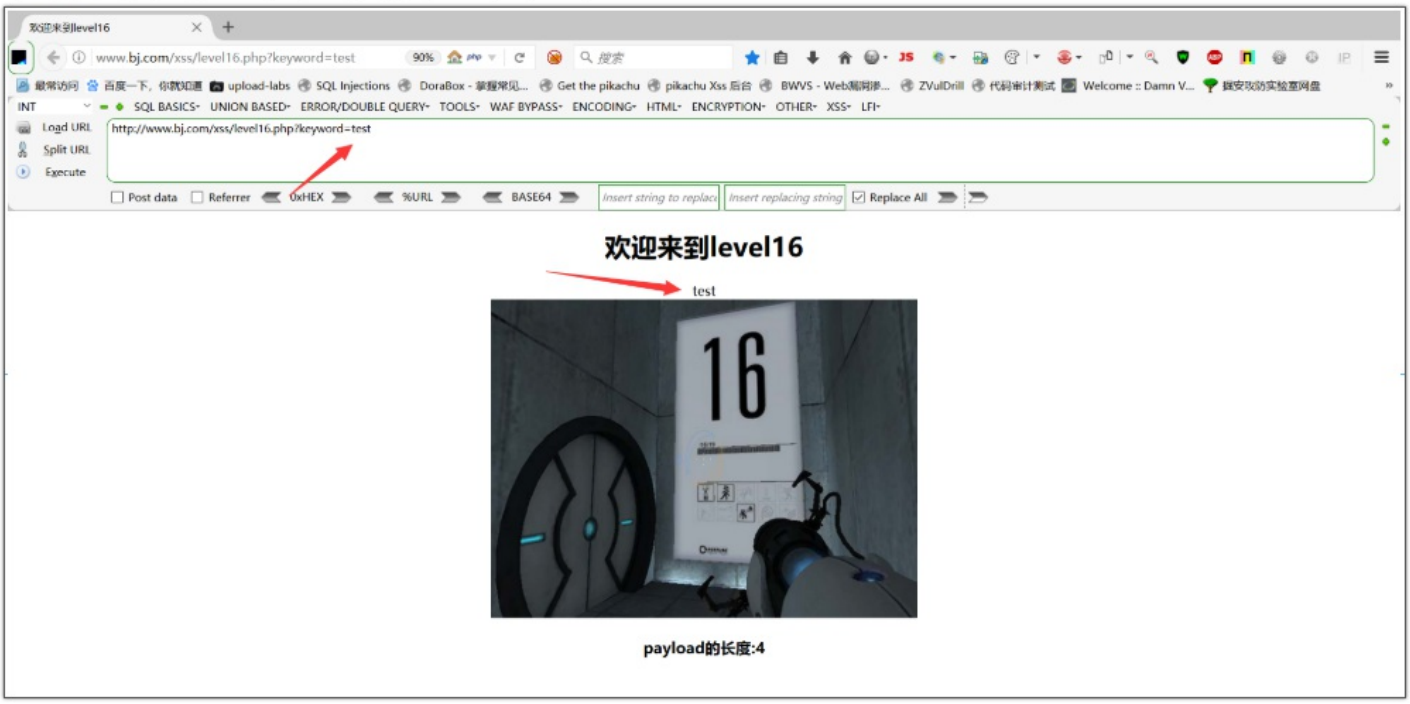
好吧又是转载的附上老哥链接：https://www.cnblogs.com/Zh1z3ven/p/12971361.html

level16

不知道为什么我的不显示了，下面，装载一位老哥的博客。

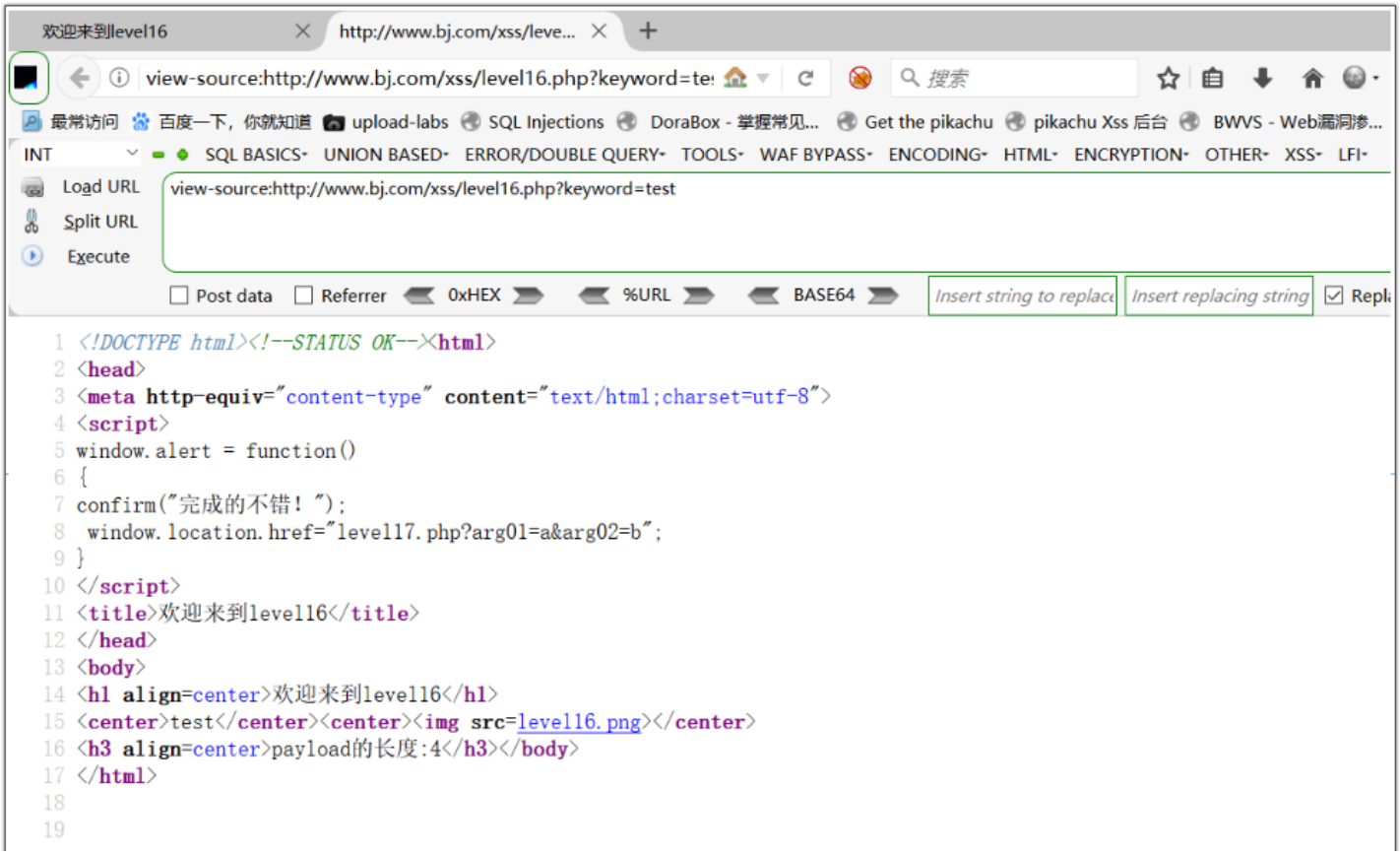
Level 16

跳转到第十六关，页面显示如下



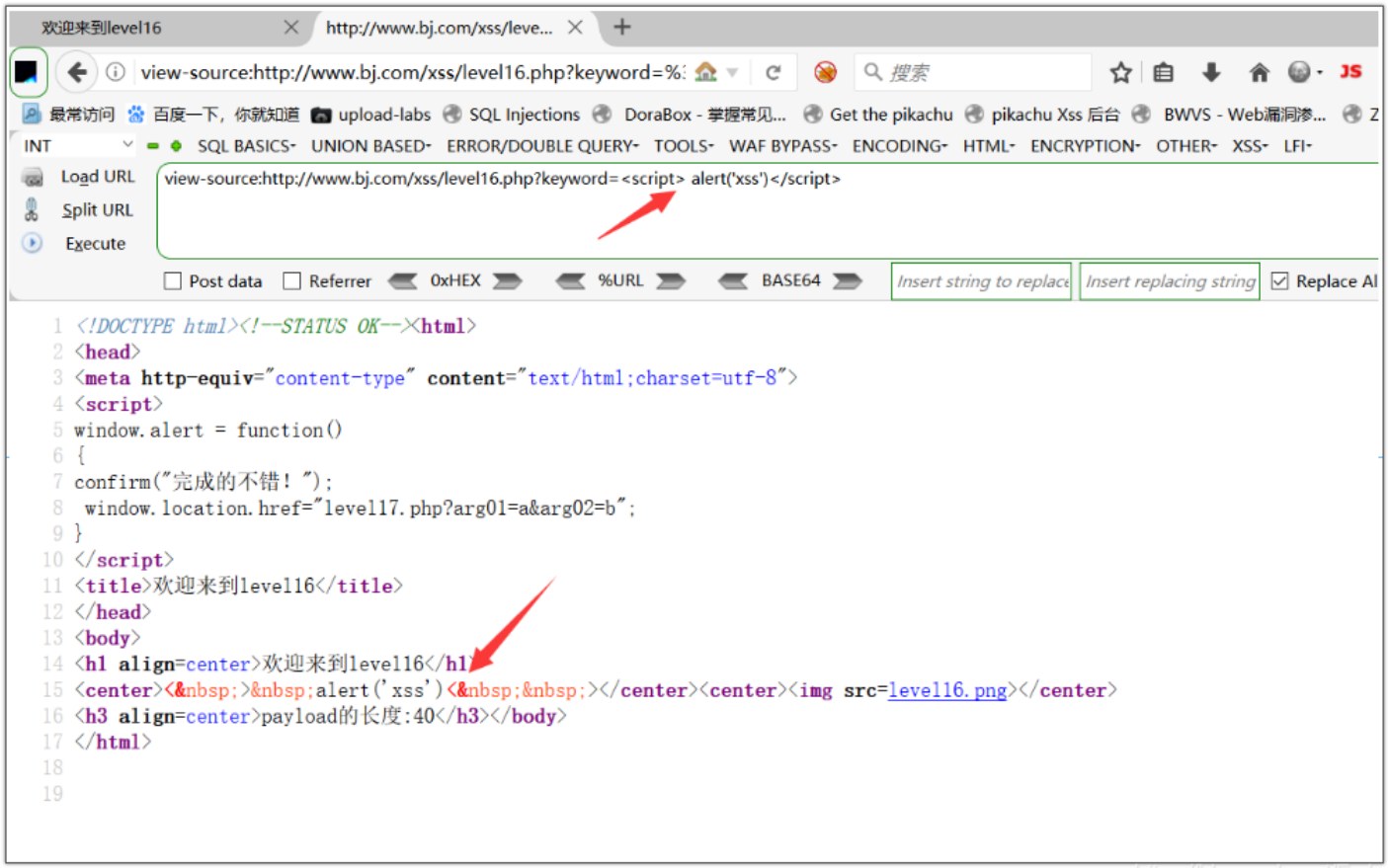
可以看到我们提交的参数值在页面中有一处显示位，接着看看网页源码

可以看到我们提交的参数值在页面中有一处显示位，接着看看网页源码




```
<!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="level17.php?arg01=a&arg02=b";
}
</script>
<title>欢迎来到level16</title>
</head>
<body>
<h1 align=center>欢迎来到level16</h1>
<center>test</center><center><img src=level16.png></center>
<h3 align=center>payload的长度:4</h3></body>
</html>
```

可以看到这里并没有什么特殊的地方，只是参数值被插入到了 标签中。那么先用最基本的弹窗代码测试一下吧。



可以看到关键字script以及/符号、空格都被编码成同样的空格字符实体了。这样也没办法去闭合前面的标签了。所以先看看源文件的代码

level16.php

```
<!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="level17.php?arg01=a&arg02=b";
}
</script>
<title>欢迎来到level16</title>
</head>
<body>
<h1 align=center>欢迎来到level16</h1>
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("script"," ",$str);
$str3=str_replace(" "," ",$str2);
$str4=str_replace("/"," ",$str3);
$str5=str_replace(" "," ",$str4);
echo "<center>".$str5."</center>";
?>
<center><img src=level16.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str5)."</h3>";
?>
</body>
</html>
```

1
2
3

可以看到在箭头1处是将参数值中的script替换成 。在箭头2处就是将参数值中的空格也替换成 。在箭头3处就是将参数值中的/符号替换成 。

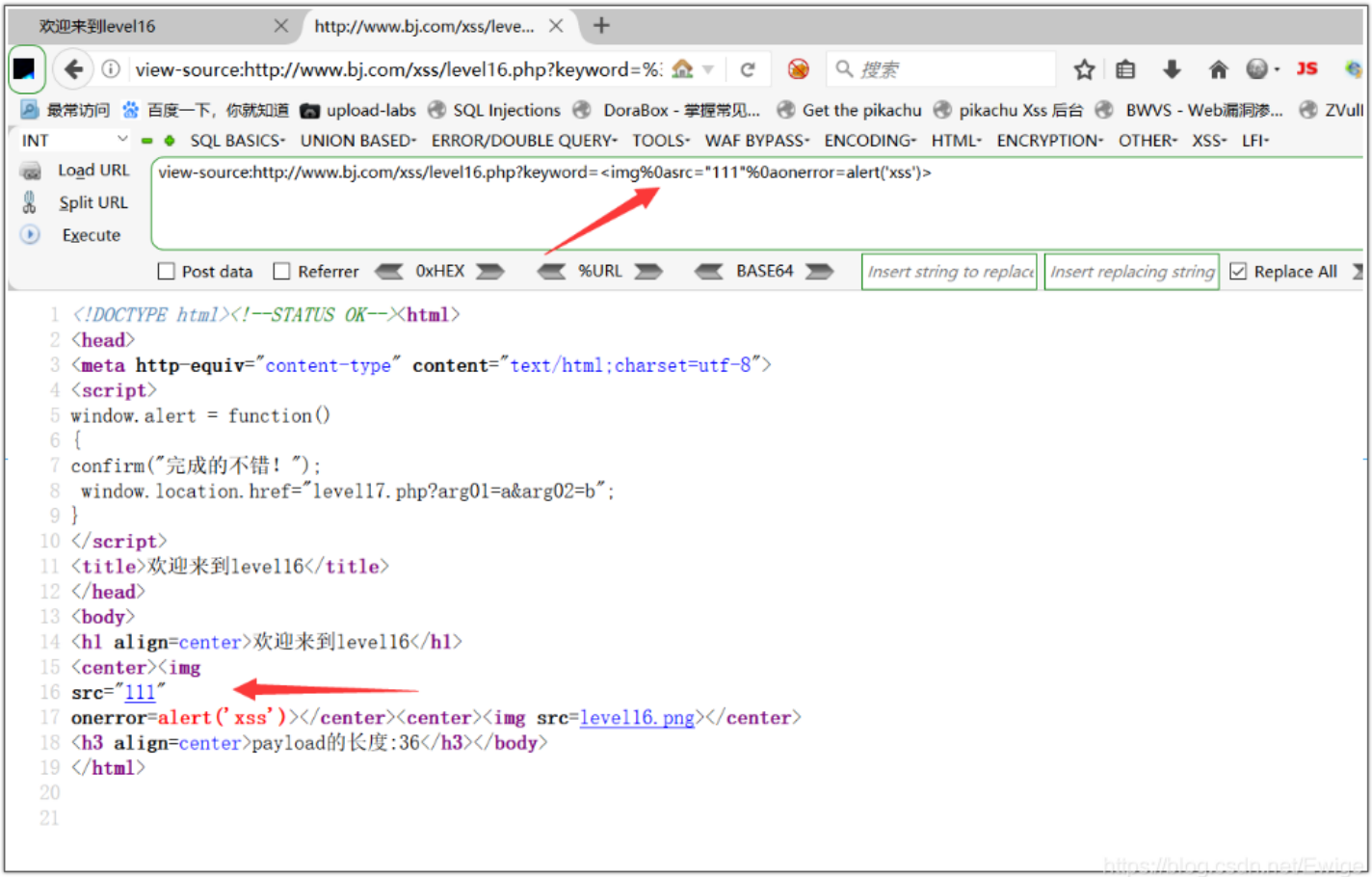
服务器端的操作跟我猜想的是一致的，但是这里要怎么才能绕过呐？

因为这里把空格都编码了，也就意味着我们无法通过空格来将字符分隔进行语义的区分，不过我们还可以用回车来将它们分开。而且这里将/符号也编码了，所以我们需要的是一个不需要闭合的标签，比如之前所用过的〈img〉。

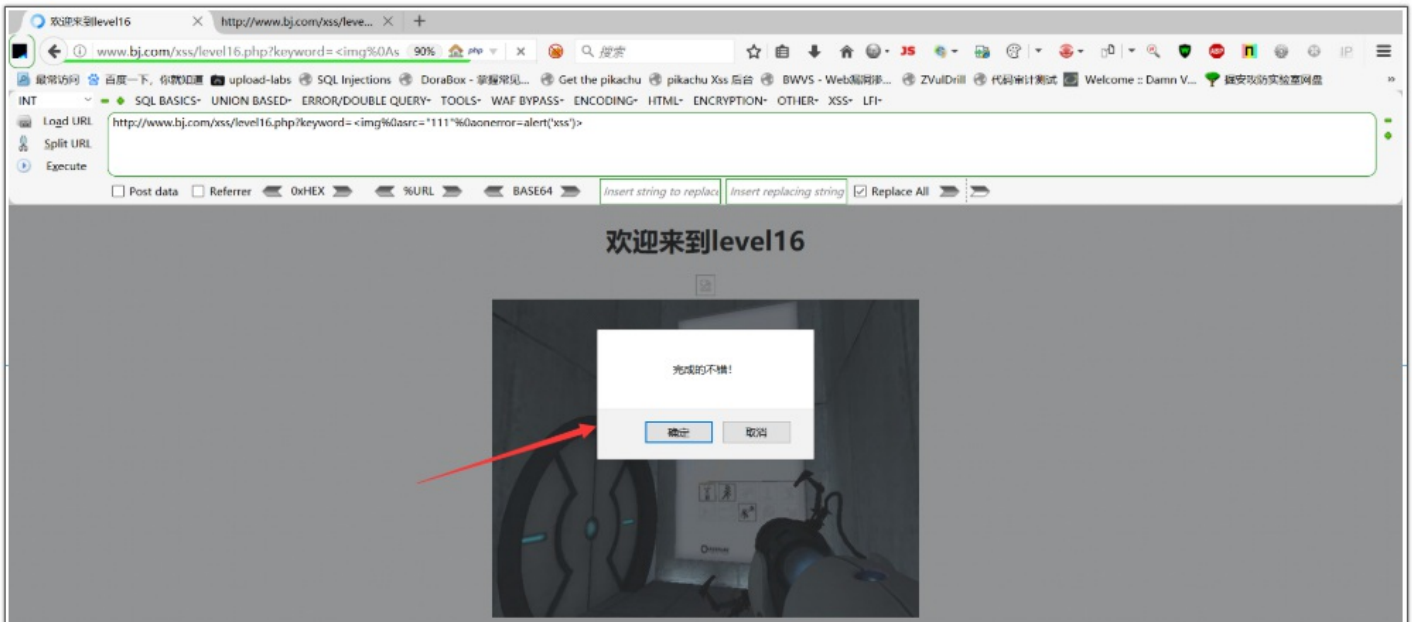
这里构造语句如下

```
<img
src=”111”
onerror=alert(‘xss’)
>
```

不过这里的回车怎么表示是一个问题，因为就像上面那样提交的话浏览器会将字符中间的多个间隔合并为一个空格。这里可以用回车的url编码格式%0a来表示。具体如下



可以看到网页源码中显示我们构造的代码是正常插入到 标签中了的，接下来就是看看是否能成功弹窗了



可以看到成功弹窗了，成功绕过第十六关。

摘自：https://www.zhaosimeng.cn/writeup/119.html

二级目录 17-20

靶场不知道怎么了，无法接收，传入的参数，下面给出一位老哥的挺详细的传送门