php scrscriptipt,HCTF 2016 writeup——web篇

转载

桔小咪 于 2021-03-19 06:57:03 发布 52 收藏
文章标签： php scrscriptipt

做了5个web题，分享一下思路。。

Level-1

1. 2099年的flag

打开链接，提示需要ios99才能得到flag，源码的注释中提示要POST，那么就修改User-Agent头为ios的头，版本改成99就好了。

Level-2

1. RESTFUL

打开题目链接链接，有这么一句

{"message":""Please me some more than <12450>!""}

很容易想到，需要使用PUT方法，并传递一个大于12450的money参数，结合题目名，联想到RESTFUL API，最终构造路径：

以PUT方式访问即可得到flag

hctf{Do_you_know_12450?}

2. giligili

打开题目链接，提示需要填入hctf{...}形式的字符串，应该就是flag了。

查看源码，发现一段混淆过的js代码。

**1) 预处理 **

先去掉一些不会用到的变量，函数，然后整理，格式化一下。整个代码大概可以分为三段：

一个存储了一些的函数对象“_”

一个保存了访问这些函数索引的数组“$”

一个函数check，用来处理主要逻辑。

2)计算answer

check函数同样也分为几个部分，我们注意到代码中对answer进行了分隔

o = answer.split("_");

所以围绕o[0]，o[1]，o[2]，o[3]看就好了

o[1]和o[2]

h = new MersenneTwister(parseInt(btoa(answer.substring(0, 4)), 32));

```
e = h.random() * 99;

e ^= h.mt[0];

l = new MersenneTwister(e);

l.random();l.random();l.random();

o = answer.split("_");

i = l.mt[~~(h.random() * 35725343) % 255];

s = ["0x" + i.toString(16), "0x" + e.toString(16).split("-")[1]];

e = -(this.eval(_[$[31]](o[1])) ^ s[0]);

if (-e != 0x697a) return false;

e ^= (this.eval(_[$[31]](o[2])) ^ s[1]);

if (-e != 0x623f21c1) return false;
```

注意到开始对answer取了前4位，而answer的形式又是hctf{...}，所以这里就是“hctf”，带进去根据后面的两个if判断，可以算出o[1]和o[2]

```
o[0]和o[3]

o[0] = o[0].substring(5);

o[3] = o[3].substring(0, o[3].length - 1);

if (o[0].length > 5) return false;

a = parseInt(_[$[23]]("1", Math.max(o[0].length, o[3].length)), 3) ^ eval(_[$[31]](o[0]));

a += _[$[31]](o[3].substring(o[3].length - 2)).split("x")[1];

if (parseInt(a.split("84")[1], $.length / 2) != 0x4439feb) return false;

d = parseInt(a, 16) == (Math.pow(2, 16) + -5 + "") + o[3].charCodeAt(o[3].length - 3).toString(16) + "53846" + (new Date().getFullYear() - 1 + "");

i = 0xffff;

n = (p = (f = _[$[23]](o[3].charAt(o[3].length - 4), 3)) == o[3].substring(1, 4));

g = 3;

t = _[$[23]](o[3].charAt(3), 3) == o[3].substring(5, 8) && o[3].charCodeAt(1) * o[0].charCodeAt(0) == 0x2ef3;

h = ((31249 * g) & i).toString(16);

i = _[$[31]](o[3].split(f).join("").substring(0, 2)).split("x")[1];

s = i == h;

return (p & t & s & d) === 1 || (p & t & s & d) === true;
```

注意到前面对o[0]和o[3]进行了截取，这主要是为了去掉前面的“hctf{”和最后的“}”

然后从后往前看，要求p，t，s，d都为真，根据这个一点一点推，中间会遇到需要暴力猜解的情况，可以算出o[0]和o[3]，最后拼起来就是flag了。不过发现这个代码逻辑有一点问题，不过最后猜了一下flag应该是完整的一句话，在这个代码中是不能通过的，但是交上去通过了：

hctf{wh3r3_iz_y0ur_neee3eeed??}

3. 兵者多诡

// 这个跟之前的swpu的ctf很像，但是比较坑的是基本没有提示。。。所以没有做过类似题目的很难有这个脑洞。

打开题目链接，是一个图片上传页面，上传后会给出上传后存储的地址。文件名会被修改为“随机字符串.png”的形式

经测试发现后台并不检测文件名和文件类型，只检测Content-Type，但是由于对文件重命名了，所以无论如何上传后都只能是png。

但是发现上传的请求url是这样的形式：

/home.php?fp=upload

所以猜测后台逻辑可能是include了一个upload文件进行处理，那么试试将upload改成“../../../etc/passwd”：

/home.php?fp=../../../etc/passwd

页面返回：

No No No!

说明猜测对了，这样我们可以试试读取upload.php的源码，是可以的，那么和swpu类似，我们可以使用php的phar协议读取压缩文件中的文件，从而实现命令执行。

我们首先做一个一句话木马shell.php，然后压缩成zip，然后将这个zip后缀改成“.png”后上传。

假设上传后的图片为路径为“uploads/xxx.jpg”，那么我们访问这个url就可以将shell代码包含到当前页面：

后台会自动给fp后面添上“.php”，然后包含，这样就可以命令执行了，读取到/var/www目录下有一个“Th1s_1s_F1a9.php html ”，cat一下，然后查看源码，找到flag：

hctf{Th1s_1s_e4sY_1s_n0T_1t?}

Level-3

1. guestbook

打开链接，有两个输入框message和code，有提示

substr(md5($code),0,4) =='cec6'

这个后面的“cec6”没刷新一次就会变

很明显要让code的md5前四位等于cec6，写了一个脚本算了一下，用1000000以内的数字去试就好了，正确会显示提交成功，然后会提示：

AWAITING FOR THE ADMIN'S APPROVAL

并把提交的message的值显示出来，既然说等待管理员的同意嘛，然后又显示我们提交的东西，那么猜测是一个XSS。经过测试，有一些过滤，下面给出绕过方法：

1.会将message中的“img”，“svg”，“script”，“on”删掉

双写绕过就好了(scrscriptipt)

2.“/”换成“_”

用escape之类的编码一下就好了

3.过滤了单引号

用双引号，或者是反引号

然后响应头中还有CSP策略，看了一下限制挺多了，就不想了，直接用dns预加载绕过，写个脚本插入标签：

这样页面就会发一个dns查询请求，无视所有的CSP限制，这样就可以用dnslog打到管理员cookie了：

admin=hctf2o16com30nag0gog0

以为这就是flag，可惜不对，所以继续打管理页面名字，然后带着这个cookie访问这个页面就可以得到flag了。

Level-3还有个web看了下没思路，就去睡觉了，总结还是太菜了。。