

# 攻防世界crypto高手题之streamgame2

原创

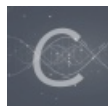
沐一·林 于 2021-09-21 12:37:51 发布 111 收藏 1

分类专栏: [CTF 密码学](#) 文章标签: [ctf](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/xiao\\_\\_1bai/article/details/120400350](https://blog.csdn.net/xiao__1bai/article/details/120400350)

版权



[CTF](#) 同时被 2 个专栏收录

167 篇文章 6 订阅

订阅专栏



[密码学](#)

51 篇文章 1 订阅

订阅专栏

## 攻防世界crypto高手题之streamgame2

继续开启全栈梦想之逆向之旅~

这题是攻防世界crypto高手题的streamgame2

### streamgame2

最佳Writeup由 [系统战队](#) · admin 提供

难度系数: ★★★★3.0

题目来源: [QWB-2018](#)

题目描述: 暂无

题目场景: 暂无

题目附件: [附件1](#)

WP [建议](#)

CSDN @沐一一沐, 一沐沐一

下载附件, 还是典型的 [LFSR](#) 类型:

```

from flag import flag
assert flag.startswith("flag{")
assert flag.endswith("}")
assert len(flag)==27

def lfsr(R,mask):
    output = (R << 1) & 0xffffffff
    i=(R&mask)&0xffffffff
    lastbit=0
    while i!=0:
        lastbit^=(i&1)
        i=i>>1
    output^=lastbit
    return (output,lastbit)

R=int(flag[5:-1],2)
mask=0x100002

f=open("key","ab")
for i in range(12):
    tmp=0
    for j in range(8):
        (R,out)=lfsr(R,mask)
        tmp=(tmp << 1)^out
    f.write(chr(tmp))
f.close()

```

(这里积累第一个经验)

原理同 `streamgame1`，可参照博客 [https://blog.csdn.net/xiao\\_\\_1bai/article/details/120399710](https://blog.csdn.net/xiao__1bai/article/details/120399710) 这里唯一的不同就是 `mask` 也是要 `截取` 的，因为 `mask` 要看成 `常数`，要求的是 `21位` 初始值，所以 `24位` 的 `mask` 要截取成 `21位`。

```

1 from flag import flag
2 assert flag.startswith("flag{")
3 assert flag.endswith("}")
4 assert len(flag)==27
5
6 def lfsr(R,mask):
7     output = (R << 1) & 0xffffffff
8     i=(R&mask)&0xffffffff
9     lastbit=0
10    while i!=0:
11        lastbit^=(i&1)
12        i=i>>1
13    output^=lastbit
14    return (output,lastbit)
15
16
17
18 R=int(flag[5:-1],2)
19 mask=0x100002
20
21 f=open("key","ab")
22 for i in range(12):
23     tmp=0
24     for j in range(8):
25         (R,out)=lfsr(R,mask)
26         tmp=(tmp << 1)^out
27     f.write(chr(tmp))
28 f.close()

```

这里应该是21位二进制数

又是同样的补位逻辑

这里6位十六进制数共24位，应该取前21位即可。  
前21位就是100000000000000000010，那么反馈函数就是 $lastbit = R2 \oplus R21$

CSDN @沫一一沫，一沫沫一

直接上脚本：

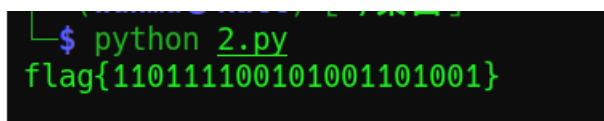
```

f = open('3key','rb') #以二进制格式打开文件
content = f.read() #读取的是\xhh类型的十六进制
key=[ '{:0>8}'.format(str(bin(i)).replace('0b',''))) for i in content] #从base64编码汲取的经验，二进制8位对齐。
key1=''.join(key)[:21] #Lastbit全部值，就是反馈函数生成的值，32位的key1
key2=key1
flag=[]
for i in range(21):
    output='?'+key1[:20] #?0100000111110111101111011111000，因为后面有key1=str(Lastbit)+key1[:31]，key1不断填补，output不断取前31位，所以这里output每次把?定在第i位上，注意output和key1是独立的分开的。
    flag.append(str(int(key2[-1-i])^int(output[-2])))
#这里之所以取负数是因为我们截取是从左往右取，而在计算机中是小端顺序，应该从右往左取才对，这里的key2[-1-i]就是小端左到右的第i位，也就是前面分析的反馈函数生成值的第i位。flag值的原第i位，现在在第21位，可以通过⊕的可逆性来求，就是R21=Lastbit ⊕ R2

key1=str(flag[i])+key1[:20]#不断填补key1，让key1向右推进，
print("flag{"+bin(int(''.join(flag[::-1]),2)).replace('0b','')+}")#这里是经过一系列操作，首先把flag变成小端顺序的[::-1]，然后就是转十六进制。

```

结果：



```

$ python 2.py
flag{110111100101001101001}

```

总结：

- 1: (这里积累第一个经验)  
原理同 `streamgame1`，可参照博客 [https://blog.csdn.net/xiao\\_\\_1bai/article/details/120399710](https://blog.csdn.net/xiao__1bai/article/details/120399710) 这里唯一的不同就是 `mask` 也是要 **截取** 的，因为mask要看成 **常数**，要求的是 **21位** 初始值，所以 **24位** 的mask要截取成 **21位**。

解毕！敬礼！