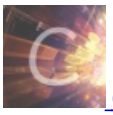
2020数字中国创新大赛虎符网络安全赛RE——game WP

原创

Cheney辰星 于 2020-04-21 09:23:46 发布 561 收藏 1

分类专栏： CTFer's WP RE CTF

本文链接：https://blog.csdn.net/SC_king/article/details/105650520


CTFer's WP 同时被 3 个专栏收录

7 篇文章 0 订阅

订阅专栏


RE

12 篇文章 0 订阅

订阅专栏


CTF

6 篇文章 0 订阅

订阅专栏

阅读字节码后，首先定义了三个列表，arr0,arr1,arr2。

主函数逻辑大致如下(伪代码形式给出)：

获取输入，然后依次调用4个check函数对输入进行check，如果其中任何一个不通过，返回失败。

```
flag = input()

if check0(flag) == FALSE:
    goto NO;
if check1(flag) == FALSE:
    goto NO;
if check2(flag) == FALSE:
    goto NO;
if check3(flag) == FALSE:
    goto NO;

print "OK"
return

NO:
print "NO"
return
```

分别查阅4个check函数。4个函数翻译过后，整理出其关键运算处(用伪代码形式)，推测出其作用。

check0:字符ASCII值必须在32~127即可打印字符

check1:长度校验，可以爆破得flag长度

check2:检查输入格式是否为flag{xxxx},并且这里可以爆破得到第一个字符5

check3:核心检验算法

```
flag = 'XXXXX' #flag{HFCTF}
s = flag
#check0:#字符ASCII值必须在32~127即可打印字符
    all(genexpr(s)) #返回false or true
#check1:    //长度校验,可以逆向出输入的长度是39，我一直想当然40，结果解出来flag老是有问题。。。
   ((s.len * s.len) % 777) ^ 233 == 513
#check2:    检查输入格式是否为flag{xxxx},同时可以获得中括号内第一个字符串
    ((((ord(s[0]) * 128 + ord(s[1])) *128 + ord(s[2])) *128 + ord(s[3])) *128 + ord(s[4])) *128 + ord(s[5])  == 3533889469877L
    && ord(s[-1]) == 125

#check3:核心检验算法


    arr = map(ord, s)
    a = arr[6:30:3]
    for i in range(len(a)):
        if((a[i] * 17684 + 372511) % 257 != arr0[i] == False:
            return FALSE;

    b = arr[-2:33:-1] * 5  //flag{xx}中括号内容中最后4位字符
    #终于看懂map这行了，zip返回一个元素是一对元组的列表，然后对列表中的每对元组进行异或合并成一个新的列表返回
    #lambda x : x[0] ^ [1]
    c = map(lambda,zip(b, arr[7:27])
    if(c != arr1)
        return FALSE;


    p = 0
    for i in range(28,34):
        if (arr[i] + 107) / 16 + 77 !=arr2[p] == True:
            goro False
        if (arr[i] + 117) % 16 + 99 != arr2[p+1] == False:
            p = p + 2
```

check3函数分别将输入分出三个部分进行不同运算校验,需要针对三个部分进行逆向，最后对逆向出的三部分字符进行合理组合，获得最终flag.

```
arr0 = [249, 91, 149, 113, 16, 91, 53, 41]
arr1 = [43, 1, 6, 69, 20, 62, 6, 44, 24, 113, 6, 35, 0, 3, 6, 44, 20, 22, 127, 60]
arr2 = [90, 100, 87, 109, 86, 108, 86, 105, 90, 104, 88, 102]

# 获取flag长度
length = 0
for i in range(100):
    if ((i * i) % 777) ^ 233 == 513:
        length = i
# print(length)

flag = list(range(length))  # len = 39
flag[0] = 'f'
flag[1] = 'l'
flag[2] = 'a'
flag[3] = 'g'
flag[4] = '{'
flag[-1] = '}'

#获取第6个字符
for j in range(256):
    if ((((ord(flag[0]) * 128 + ord(flag[1])) *128 + ord(flag[2])) *128 + ord(flag[3])) *128 + ord(flag[4])) *12
8 + j  == 3533889469877:
        flag[5] = chr(j)
        print(chr(j))
#print(flag)

# 第一部分
```

```python
# 第一部分
a = ''
for i in range(8):
    for j in range(256):
        if ((j * 17684 + 372511) % 257 == arr0[i]):
            a += chr(j)
            flag[6 + i * 3] = chr(j)
print(a)
# print(flag)

# 第二部分
b = list(range(4))
for i in range(4):
    for j in range(256):
        if j ^ ord(flag[12 + i * 3]) == arr1[5 + i * 3]:
            b[(1 + i * 3) % 4] = j
print(b)
flag[-2] = chr(b[0])
flag[-3] = chr(b[1])
flag[-4] = chr(b[2])
flag[-5] = chr(b[3])
#print(flag)

b = b * 5
c = ""
for i in range(len(b)):
    for j in range(256):
        if b[i] ^ j == arr1[i]:
            flag[7 + i] = chr(j)
            c += chr(j)
print(c)
#print(flag)

#第三部分   我用Python3跑不出，，最后用同样的判断条件C++跑出来了，不知道为什么。
flag[28] = '1'
flag[29] = '5'
flag[30] = '4'
flag[31] = '1'
flag[32] = 'p'
flag[33] = 'N'

for ch in flag:
    print(ch, end='')
```

```cpp
#include <iostream>
#include <vector>
#include <string>
#include <algorithm>

using namespace std;

int main()
{
 vector<int>arr2{ 90, 100, 87, 109, 86, 108, 86, 105, 90, 104, 88, 102 };


 string d;
 int p = 0;
 for (int i = 0; i < 6; ++i)
 {
  for (int j = 0; j < 128; ++j)
  {
   if ((((j + 107) / 16) + 77) == arr2[p] && ((j + 117) % 16) + 99 == arr2[p + 1])
    d.push_back((char)j);
  }
  p += 2;
  }
 cout << d << endl;

 return EXIT_SUCCESS;
}
```

> flag{5LZG50ex5Yi75VqE5YePLlK1541pNu3Fq}

> 看了官方WP发现，我把最后四个字符弄反了…难怪一直提交不对，无语。还有就是第三部分的flag字符，我用Python3一直跑不出，我一怀疑是不是我条件看错了，反复校验发现没问题，于是用C++跑了下，出来了，无语。但官方用Python2能跑出，不是很明白。