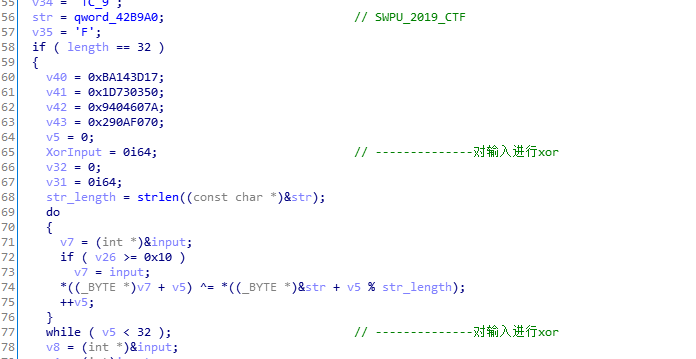
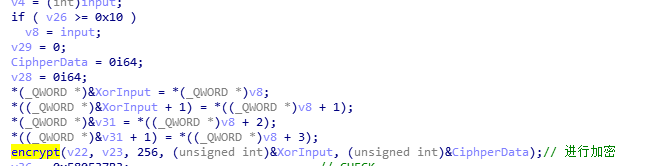
算法直来直去的很容易分析，分为三部分。

# 0x0 对输入进行xor运算



根据” SWPU\_2019\_CTF”来异或的。

# 0x1 加密



分析加密函数，可以发现前面一堆都是为了得到table的。

我们进行动态调试即可很快dump下了table。

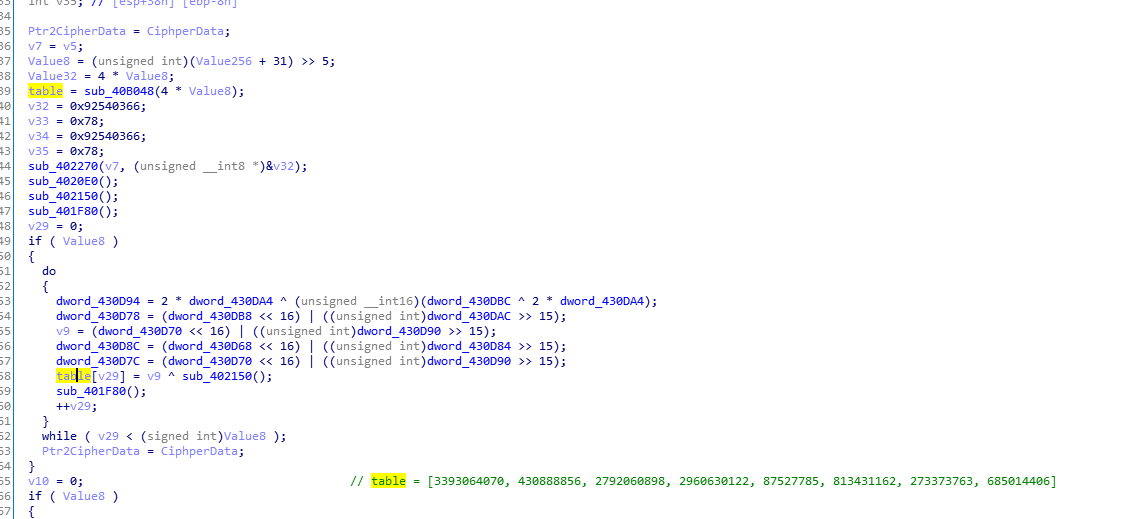
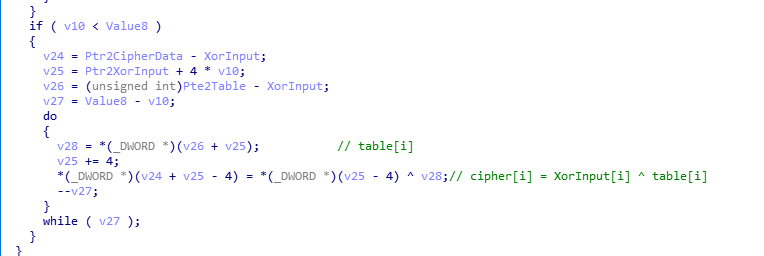


table = [3393064070, 430888856, 2792060898, 2960630122, 87527785, 813431162, 273373763, 685014406]

得到table后执行运算时最下面的。



其中需要理解的时v24+v25-4其实就是cipher[i]，v26+v25就是table[i]

循环里面进行的操作就是cipher[i] = XorInput[i] ^ table[i]。

# 0x2 脚本

import struct  
XorInput = ""  
flag = ""  
str = "SWPU\_2019\_CTF"  
cmp\_data = [0xF80F37B3, 0x5DAEBCBC, 0x864D5ABA, 0xD3629744, 0x1624BA4F, 0x1A729F0B, 0x266D6865, 0x67C86BBA]  
table = [3393064070, 430888856, 2792060898, 2960630122, 87527785, 813431162, 273373763, 685014406]  
  
for i in range(8):  
 XorInput += struct.pack("<L",cmp\_data[i] ^ table[i])  
  
for i in range(len(XorInput)):  
 flag += chr(ord(XorInput[i]) ^ ord(str[i % len(str)]))  
print flag

flag：swpuctf{y0u\_@re\_s0\_coo1}