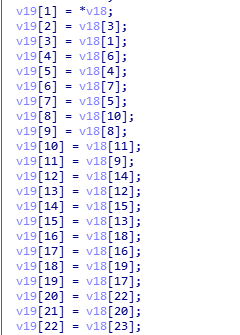
拖进ida分析，可以很快找到xxtea加密的函数。

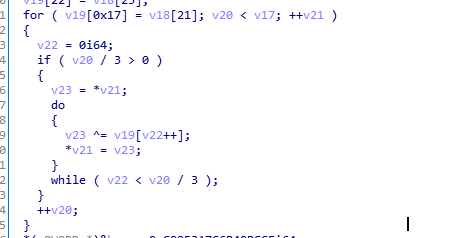


动态调试的话可以发现key就是输入的前四个字节。

然后继续往下看可以发现是对加密后的数据进行了换位操作。

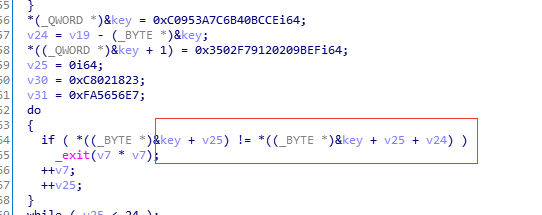


然后下面是对换位后的数据进行运算，这部分运算我们用z3来解。



最后是和比较数据进行对比，比较数据为

cmp\_data = [206, 188, 64, 107, 124, 58, 149, 192, 239, 155, 32, 32, 145, 247, 2, 53, 35, 24, 2, 200, 231, 86, 86, 250]



写出z3脚本得到换位后的数据

from z3 import \*  
data = [206,188,64,165,178,244,231,178,157,169,18,18,200,174,91,16,61,29,215,248,220,220,112]  
print len(data)  
  
  
cmp\_data = [206, 188, 64, 107, 124, 58, 149, 192, 239, 155, 32, 32, 145, 247, 2, 53, 35, 24, 2, 200, 231, 86, 86, 250]  
s = Solver()  
flag = [BitVec(('x%d' % i),8) for i in range(24) ]  
  
for i in range(24):  
 a = flag[i]  
 for j in range(i / 3):  
 a ^= flag[j]  
 flag[i] = a  
for i in range(24):  
 s.add(flag[i] == cmp\_data[i])  
  
if s.check() == sat:  
 model = s.model()  
 print model  
  
else:  
 print "unsat"

data =[206,188,64,165,178,244,231,178,157,169,18,18,200,174,91,16,6,61,29,215,248,220,220,112]

写脚本还原换位前的数据。

arr1 = [2,0,3,1,6,4,7,5,10,8,11,9,14,12,15,13,18,16,19,17,22,20,23,21]  
data = [206,188,64,165,178,244,231,178,157,169,18,18,200,174,91,16,6,61,29,215,248,220,220,112]  
data1 = [0 for i in range(24)]  
for i in range(24):  
 data1[arr1[i]] = data[i]  
print data1

data1 =

[188, 165, 206, 64, 244, 178, 178, 231, 169, 18, 157, 18, 174, 16, 200, 91, 61, 215, 6, 29, 220, 112, 248, 220]

然后进行xxtea解密，密钥就是flag格式前四位“flag”。

# -\*- coding: utf-8 -\*-  
import struct  
  
\_DELTA = 0x9E3779B9  
  
  
def \_long2str(v, w):  
 n = (len(v) - 1) << 2  
 if w:  
 m = v[-1]  
 if (m < n - 3) or (m > n): return ''  
 n = m  
 s = struct.pack('<%iL' % len(v), \*v)  
 return s[0:n] if w else s  
  
  
def \_str2long(s, w):  
 n = len(s)  
 m = (4 - (n & 3) & 3) + n  
 s = s.ljust(m, "\0")  
 v = list(struct.unpack('<%iL' % (m >> 2), s))  
 if w: v.append(n)  
 return v  
  
  
  
def decrypt(str, key):  
 if str == '': return str  
 v = \_str2long(str, False)  
 k = \_str2long(key.ljust(16, "\0"), False)  
 n = len(v) - 1  
 z = v[n]  
 y = v[0]  
 q = 6 + 52 // (n + 1)  
 sum = (q \* \_DELTA) & 0xffffffff  
 while (sum != 0):  
 e = sum >> 2 & 3  
 for p in xrange(n, 0, -1):  
 z = v[p - 1]  
 v[p] = (v[p] - ((z >> 5 ^ y << 2) + (y >> 3 ^ z << 4) ^ (sum ^ y) + (k[p & 3 ^ e] ^ z))) & 0xffffffff  
 y = v[p]  
 z = v[n]  
 v[0] = (v[0] - ((z >> 5 ^ y << 2) + (y >> 3 ^ z << 4) ^ (sum ^ y) + (k[0 & 3 ^ e] ^ z))) & 0xffffffff  
 y = v[0]  
 sum = (sum - \_DELTA) & 0xffffffff  
 return \_long2str(v, True)  
  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 key = "flag"  
 data1 = [188, 165, 206, 64, 244, 178, 178, 231, 169, 18, 157, 18, 174, 16, 200, 91, 61, 215, 6, 29, 220, 112, 248, 220]  
 s = "".join(map(chr, data1))  
 s = decrypt(s, key)  
 print repr(s)

Flag： **flag{CXX\_and\_++tea}**