护网杯 CRYPTO fez

原创

alex19914 于 2018-10-16 10:23:53 发布 431 收藏 1
分类专栏： CTF 文章标签： CTF

fez

原题包含两个文件：fez.py和fez.log。

fez文件内容如下：

```
import os
def xor(a,b):
    assert len(a)==len(b)
    c=""
    for i in range(len(a)):
        c+=chr(ord(a[i])^ord(b[i]))
    return c
def f(x,k):
    return xor(xor(x,k),7)
def round(M,K):
    L=M[0:27]
    R=M[27:54]
    new_l=R
    new_r=xor(xor(R,L),K)
    return new_l+new_r
def fez(m,K):
    for i in K:
        m=round(m,i)
    return m

K=[]
for i in range(7):
    K.append(os.urandom(27))
m=open("flag","rb").read()
assert len(m)<54
m+=os.urandom(54-len(m))

test=os.urandom(54)
print test.encode("hex")
print fez(test,K).encode("hex")
print fez(m,K).encode("hex")
```

fez.log中，则是保存了 打印的三行记录，分别对应了test.encode("hex")、fez(test,k).encode("hex")、fez(m,k).encode("hex")。

2315d80c2dd73098953686be6c82aa63c1d362eb0095e4621cce28bec4c921ce016afc7f39fd93b14b6c28ce69c
308e590a180473ab4d23a0c67b65fe2bf2d0a9f1b255e4e2610b0c90e8e210c8ed4f2b9a3b09c1886a781f94fee4
e822e918e578a7af4f0859a99aab5d7563644beb4207a73d5fc4560d3deb696320cec479431a4f724310499baf5

那么加密的算法应该就是fez中进行加密的，看了py文件，发现加密的本质是异或运算。

异或的运算的性质有：

1.交换律

2.结合律

3.对于任何数x，x^x=0，x^0=x

4.a^b=c,a^c=b

分析加密的过程，原字符串分解为L,R，经过7次循环之后，变成了x^R+y^L^R。x与y的值与加密用的K有关，K是随机生成的。这里K的值不确定，但是通过异或运算的性质，fez(test,K)与fez(m,K)异或把K消去。

fez(test,K)=x^test_R+y^test_L^test_R，fez(m,K)=x^m_R+y^m_L^m_R

即x^test_R^x^m_R=test_R^m_R，这个作为test与m的异或结果的右半部分R，记左半部分为L。

再根据y^test_L^test_R^y^m_L^m_R=L^R，再与R异或得到L。

此时即可得到test与m的异或：L+R。

最后与test异或即得到包含flag信息的m。m=test^m^test

代码如下：

```
def xor(a,b):
    assert len(a)==len(b)
    c=""
    for i in range(len(a)):
        c+=chr(ord(a[i])^ord(b[i]))
    return c

test =
'2315d80c2dd73098953686be6c82aa63c1d362eb0095e4621cce28bec4c921ce016afc7f39fd93b14b6c28ce69c7096b91fd2db0862
d'.decode('hex')
test_K =
'308e590a180473ab4d23a0c67b65fe2bf2d0a9f1b255e4e2610b0c90e8e210c8ed4f2b9a3b09c1886a781f94fee4f77488c0b30f239
5'.decode('hex')
m_K =
'e822e918e578a7af4f0859a99aab5d7563644beb4207a73d5fc4560d3deb696320cec479431a4f724310499baf5230db7e56764915d
0'.decode('hex')

L0 = test[:27]
R0 = test[27:]
test_L = test_K[:27]
test_R = test_K[27:]
m_L = m_K[:27]
m_R = m_K[27:]
R = xor(test_L, m_L)
RL = xor(test_R, m_R)
L = xor(R, RL)
test_m = L+R
m = xor(test, test_m)
print m
```