攻防世界 Crypto高手进阶区 6分题 xor_game

原创

思源湖的鱼 于 2021-01-20 17:57:42 发布 477 收藏
分类专栏： ctf 文章标签： ctf 攻防世界 xor crypto

ctf 专栏收录该内容
200 篇文章 23 订阅
订阅专栏

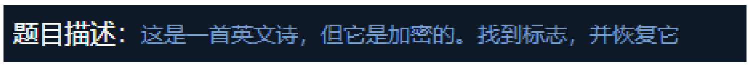
前言

继续ctf的旅程
攻防世界Crypto高手进阶区的6分题
本篇是xor_game的writeup

发现攻防世界的题目分数是动态的
就仅以做题时的分数为准了

解题过程

题目描述


题目描述：这是一首英文诗，但它是加密的。找到标志，并恢复它

得到一段py和一个密文
分别如下

```python
from Crypto.Util.strxor import strxor
import base64
import random


def enc(data, key):
    key = (key * (len(data) / len(key) + 1))[:len(data)]
    return strxor(data, key)


poem = open('poem.txt', 'r').read()
flag = "hctf{xxxxxxxxxxx}"

with open('cipher.txt', 'w') as f:
    f.write(base64.b64encode(enc(poem, flag[5:-1])))
    f.close()
```

```
ciMbOQxffx0GHQtSBB0QSQIORihXVQAUOUkHNgQLVAQcAVMAAAMCASFEGQYcVS8BNh8BGAoHFlMAABwCTSVQC2UdMQx5FkkGEQQAAVMAAQtHRCNL
F0NSORscMkkaHABSExIYBQseUmBCFgtSKwEWfwELFRcGbzwEDABHVS8DDAcXfwUcMQwCDUUBCgYYSQEBATNKGwQeOkkbPhsYERYGDB0TYzwCUSVC
DE8dKh0BNg4GAAkLSVMWHBpHQCxQF08AOhkWPh1OAA0XRQQRBQJKQyVKFghSMA95Gh8LGhEHBB8YEE4UViFaEQEVfwAdfx0GEUUWAAARGxpHTiFQ
ERx4FkkROgUHERMXRTpUCANtYy9RFk8TLEkHNwxOFhcbAhsASR0STC1GCk8UMwYEOhsdfiEdRR0bHU4QSDRLHR0XO0kGMQ0LEgATERYQSQgORDJa
WAsXMgYdfxsbGAB4LRYVGxpHUyFXHU8TMQ1TPRsLFREaDB0TSRoIASJGGR1SKwEWfwUBFQFSChVUHQYCASNWFQ0XLRocMgxkNgoAABd+PRkIKwkD
EAoTLQ1TKwELVAgHFhoXRU4BUy9OWBsaOkkeMAYAVAQcAVMXCBwEQDNQci4HJwAfNggcDUUXHQcGDAMCASFGCxsaOh0aPAAdGUUQBBoASRoIASNC
CBsHLQxTMgAdABx4IxoYBQcJRmBXEApSNgcHOgcdEUUeDBURRU4FVDQDGQMBMEkVNgUCHQsVRQccDE4XVDJGcjsaOhsWfwgcEUUTCQQVEB1HTCVO
Fx0bOhpTKwEcGxAVDRwBHU4TSSUDHQ4AKwF5FkkMEQkbAAURSSdHQC0pPAYXO0kSLEkaHABSFAYdDBpHQyVCDRsLfwYVfwgbABAfC1MYDA8RRDMp
KwcXMQ5TNhpOGgoGRRAcCAEUDWBQFQAZOkkUOhoaARcXbzYCDABHVilPDE8TMxocfxsLAAQbCxYQSQwITyUDCB0dKg0fJkk/HQsVRTURBwlHTDVQ
GwMXVSYQPBwCAG8mDQERDGQuAShGGR1SMwYFOkVOPUUQAB8dDBgCASlNWAMdKQx5EwYYEUUbFlMVSR4ITiwDFwlSLB0BKg4JGAwcAlMWBRsCDCdR
HQocfwgfOAgLfiQBRRcRGgELQDRGWAIbPBsccgsbBhYGRRwSSRkOTyQpOgMXOg0aMQ5OAA0ACgYTAU4KWGBVHQYcLGMqOggcB0UBERIAAAEJRCQD
EQFSKwEWfwsLGAwXA3kyBhsVKwkDGgoeNgwFOkkaHAQGRRIYBU4EQC4DEAoTLWM2KQwAVAQcERoXAB4GVSUDHAYBPBsWKwxCVCxSCBYASRoPRGBM
DAcXLUkHNwwHBkUdEh1+OgEKRGBAGQFSMQYHfw4cFRYCRQccDE4KTi1GFht4EwwVK0kaG0UGDRZULA8UVWBXF08VMEkkOhoaWEUGDRZUDQsGRWBO
DRwGfwccK0kcEREHFx1UHQFHTy9UEAoAOmMgOgxCVCxSEhYVG049QC4DPgMdKAwBLEkBGkUfHFMcDA8DDWBKFk8UKgUffwsCGwofRRIYBgAAATRL
HU8FPhBTPgUCVBEaAFMDCBdtZzJGCRoXMR0fJkkDHRYBABdUGgEKRGwDGhoGfwgfLAZOEAAXFR8NSQMIVyVHWA0Lfx4aMQ1CVAMACgAARU4UTy9U
WAAAfxsSNgdkMgwEAHkkGw8NTyEDKA4APgQaKwhCVBYdCh1UCB1HUi9MFk8TLGMfNg8LVAcXRRERCBsTSCZWFE8eNgIWfxobGQgXF1MSBQEQRDJQ
WA4cO0kXOggaHEUeDBgRSQ8SVTVOFk8eOggFOhpkNQkBClMXCBwCASFBFxoGfx4bPh1OHAQB
```

有点类似这题
https://findneo.github.io/180527suctf/#Cycle

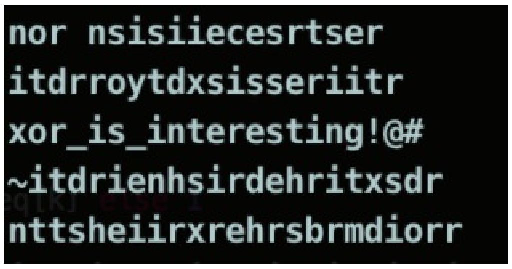
挪用下脚本

```python
# coding:utf8
 # by https://findneo.github.io/
def getCipher(file='cipher.txt'):
'''从文件中读取十六进制串，返回十六进制数组
'''
    c=''.join(map(lambda x:x.strip(),open('cipher.txt').readlines())).decode('base_64')
    cc= [ord(i) for i in c]
    # print cc,len(cc)
    return cc
    # c = open(file).read()
    # codeintlist = []
    # codeintlist.extend(
    #     (map(lambda i: int(c[i:i + 2], 16), range(0, len(c), 2))))
    # return codeintlist


def getKeyPool(cipher, stepSet, plainSet, keySet):
''' 传入的密文串、明文字符集、密钥字符集、密钥长度范围均作为数字列表处理.形如[0x11,0x22,0x33]
    返回一个字典，以可能的密钥长度为键，以对应的每一字节的密钥字符集构成的列表为值，密钥字符集为数字列表。
```

```
        形如{
                1:[[0x11]],
                3:[
                    [0x11,0x33,0x46],
                    [0x22,0x58],
                    [0x33]
                   ]
            }
'''
    keyPool = dict()
    for step in stepSet:
        maybe = [None] * step
        for pos in xrange(step):
            maybe[pos] = []
            for k in keySet:
                flag = 1
                for c in cipher[pos::step]:
                if c ^ k not in plainSet:
                    flag = 0
            if flag:
                maybe[pos].append(k)
    for posPool in maybe:
        if len(posPool) == 0:
            maybe = []
            break
    if len(maybe) != 0:
        keyPool[step] = maybe
return keyPool


def calCorrelation(cpool):
'''传入字典，形如{'e':2,'p':3}
    返回可能性，0~1,值越大可能性越大
    (correlation between the decrypted column letter frequencies and
    the relative letter frequencies for normal English text)
'''
    frequencies = {"e": 0.12702, "t": 0.09056, "a": 0.08167, "o": 0.07507, "i": 0.06966,
                   "n": 0.06749, "s": 0.06327, "h": 0.06094, "r": 0.05987, "d": 0.04253,
                   "l": 0.04025, "c": 0.02782, "u": 0.02758, "m": 0.02406, "w": 0.02360,
                   "f": 0.02228, "g": 0.02015, "y": 0.01974, "p": 0.01929, "b": 0.01492,
                   "v": 0.00978, "k": 0.00772, "j": 0.00153, "x": 0.00150, "q": 0.00095,
                   "z": 0.00074}
    relative = 0.0
    total = 0
    fpool = 'etaoinshrdlcumwfgypbvkjxqz'
    total = sum(cpool.values())  # 总和应包括字母和其他可见字符
    for i in cpool.keys():
        if i in fpool:
            relative += frequencies[i] * cpool[i] / total
    return relative


def analyseFrequency(cfreq):
    key = []
    for posFreq in cfreq:
        mostRelative = 0
    for keyChr in posFreq.keys():
        r = calCorrelation(posFreq[keyChr])
        if r > mostRelative:
            mostRelative = r
```

```python
            keychar = keyChr
    key.append(keychar)

return key


def getFrequency(cipher, keyPoolList):
''' 传入的密文作为数字列表处理
    传入密钥的字符集应为列表，依次包含各字节字符集。
        形如[[0x11,0x12],[0x22]]
    返回字频列表，依次为各字节字符集中每一字符作为密钥组成部分时对应的明文字频
        形如[{
                0x11:{'a':2,'b':3},
                0x12:{'e':6}
             },
             {
                0x22:{'g':1}
             }]
'''
    freqList = []
    keyLen = len(keyPoolList)
    for i in xrange(keyLen):
        posFreq = dict()
        for k in keyPoolList[i]:
            posFreq[k] = dict()
            for c in cipher[i::keyLen]:
                p = chr(k ^ c)
                posFreq[k][p] = posFreq[k][p] + 1 if p in posFreq[k] else 1
        freqList.append(posFreq)
    return freqList


def vigenereDecrypt(cipher, key):
    plain = ''
    cur = 0
    ll = len(key)
    for c in cipher:
        plain += chr(c ^ key[cur])
        cur = (cur + 1) % ll
    return plain


def main():
    ps = []
    ks = []
    ss = []
    ps.extend(xrange(0xff))
    ks.extend(xrange(0x20,0x80))
    ss.extend(xrange(1, 50))
    cipher = getCipher()

    keyPool = getKeyPool(cipher=cipher, stepSet=ss, plainSet=ps, keySet=ks)
    for i in keyPool:
        freq = getFrequency(cipher, keyPool[i])
        key = analyseFrequency(freq)
        print ''.join(map(chr,key))


if __name__ == '__main__':
    main()
```

得到



```
nor nsisiiecesrtser
itdrroytdxsisseriitr
xor_is_interesting!@#
~itdrienhsirdehritxsdr
nttsheiirxrehrsbrmdiorr
```

得到flag: `hctf{xor_is_interesting!@#}`

结语

xortools也可以用