ctf-wechall-Crypto

原创

逃课的小学生 于 2018-07-23 17:19:49 发布 1809 收藏
分类专栏： ctf wechall crypto 文章标签： ctf wechall crypto

1.Caesar I

获得一段字符串，根据提示使用凯撒解密，即可获得唯一一句相对有意义的话QUICK BROWN FOX JUMPS OVER THE LAZY DOG OF CAESAR AND YOUR UNIQUE SOLUTION IS.....。

2.Transposition I

获得一段字符串，根据提示使用置换加密，看到开头字符串“oWdnreuf.lY uoc”，猜测开头应该是“Wonderful. You"，以此作为依据，发现当以长度为6做划分可得到有意义的句子Wonderful. You can read the message way better when the letters are in correct order. I think you would like to see your password now: .....。，

3.Simple Substitution I

获得一段字符串，根据提示是做了单表替换，使用Substitution Cipher Solver工具快速对该字符串进行解密，观察，对不对的单词进行微调后（http://cryptoclub.org/tools/cracksub_topframe.php）得到结果by the almighty god you can read this my friend i am impressed very well done your solution key is pplrlpcdmsfh this little challenge was not too hard was it(注意大小写）

4.Caesar II

由题意可知这一次使用的是移位加密，不同于凯撒加密的是范围由26个字母变为128个ascii字符，但思路一致，可通过

```
miwen="15 3D 3D 32 20 38 3D 30 7A 20 47 3D 43 20 41 3D 3A 44 33 32 20 3D 3C 33 20 3B 3D 40 33 20 31 36 2F 3

list1=miwen.split(' ')
mingwen=""
sign=True

for j in xrange(1,128):
    for i in list1:
        mingwen=mingwen+chr((int(i,16)+j)%128)
    for i in mingwen:
        if ord(i)>126 or ord(i)<33:
            sign=False
            break
    if sign:
        print mingwen
        print j

    mingwen=""
    sign=True
```

找出基本由正常英文字母和一些标点符号组成的句子"GoodRjob,RyouRsolvedRoneRmoreRchallengeRinRyourRjourney.RThisRoneRwasRfairlyReasyRtoRcrack.R128RkeysRisRaRquiteRsmallRkeyspace,RsoRitRshouldn'tRhaveRtakenRyouRtooRlongRtoRdecryptRthisRm后将”R“改为空格，就得到正常英语句子，即可得到结果

5.Digraphs

由题意可知，这里的加密方法是将一个字符加密为两个字符。所以首先编程提取出组成句子的由两个字符组成的基本单元，并将其随机替换成一个字符组成的单元（即对任意一个基本单元都替换成唯一对应的一个字符，不同基本单元对应字符不同），此时题目变为了单表替换（但其中由大小写和标点），观察字符串，我们看到句子结尾和第一个单词结尾相同，意味着由30个密文字符（15个明文字符）组成的单词可单独成句，可以想到“congratulations“，然后剩余字符按照正确英语单词和语法进行填充。即可获得句子Congratulations. You decrypted this message successfully. Has not too difficult either? was itx Hell? good job. Anter this keyword as solution: .....代码如下：

```
s="tchwxlfpdgzkkvkafczkkvhlhwxlcsxq fkhwka kjxabedgroockvxakj kvzxhlcs anxacscszkfpxa cskabebexacscsyskafcf
'''
将组成句子的基本单元提取出来（字符串不能有空格）
lists=s.split(" ")
list1=[]
i=0
str1=""
for i in lists:
    if list1.count(i)==0:
        list1.append(i)

print list1
'''
#将基本单元随即换成字符并作调整知道得到正确的句子
dict1={'tc':'C','hw':'o','xl':'n','fp':'g','dg':'r','zk':'a','kv':'t','ka':'u','fc':'l','hl':'i','cs':'s','
mingwen=""
str1=""
i=0
while i<len(s):
    if s[i]==" ":
        mingwen+=" "
        i=i+1
    else:
        str1=s[i:i+2]
        mingwen+=dict1[str1]
        i=i+2

print mingwen
```

6.Baconian

由题意可知这是使用培根加密，观察字符串可知道是将大小写换成不同字母进行培根解密即可

```
s="BaCoN's cIphEr or THE bacOnIAN CiPHer iS a meThOD oF sTEGaNOGrapHY (a METhoD Of HidIng A sECRet MeSsaGe

codebook1 = {
    'A':"aaaaa",
    'B':"aaaab",
    'C':"aaaba",
    'D':"aaabb",
    'E':"aabaa",
    'F':"aabab",
    'G':"aabba",
    'H':"aabbb",
    'I':"abaaa",
    'J':"abaab",
    'K':"ababa",
    'L':"ababb",
    'M':"abbaa",
    'N':"abbab",
    'O':"abbba",
    'P':"abbbb",
    'Q':"baaaa",
    'R':"baaab",
    'S':"baaba",
    'T':"baabb",
```

```
    'U':"babaa",
    'V':"babab",
    'W':"babba",
    'X':"babbb",
    'Y':"bbaaa",
    'Z':"bbaab",
}
def zhuanhua(s):
    str1=""
    j=0
    for i in s:
        if ord(i)>64 and ord(i)<91:
            str1=str1+"b"
            j=j+1
        elif ord(i)>96 and ord(i)<123:
            str1=str1+'a'
            j=j+1
        if j==5:
            str1+=" "
            j=0
    return str1
def decode(s):
    cipher=""
    ss = s.split(" ")
    for c in ss:
        sign=True
        for k in codebook1.keys():
            if codebook1[k] == c:
                cipher+=k
                sign=False
                break
        if sign:
            #cipher+=c
            pass
    return(cipher)

a=zhuanhua(s)
b=decode(a)
print b
mingwen=""
for i in b:
    if i=="X":
        mingwen+=" "
    else:
        mingwen+=i
print mingwen.lower()
```

注意要将无法匹配字典中的单元扔掉，且原题中有将‘x’转化为空格的提醒。

7.Substitution II（与Digraphs基本相同）

由题意可知，这里的加密方法是将一个字符加密为0-255中随机一个16进制数。所以首先编程提取出组成句子的由组成的16进制数基本单元，并将其随机替换成一个字符组成的单元（即对任意一个基本单元都替换成唯一对应的一个字符，不同基本单元对应字符不同），此时题目变为了单表替换（但其中由大小写和标点），观察字符串，我们看到句子结尾和第一个单词结尾相同，意味着由30个密文字符（15个明文字符）组成的单词可单独成句，可以想到“congratulations“，然后剩余字符按照正确英语单词和语法进行填充。即可获得句子 Congratulations! This one was harder, but you got it! very well done fellow hacker! The problem with this cipher is that the key is pretty long! I will come up with a better encryption sheme any soon! your solution is: .....代码如下：

```
s="79 85 89 11 4B 63 52 E9 96 63 52 00 85 89 D0 D7 6D B5 F6 00 D0 6D 85 89 98 6D E1 63 D0 6D F6 63 4B 32 98
'''
lists=s.split(" ")
list1=[]
i=0
str1=""
for i in lists:
    if list1.count(i)==0:
        list1.append(i)

print list1

'''
dict1={'79':'C','85':'o','89':'n','11':'g','4B':'r','63':'a','52':'t','E9':'u','96':'l','00':'i','D0':'s','
lists=s.split(" ")
str1=""
for i in lists:
    str1+=dict1[i]

print str1
```

8.Pimitive Encryption

由题目可知原文件zip与key做异或得到我们下载的zip，用winhex打开我们下载的zip与zip文件头部做异或可得到“0x332e31343135”，发现其对应字符“3.1415”，猜测是用圆周率作为key进行加密，用圆周率解密后得到zip文件可以解压：

```
mask_in=open("pimitive.zip","rb")
mask= bytearray(mask_in.read())
mask_in.close()
key_in = open('1.txt','rb')
key = bytearray(key_in.read(len(mask)))
key_in.close()
for x in xrange(len(mask)):
    key[x] ^= mask[x]
key_out = open('masked_key.zip','wb')
key_out.write(key)
key_out.close()
```

获得bmp文件，打开即可获得结果

9.Chessy Hawks

获得一个棋盘，可以想到直角坐标系，以左下角为原点，x轴坐标为a,b,c,d.....，y轴坐标为1,2,3,4....，组合在每一点可形成一个十六进制数，棋盘上有两种颜色圈住的数字,猜测一种为加，另一种为减，可得结果“6652333350776e31335a

10.Bacon Returns

由题意可知这是使用了培根加密，但发现在结尾处全都是由小写字母组成，猜测密文可能只有大写字母，将其按正常字母顺序对半分开，前一部分和后一部分对应不同字符，用培根解密即可得到结果：you can read the hidden message so i will tell you the solution which is twelve random letters .....

```
s="bacON's CIpHER OR tHe bacOnIaN CIPHeR iS a meTHoD oF StEGAnoGrApHy (A meThOD Of hidiNG a SecReT meSsAGE
codebook1 = {
    'A':"aaaaa",
    'B':"aaaab",
    'C':"aaaba",
    'D':"aaabb",
    'E':"aabaa",
    'F':"aabab",
    'G':"aabba",
    'H':"aabbb",
    'I':"abaaa",
    'J':"abaab",
    'K':"ababa",
    'L':"ababb",
    'M':"abbaa",
    'N':"abbab",
    'O':"abbba",
    'P':"abbbb",
    'Q':"baaaa",
    'R':"baaab",
    'S':"baaba",
    'T':"baabb",
    'U':"babaa",
    'V':"babab",
    'W':"babba",
    'X':"babbb",
    'Y':"bbaaa",
    'Z':"bbaab",
}


def zhuanhua1(s):
    str1=""
    j=0
    for i in s:
        if ord(i)>64 and ord(i)<78:
            str1=str1+"a"
            j=j+1
        elif ord(i)>77 and ord(i)<91:
            str1=str1+'b'
            j=j+1
        if j==5:
            str1+=" "
            j=0
    return str1
def decode(s):
    cipher=""
```

```
    ss = s.split(" ")
    for c in ss:
        sign=True
        for k in codebook1.keys():
            if codebook1[k] == c:
                cipher+=k
                sign=False
                break
        if sign:
            #cipher+=c
            pass
    return(cipher)

a=zhuanhua1(s)
b=decode(a)
print b
mingwen=""
for i in b:
    if i=="X":
        mingwen+=" "
    else:
        mingwen+=i
print mingwen.lower()
```

11.Gizmore Encryption

由加密代码可得开始x=1,k=-1，每一轮k=k+x，当k大于密钥长度时k=0，x=x+1，当x大于密钥长度时x=1，然后密文对应字节等于明文对应字节异或密钥[k]异或e，同时已知密钥长度位11，且是a-z，A-Z中，而明文在正常英文字符中，可通过对每一位密钥进行分别加密和语义，确定密钥以及明文，代码如下：

```
s="64 74 7A 59 6B 3A 36 5B 7E 37 72 49 73 69 65 2A 6D 45 5E 73 37 55 6B 7F 0C 77 73 49 36 4D 70 40 6F 0C 63
list1=s.split(" ")
keystr1="abcdefghigklmnopqrstuvwxyzABCDEFGHIJKLMNPQRSTUVWXYZ"
mingwen=""
lkey=11
for ch in keystr1:
 #从0开始增长密钥，将可以确定的密钥位确定，不可以确定的密钥位可以线填充，最后根据语义确定
    key='I'+ch
    x=1
    k=-1
    sign=True
    for i in list1:
        k=k+x
        if k>=lkey:
            k=0
            x=x+1
            if x>=lkey:
                x=1
        if (k%lkey)<len(key):
            mingwench=(ord(key[k%lkey])^(int(i,16)^ord('e')))%128
            if (mingwench<91 and mingwench>64) or (mingwench<123 and mingwench>96)or (mingwench<58 and ming
                mingwen+=chr(mingwench)
            else:
                sign=False
                break

        else:
            mingwen+=" "
    if sign:
        print mingwen
        print ch
    mingwen=""
```

12.Save the world

有题可得3个对相同明文rsa加密获得的密文c1,c2,c3和三个公钥（n1,e1),（n2,e2),（n3,e3)，且e1=e2=e3=3。这时我们可以用RSA低幂爆破来直接计算明文，而且我们还能得知m的三次方除n1为c1，除n2为c2，除n3为c3，可用中国剩余定理计算出一个基础数在进行爆破，代码如下：

```
import gmpy2
gmpy2.get_context().precision=5000

n1=671083372851301528411425570489798363926593218918571012267323372818703192196300633084252610010945683500460
c1=590583470961093711687952521228462641644518583865053732247172513867600707159983216163228720454963609260100
n2=722918206448018510503308481101594092020489195345279551641203027623966445879735600569615595072633862765900
c2=688326368633095218427382289121359100592470516893797580305646937922340954461468080988808803277067259694070
n3=104783788987827067930027452786608829382069127222045660514968202129202628419456254428420482380928992774980
c3=315302339544352348716154744610822578608145748525168323485667507452479805297780607411210625649213062750420
N1=n2*n3
N2=n1*n3
N3=n1*n2
N=n1*n2*n3
proN1=gmpy2.invert(N1,n1)
proN2=gmpy2.invert(N2,n2)
proN3=gmpy2.invert(N3,n3)
N=((c1*N1*proN1)%N+(c2*N2*proN2)%N+(c3*N3*proN3)%N)%N
i=1
while True:
    res=gmpy2.iroot(N*i,3)
    if res[1]:
        print res[0]
        break
    i=i+1
    print i
```

根据题意将结果后二十位填入即可