CTF crypto 密码类 题型积累

原创

quarter26 于 2019-04-04 13:52:56 发布 16760 收藏 199
分类专栏： CTF 密码学 文章标签： ctf python crypto 密码 渗透

本文链接：https://blog.csdn.net/weixin_42037232/article/details/89018997

一点点积累一下 CTF 中 crypto 密码类的题型。

感谢 BugKu 提供了很多加解密工具，链接：http://tool.bugku.com/

感谢 SSL在线工具网址提供了很多工具，链接：http://www.ssleye.com/

我常用的 text bin hex base64 dec 转码工具：https://conv.darkbyte.ru/

1. 摩斯密码

特征：点和横的组合。

在线工具：http://www.zhongguosou.com/zonghe/moErSiCodeConverter.aspx 或 http://tool.bugku.com/mosi/

e.g.

-... -.- -.-. - ..-. -- .. ... -.-.

解密结果：BKCTFMISC

2. 栅栏密码

特征：大小写和字符，其实就是分组替换加密

在线工具：http://tool.bugku.com/jiemi/

e.g.

一只小羊翻过了2个栅栏

KYsd3js2E{a2jda}

解密结果：KEY{sad23jjdsa2}

3. Ook 密码

特征：Brainfuck 类型密码，密文由 Ook 和 三种标点 . ! ? 构成，不见得都得用上，有的是Ook，有的没有Ook只有标点。

在线工具：https://tool.bugku.com/brainfuck/

例子就不举了

4. Brainfuck 密码

特征： + - [ ] { } . < > 这些符号构成的密码

在线工具：http://tool.bugku.com/mosi/

e.g.

```
+++++ +++++ [->++ +++++ +++<] >++.+ +++++ .<+++ [->-- -<]>- -.+++ +++.<
++++[ ->+++ +<]>+ +++.< +++[- >---< ]>--- .---- .<+++ ++++[ ->--- ----<
]>--- ----- ----- .<+++ ++++[ ->+++ +++++< ]>+++ ++.<+ +++++ +[->- -----
-<]>. <++++ ++++[ ->+++ +++++ <]>++ .<+++ [->-- -<]>- ----. <++++ +++[-
>---- ---<] >---- ----. +++++ +..++ +++.+ .<+++ [->-- -<]>- --.<+ +++++
+[->+ +++++ +<]>+ ++.++ +.+++ +++++ +.--- -.+++ ++.<+ ++[-> +++<] >++++
++.<
```

解密结果： flag{ok-c2tf-3389-admin}

5. 凯撒密码：

特征：英文字母排序往后延，密钥是 1 <= k <= 26，y = x + k；

在线工具：http://tool.bugku.com/jiemi/ 或者 CrypTool

e.g.

密文：
MSW{byly_Cm_slol_lYqUlx_yhdls_Cn_Wuymul_il_wuff_bcg_pCwnll_cm_u_Yrwyffyhn_guh_cz_sio_quhn_ni_ay

放在CrypTool 里跑一下就好了，或者 1 - 25 整个都算一遍，找个明文出来：

SYC{here_ls_yOur_rEwArd_enjOy_lt_Caesar_or_call_him_vlctOr_is_a_Excellent_man_if_you_want_to_get_hi

6. Base64 编码 及其混合

特征：看到密文最后两个字符是相同的，就有可能是Base64编码过的，因为Base64编码结尾通常是 ==

有可能和其他密码混合使用，进行二次加密/编码。

Base64 是编码（encode），不是加密（encrypt）

在线工具：这道题python简单点，还有一个解码工具平时用的也多： https://conv.darkbyte.ru/

e.g.

也是Bugku里的一道题：

e6Z9i~]8R~U~QHE{RnY{QXg~QnQ{^XVlRXlp^Xl5Q6Q6SKY8jUAA

结尾是 AA，有点像Base64编码，但是被加密过了，这么蠢的加密方式又有点像凯撒，看ASC Ⅱ码，发现 = 是61， A是 65，往后延了4位，那么我们把密文所有数据的ASC Ⅱ码 减4，得到Base64的代码，再解码即得到明文flag： key{68743000650173230e4a58ee153c68e8}

Python 代码：

```
import base64

# strs is the ciphertext
strs = "e6Z9i~]8R~U~QHE{RnY{QXg~QnQ{^XVlRXlp^XI5Q6Q6SKY8jUAA";

# offset is the key of caesar cipher
offset = ord('A')-ord('=')
print(offset)

# caesar cipher shifting as base64_str
list_str=[]
for i,ch in enumerate(strs):
    c = chr(ord(ch)- offset )
    list_str.append(c)
base64_str = "".join(list_str)
print(base64_str)

# base64 decoding
plain_text = base64.b64decode(base64_str)
print(plain_text)
```

7. 类栅栏密码

特征：会给你一串乱序的密文 C，同时给你从1 到 n 的一串乱序的数，找下 C 对于 n 的最大公约数，然后将他们重新排序。

动手画一画就出来了

e.g.

BugKu中的一道题：

lf5{ag024c483549d7fd@@1}　，一张纸条上凌乱的写着2 1 6 5 3 4

2	1	6	5	3	4
l	f	5	{	a	g
0	2	4	c	4	8
3	5	4	9	d	7
f	d	@	@	1	}

再按照123456的顺序数从第一行排到最后一行

flag{52048c453d794df1}@@

提交 flag发现不对，这两个@@有点奇怪，去掉@就成功了，那么一般这种字符都是迷惑人的。

8. 由0和1组成的摩斯密码

特征：密文由0和1构成，可能由空格、tab或则其他字符分割，去掉这些分割后，0和1的总数是5的倍数

将 0 和 1 替换为 . 或 - ，多试几次就可能有发现。

e.g.

BugKu中的一道题：

0010 0100 01 110 1111011 11 11111 010 000 0 001101 1010 111 100 0 001101 01111 000 001101 00 10 1 0 010 0 000 1 01111 10 11110 101011 1111101

把0换成点 . 1换成 - 后：

..-. .-.. .- --. ----.-- -- ----- .-. ... . ..--.- -.-. --- -.. . ..--.- .---- ... ..--.- .. -. - . .-. . ... - .---- -. ----. -.-.-- -----.-

在线工具解密一下，得到：

FLAG%u7bM0RSE_CODE_1S_INTEREST1N9!%u7d

结果有点奇怪，但八九不离十了。把%u7b 换成 { 把%u7d 换成 }，然后大小写切换

flag{m0rse_code_1s_interest1n9!}

这类题很多都只能靠直觉。。。大写提交不成就用小写试试。

9. Base64 等等各种编码来回转（不知道是谁想得出这么傻b的题型，很无聊）

特征：给你一段Base64编码，然后转，各种转，瞎tm转吧。

主要是考察对各种编码模式的认识，这里小结一下：

编码模式	定义	特征
Base64	Base64是一种基于64个可打印字符来表示二进制数据的表示方法	由 A-Z，a-z，0-9，+/ 这64各字符构成，结尾通常是两个等号 ==
Hex 十六进制	就是16进制，有很多表现形式，由0-9，A-F构成	由0-9，A-F构成，前面肯能是 /x 或者 0x等等 如：\x5c\x75\x30 或 0x5c 0x75 0x30
Oct 八进制	0-7	斜杠 / 加数字 1 - 7，例如：\134\170\63\71
unicode	分很多种，utf-8 utf-16等等	/uxxxx，其中xxxx可能是16进制
URL转义	把很多字符转为百分号开头的 %	比如空格是 %20
html 转义	&# 开头	由 &# 开头，后面加数字

e.g.

密文太长，贴个链接：https://ctf.bugku.com/files/39488475fd87c064f9401eec2299c03e/1.txt

这种题就是根据编码模式来回转义就好了，真是无聊。。

10. 键盘格子密码

特征：由三到四个英文字母或数字为一组。

在键盘上找对应的键位，中间围起来的就是密文（这能能想得出。。。）

e.g.

r5yG lp9l BjM tFhBT6uh y7iJ QsZ bhM

t o n g y u a n

11. 托马斯杰斐逊 转轮密码

特征：给你一个密码表，n行的26个字母，key是 1 - n 的数列， 密文是 n 个英文字母

根据 key 找对应行的密码表，然后在密码表上找密文字母，以这个字母为开头，重新排序。

e.g.

1： <ZWAXJGDLUBVIQHKYPNTCRMOSFE <
2： <KPBELNACZDTRXMJQOYHGVSFUWI <
3： <BDMAIZVRNSJUWFHTEQGYXPLOCK <
4： <RPLNDVHGFCUKTEBSXQYIZMJWAO <
5： <IHFRLABEUOTSGJVDKCPMNZQWXY <
6： <AMKGHIWPNYCJBFZDRUSLOQXVET <
7： <GWTHSPYBXIZULVKMRAFDCEONJQ <
8： <NOZUTWDCVRJLXKISEFAPMYGHBQ <
9： <QWATDSRFHENYVUBMCOIKZGJXPL <
10： <WABMCXPLTDSRJQZGOIKFHENYVU <
11： <XPLTDAOIKFZGHENYSRUBMCQWVJ <
12： <TDSWAYXPLVUBOIKZGJRFHENMCQ <
13： <BMCSRFHLTDENQWAOXPYVUIKZGJ <
14： <XPHKZGJTDSENYVUBMLAOIRFCQW <

密钥： 2,5,1,3,6,4,9,7,8,14,10,13,11,12

密文：HCBTSXWCRQGL ES

在第 2 行密码表中找H开头的字母，然后以H开头再到尾过一遍，以此类推，整理出另一个密码表：

HGVSFUWIKPBELNACZDTR X MJQOY
CPMNZQWXYIHFRLABEUOT S GJVDK
BVIQHKYPNTCRMOSFEZWA X JGDLU
TEQGYXPLOCKBDMAIZVRN S JUWFH
SLOQXVETAMKGHIWPNYCJ B FZDRU
XQYIZMJWAORPLNDVHGFC U KTEBS
WATDSRFHENYVUBMCOIKZ G JXPLQ
CEONJQGWTHSPYBXIZULV K MRAFD
RJLXKISEFAPMYGHBQNOZ U TWDCV
QWXPHKZGJTDSENYVUBML A OIRFC
GOIKFHENYVUWABMCXPLT D SRJQZ
LTDENQWAOXPYVUIKZGJB M CSRFH
ENYSRUBMCQWVJXPLTDAO I KFZGH
SWAYXPLVUBOIKZGJRFHE N MCQTD

然后在这里找一些比较明显的（语句通顺的）话，就是flag。

XSXSBUGKUADMIN
提交不对的话就大小写换一下：xsxsbugkuadmin

这个其实可以写个程序出来，遍历密码表即可。

```
# Rotor cipher decoder
# parameter input
rotor = [
    "ZWAXJGDLUBVIQHKYPNTCRMOSFE", "KPBELNACZDTRXMJQOYHGVSFUWI",
    "BDMAIZVRNSJUWFHTEQGYXPLOCK", "RPLNDVHGFCUKTEBSXQYIZMJWAO",
    "IHFRLABEUOTSGJVDKCPMNZQWXY", "AMKGHIWPNYCJBFZDRUSLOQXVET",
    "GWTHSPYBXIZULVKMRAFDCEONJQ", "NOZUTWDCVRJLXKISEFAPMYGHBQ",
    "QWATDSRFHENYVUBMCOIKZGJXPL", "WABMCXPLTDSRJQZGOIKFHENYVU",
    "XPLTDAOIKFZGHENYSRUBMCQWVJ", "TDSWAYXPLVUBOIKZGJRFHENMCQ",
    "BMCSRFHLTDENQWAOXPYVUIKZGJ", "XPHKZGJTDSENYVUBMLAOIRFCQW"
]

cipher = "HCBTSXWCRQGLES"

key = [2, 5, 1, 3, 6, 4, 9, 7, 8, 14, 10, 13, 11, 12]

tmp_list=[]

for i in range(0, len(rotor)):
    tmp=""
    k = key[i] - 1
    for j in range(0, len(rotor[k])):
        if cipher[i] == rotor[k][j]:
            if j == 0:
                tmp=rotor[k]
                break
            else:
                tmp=rotor[k][j:] + rotor[k][0:j]
                break
    tmp_list.append(tmp)
# print(tmp_list)

message_list = []
for i in range(0, len(tmp_list[i])):
 tmp = ""
 for j in range(0, len(tmp_list)):
  tmp += tmp_list[j][i]
 message_list.append(tmp)

print(message_list)
```

12. Base91 编码

特征：基本上是键盘上所有可打印的 ASC II 字符（0x21-0x7E），A-Z、a-z、1 - 9、 !@#$%^&*()_+-={}[]|\:;'"<,>.?/~· 之类的。

参考： http://base91.sourceforge.net/ （里面有工具源码）

在线工具：http://ctf.ssleye.com/base91.html

e.g.

@iH<,{bdR2H;i6*Tm,Wx2izpx2!

13. 核心价值观编码

特征：富强民主文明和谐自由平等公正法治爱国敬业诚信友善

我真tm服了

谁知道这啥编码规则？？？？？？？

在线工具：http://ctf.ssleye.com/cvencode.html

e.g.

公正公正公正诚信文明公正民主公正法治法治友善平等和谐敬业和谐富强和谐富强和谐文明和谐平等公正公正和谐法治公正公正公正文明和谐民主和谐敬业和谐平等和谐敬业和谐敬业和谐和谐和谐公正法治友善法治

14. Linux系统的 shadow 文件格式

特征：就是Linux的shadow文件格式。。。

工具：Kali Linux 中的 John

e.g.

root:6HRMJoyGA$26Flgg6CU0bGUOfqFB0Qo9AE2LRZxG8N3H.3BK8t49wGlYbkFbxVFtGOZqVlq3qQ6k0oetDbn2aVzdhuVQ6US.:17770:0:99999:7:::
Linux的 /etc/shadow 文件存储了该系统下所有用户口令相关信息，只有 root 权限可以查看，用户口令是以 Hash + Salt 的形式保护的。
每个字段都用 “$” 或“:”符号分割；
第一个字段是用户名，如root ；
第二个字段是哈希算法，比如 6 代表SHA-512，1 代表 MD5；
第三个字段是盐，比如上面的 HRMJoyGA
第四个字段是口令+盐加密后的哈希值
后面分别是密码最后一次修改日期、密码的两次修改间隔时间（和第三个字段相比）、密码的有效期(和第三个字段相比）、密码修改到期前的警告天数（和第五个字段相比）、密码过期后的宽限天数（和第五个字段相比）、账号失效时间，这里不太重要要；

直接跑 John 试试

john shadow

如果解开了，加 --show 查看解密口令

john --show shadow

15. ZIP 伪加密

特征：一个ZIP压缩包，建议先读一下Zip文件解析与利用，里面提到：

一格zip文件有三个部分组成：压缩源文件数据区 + 压缩源文件目录区 + 压缩源文件目录结束标志；

每一部分都由明文 PK （50 4B）开始；

这是三个头标记，主要看第二个；

压缩源文件数据区：
50 4B 03 04：这是头文件标记

压缩源文件目录区：
50 4B 01 02：目录中文件文件头标记

后面两位（如 1F 00 或 3F 00）：压缩使用的 pkware 版本

再后面两位（如 14 00）：解压文件所需 pkware 版本

再后面两位：全局方式位标记（有无加密，这个更改这里进行伪加密，00 00 是无密码，改为09 00打开就会提示有密码了）

压缩源文件目录结束标志 ：
50 4B 05 06：目录结束标记

工具：ZipCenOp.jar 或 WinHex

e.g. BugKu 上的一道题 https://ctf.bugku.com/challenges#zip%E4%BC%AA%E5%8A%A0%E5%AF%86

1. 使用 ZipCenOp.jar

链接：https://pan.baidu.com/s/1yDcWWhY0lSlBArEJ4S6qUw
提取码：g3it
（需要java环境）windows下在cmd中输入：
java -jar ZipCenOp.jar r xxx.zip

直接破解ZIP包；

2. 使用 WinHex

我们用winhex打开压缩包，搜索504B，点击第二个504B，从后面找第七、八位，发现是 09 00，改为 00 00 即可。
这种方式只适用于ZIP的伪加密，真加密了此方法不适用。

16. RSA 加解密

特征：给一些 RSA 算法的参数，然后加密\解密消息获取 flag。

说一下 RSA 算法模式：

分三部分，密钥生成、加密、解密：

a) 密钥生成
   1) 选取两个长度为 K 的素数 P 和 Q，计算 N = P * Q；
   2) 计算 phi(N) = (P-1) * (Q-1)， 其中 phi(N) 是 Z_(N^*) 的阶；
   3) 随机选取一个int整数 e ∈ [1, phi(N) - 1]，使得 gcd(e, phi(N)) = 1；
   4) 计算它的逆 d，使得 [e * d mod phi(N)] = 1；
   5) 输出私钥和公钥 sk = (N, d), pk = (N, e)；

b) 加密
   c = m^e mod(N)

c) 解密
   m = c^d mod(N)

工具：RsaCtfTool （用于输出RSA参数） libnum （用于密文的计算）

具体参考Bugku-加密-rsa(WriteUp)，代码放在下面：

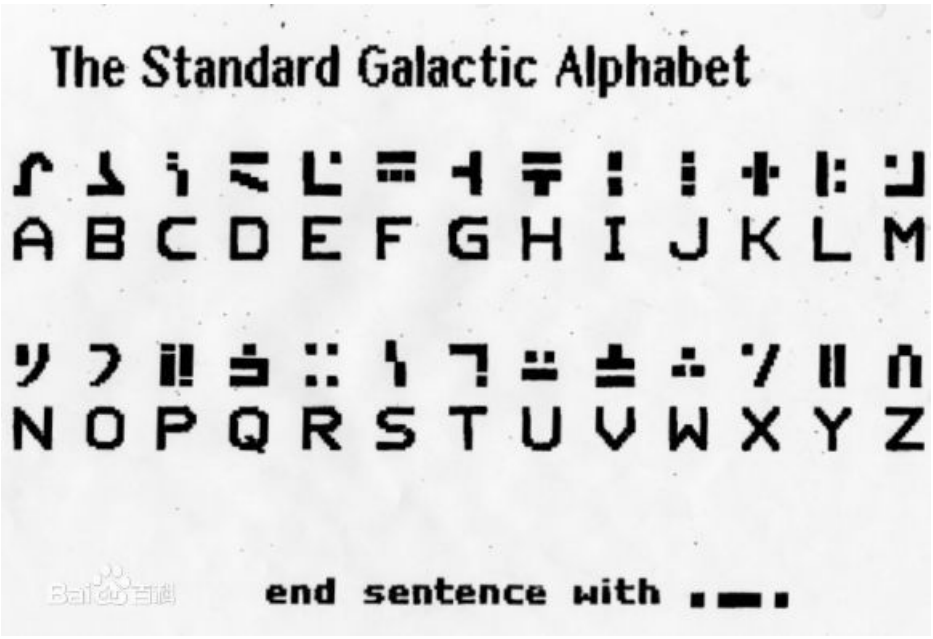
（记得提前安装上面两个工具，其中RsaCtfTool 依赖很多库，参考Github上的Readme；libnum的安装在git clone后在libnum的目录下运行 python setup.py install）

```
# compute public key as test.pub
python RsaCtfTool.py --createpub -n 46065781388428960989637205658554417248531811702624626389974432923749270

# compute private key as test.key
python RsaCtfTool.py --publickey test.pub --private > test.key

# compute RSA paras are in the file "info"
python RsaCtfTool.py --key test.key --dump > info
```

```
#coding:utf-8
from libnum import n2s,s2n
import base64


def gcd(a, b):
    if a < b:
        a, b = b, a
    while b != 0:
        temp = a % b
        a = b
        b = temp
    return a

def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

def modinv(a, m):
    g, x, y = egcd(a, m)
    if g != 1:
        raise Exception('modular inverse does not exist')
    else:
        return x % m

if __name__ == "__main__":
    p = 15991846970993213322072626901560749932686325766403404864023341810735319249066370916090640926219079 3
    q = 28805791771260259486856902729020438686670354441296247148207862836064657849735343618207098163901787 2
    e = 35461110244130757205657218182792589919834535022875373093108939327546391654445662689424541509610783 4
# tmp = base64.b64decode("qzogS7X8M3ZOpkUhJJcbukaRduLyqHAPblmabaYSm9iatuulrHcEpBmil7V40N7gbsQXwYx5EBH5r5V2H
#  =
    d = modinv(e, (p - 1) * (q - 1))
    # c=s2n(tmp)
    c=38230991316229399651823567590692301060044620412191737764632384680546256228451518238842965221394711848
    n = p * q
    m = pow(c, d, n)
    print n2s(m)
```

17. 标准银河字母

特征：游戏《指挥官基恩》系列（我TM （*！%……！（*&￥！@#！#。。。。。。）



18. 仿射密码 affine cipher

特征：可能会提示你是放射密码 affine，公式： y = k * x + b mod 26 （跟一元一次函数似的）
后面的取模，如果都是英文字母的话是26，不排除有其他形式，比如ASC Ⅱ什么的，取模可能会换。

工具：python代码

```
# Q: y = 17x-8 flag{szzyfimhyzd}

flag = "szzyfimhyzd"

flaglist = []

for i in flag:
    flaglist.append(ord(i)-97)

flags = ""
for i in flaglist:
    for j in range(0,26):
        c = (17 * j - 8) % 26
        if(c == i):
            flags += chr(j+97)
print('flag{'+ flags + '}')
```

19. 进制转换

特征：二进制 b开头，八进制 o开头，十进制 d开头，十六进制 x开头

e.g. BugKu里的一道题

```
d87 x65 x6c x63 o157 d109 o145 b100000 d116 b1101111 o40 x6b b1100101 b1101100 o141 d105 x62 d101 b1101001
```

```
s = 'd87 x65 x6c x63 o157 d109 o145 b100000 d116 b1101111 o40 x6b b1100101 b1101100 o141 d105 x62 d101 b110
ss = s.split()
sss = []
print(ss)
for i in ss:
    if i[0] == 'd':
        i = i[1:]
        i = int(i,10)
        i = chr(i)
        sss.append(i)
    elif i[0] == 'x':
        i = i[1:]
        i = int(i,16)
        i = chr(i)
        sss.append(i)
    elif i[0] == 'o':
        i = i[1:]
        i = int(i,8)
        i = chr(i)
        sss.append(i)
    elif i[0] == 'b':
        i = i[1:]
        i = int(i,2)
        i = chr(i)
        sss.append(i)
print(sss)
flag = ''.join(sss)
print(flag)
```