Bugku——加密wp

原创

ChanCherry、 于 2020-02-26 16:00:25 发布 1316 收藏 3

分类专栏： CTF WP

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/ITmincherry/article/details/104510494

版权

CTF WP 专栏收录该内容

24 篇文章 1 订阅

订阅专栏

> 前言：太久没刷题了，今天心血来潮一口气刷完了Bugku的加密（感觉拖了好久，拖延症晚期），下为wp。

rsa

```
N : 46065781388428960989637205658554417248531811702624626389974432923749270182062721955600778820059011913617389598900138215153600685382332638289236314360431451868638878600298924880081486124859507532627709964533869497709745916853089877600729369572810197606942397169652423775522718706141820284991147912479399072259 7
e : 35461110244130757205657218182792589919834535022875373093108939327546391654445662689424541509610783446577840953237318712531855461472259930179152891621283936812106603554100880826153450058602365276771227162578520428096468800468032830012484968047710530251937737009257810782711682139182621097232037761496754782761 9

enc : 38230991316229399651823567590692301060044620412191737764632384680546256228451518238842965221394711848337832459443844446889468362154188214840736744657885858943810177675871991111466653158257191139605699916347308294995664530280816850482740530602254559123759121106338359220242637775919026933563326069449424391192
```

可以看到e非常大，e非常大的时候可以使用wiener attack的方法进行破解，工具RsaCtfTool集成了wiener attack的方法，可以用RsaCtfTool计算私钥。具体代码如下：

```
root@kali:~/RsaCtfTool# python RsaCtfTool.py --createpub -n 460657813884289609896372056585544172485318117026246263899744329237492701820627219556007788200590119136173895989001382151536006853823326382892363143604314518686388786002989248800814861248595075326277099645338694977097459168530898776007293695728101976069423971696524237755227187061418202849911479124793990722597 -e 3546111024413075720565721818279258991983453502287537309310893932754639165444566268942454150961078344657784095323731871253185546147225993017915289162128393681210660355410088082615345005860236527677122716257852042809646788004680328300124849680477105302519377370092578107827116821391826210972320377614967547827619 > pub_key.pem
root@kali:~/RsaCtfTool# python RsaCtfTool.py --publickey pub_key.pem --private > pub.key
root@kali:~/RsaCtfTool# python RsaCtfTool.py --key pub.key --dumpkey
[*] n: 460657813884289609896372056585544172485318117026246263899744329237492701820627219556007788200590119136173895989001382151536006853823326382892363143604314518686388786002989248800814861248595075326277099645338694977097459168530898776007293695728101976069423971696524237755227187061418202849911479124793990722597
[*] e: 3546111024413075720565721818279258991983453502287537309310893932754639165444566268942454150961078344657784095323731871253185546147225993017915289162128393681210660355410088082615345005860236527677122716257852042809646788004680328300124849680477105302519377370092578107827116821391826210972320377614967547827619
[*] d: 826466797229427501729333977237178332216882214947197683422108239340936369189 5
[*] p: 1599184697099321332207262690156074993268632576640340486402334181073531924906637091609064092621907936884551044403140032222914777168296113242048189736284319 9
[*] q: 2880579177126025948685690272902043868667035444129624714820786283606465784973534361820709816390178728736856976847252134463556733429935676008050745464020700 3
```

使用p,q,e解密密文，即可得到`flag{Wien3r_4tt@ck_1s_3AsY}`

```
#coding:utf-8
from libnum import n2s,s2n
import base64
def gcd(a, b):   #求最大公约数
 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=159918469709932133220726269015607499326863257664034048640233418107353192490663709160906409262190793688455104440314003222291477716829611324204818973628431 99
 q=288057917712602594868569027290204386866703544412962471482078628360646578497353436182070981639017872873685697684725213446355673342993567600805074546402070 03
 e = 354611102441307572056572181827925899198345350228753730931089393275463916544456626894245415096107834465778409532373187125318554614722599301791528916212839368121066035541008808261534500586023652767712271625785204280964688004680328300124849680477105302519377370092578107827116821391826210972320377614967547827619
 # tmp = base64.b64decode("qzogS7X8M3ZOpkUhJJcbukaRduLyqHAPblmabaYSm9iatuulrHcEpBmil7V40N7gbsQXwYx5EBH5r5V2HRcEIOXjgfk5vpGLjPVxBLyXh2DajHPX6KvbFpQ8jNpCQbUNq8Hst00yDSO/6ri9dk6bk7+uyuN0b2K1bNG5St6sCQ4qYEA3xJbsHFvMqtvUdhMiqO7tNCUVTKZdN7iFvSJqK2IHosIf7FqO24zkHZpHi31sYU7pcgYEaGkVaKs8pjq6nbnffr4URfoexZHeQtq5UAkr95zD6WgvGcxaTDKafFntboX9GR9VUZnHePiio7nJ3msfue5rkIbISjmGCAlj+w==")
 d = modinv(e, (p - 1) * (q - 1))
 # c=s2n(tmp)
 c = 382309913162293996518235675906923010600446204121917377646323846805462562284515182388429652213947118483378324594438444468894683621541882148407367446578858589438101776758719911114666531582571911396056999163473082949956645302808168504827405306022545591237591211063383592202426377759190269335633260694494243911 92
 n = p*q
 m=pow(c,d,n)
 print (n2s(m))
```

散乱的密文

```
1f5{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

2	1	6	5	3	4

f	d	@	@	1	}

24个字符分为四组，按照 216534 的顺序进行读取，然后按照123456的顺序排序后组成序列，之后再进行对于栅栏加密的解密，4个字符为一组，写个脚本：

```
str_ciphertext = "lf5{ag024c483549d7fd@@1}"
str_temp = ""
key = "216534"
str_plaintext = ""
k = 1
while k <= 6:
    for j, elem in enumerate(key):  # j用来控制密文的下标，k控制顺序是123456
        if k == int(elem):
            str_temp += str_ciphertext[j] + str_ciphertext[j + 6] + str_ciphertext[j + 12] + str_ciphertext[j +
18]
            k += 1
            break
print(str_temp)  # 栅栏密码，4个字符为一组，进行抽取，之后可以得到明文
for i in range(0, 4):
    str_plaintext += str_temp[i] + str_temp[i + 4] + str_temp[i + 8] + str_temp[i + 12] + str_temp[i + 16] + str
_temp[i + 20]
print(str_plaintext)
```

得到flag{52048c453d794df1}@@，注意提交的时候去掉@@

凯撒部长的奖励

> 就在8月，超师傅出色地完成了上级的特遣任务，凯撒部长准备给超师傅一份特殊的奖励，兴高采烈的超师傅却只收到一长串莫名的密文，超师傅看到英语字串便满脸黑线，帮他拿到这份价值不菲的奖励吧。 密文：MSW{byly_Cm_slol_lYqUlx_yhdls_Cn_Wuymul_il_wuff_bcg_pCwnll_cm_u_Yrwyffyhn_guh_cz_sio_quhn_ni_ayn_bcm_chzilguncihm_sio_wuh_dich_om}
> 题目来源：第七季极客大挑战

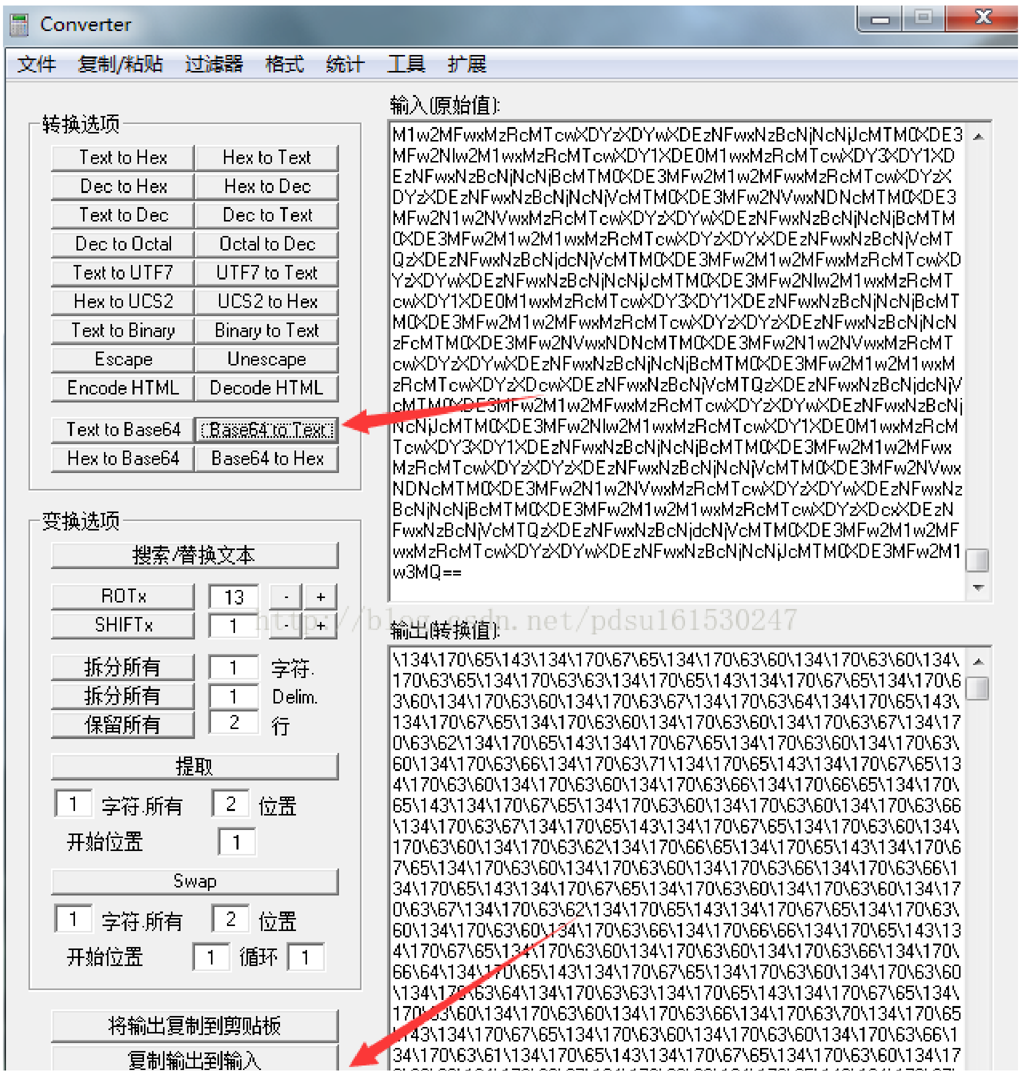
凯撒密码解密：

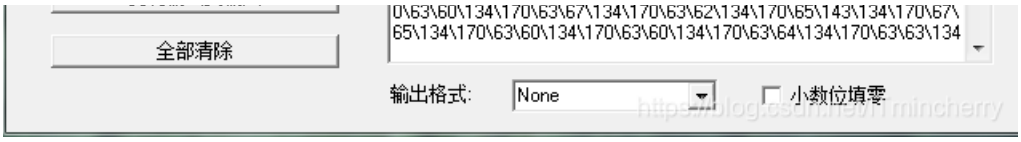
注意本题来源"极客大挑战"，因此其答案格式是"SYC{}"。

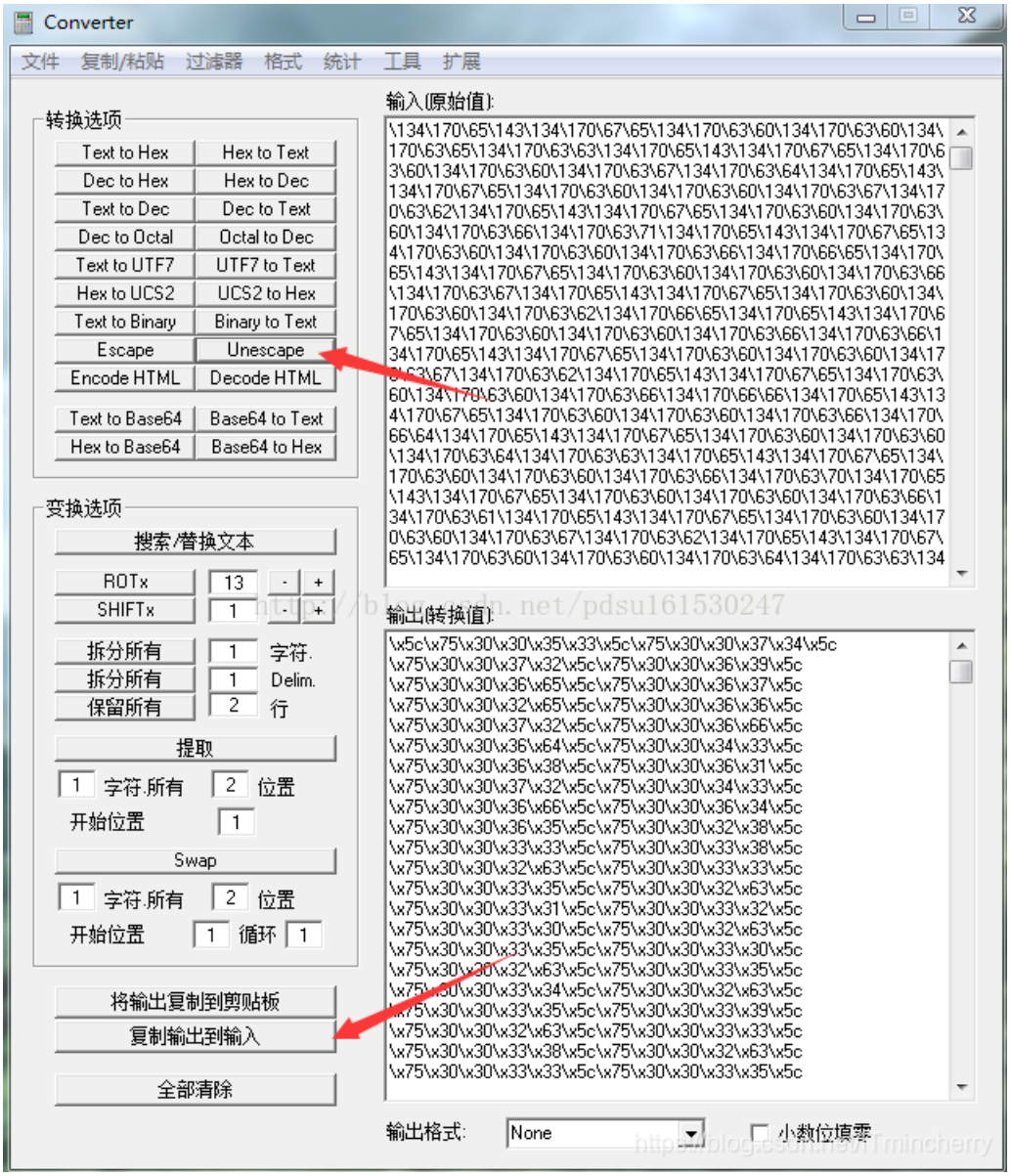
SYC{here_Is_yOur_rEwArd_enjOy_It_Caesar_or_call_him_vIctOr_is_a_Excellent_man_if_you_want_to_get_his_informations_you_can_join_us}

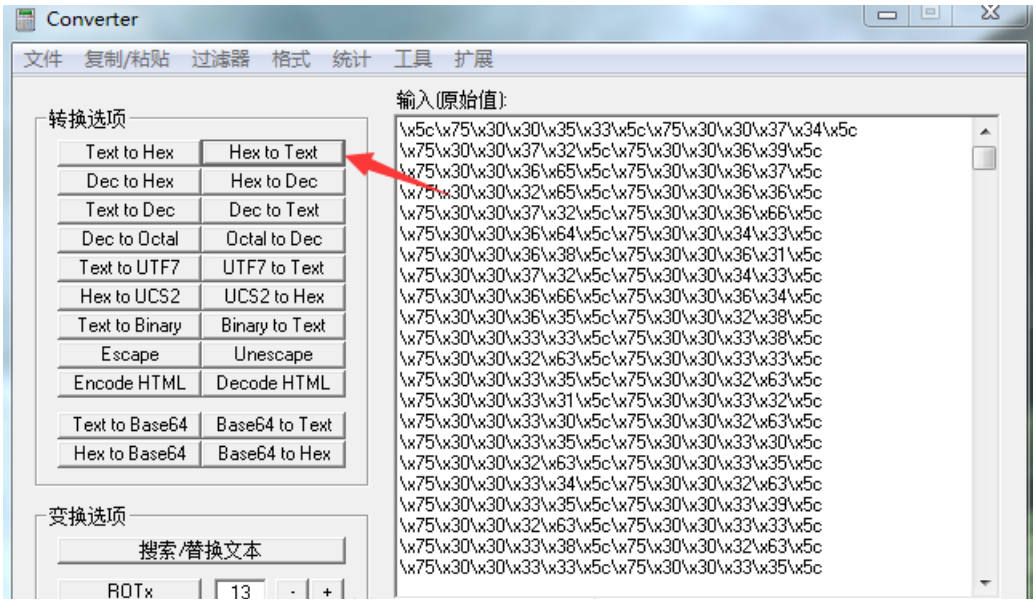
一段Base64

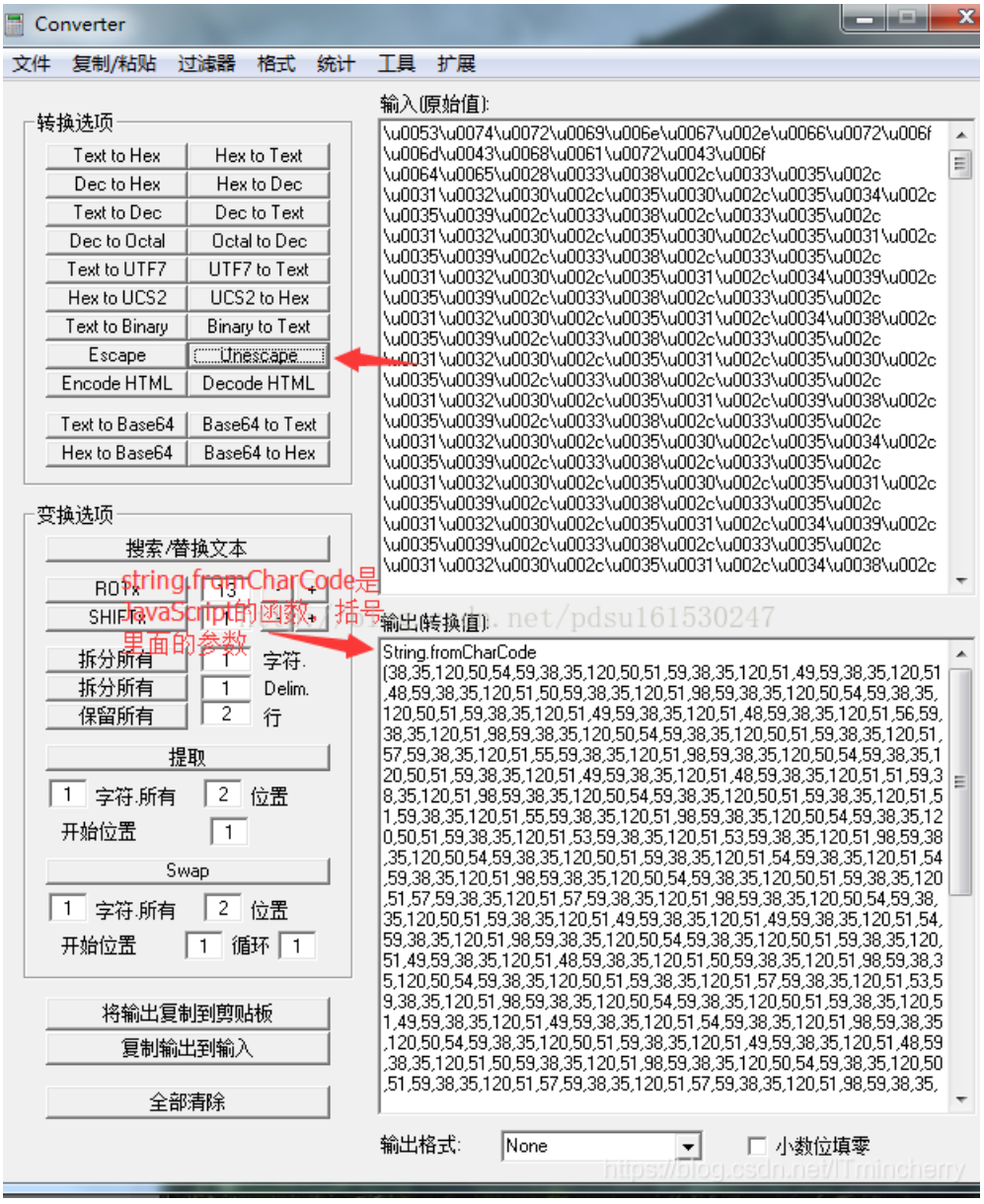
将base64编码复制粘贴到Converter这个软件里面。




全部清除
输出格式:
None
小数位填零

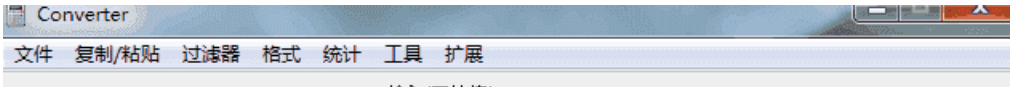

Converter
文件 复制/粘贴 过滤器 格式 统计 工具 扩展
转换选项
Text to Hex
Hex to Text
Dec to Hex
Hex to Dec
Text to Dec
Dec to Text
Dec to Octal
Octal to Dec
Text to UTF7
UTF7 to Text
Hex to UCS2
UCS2 to Hex
Text to Binary
Binary to Text
Escape
Unescape
Encode HTML
Decode HTML
Text to Base64
Base64 to Text
Hex to Base64
Base64 to Hex
变换选项
搜索/替换文本
ROTx
13
SHIFTx
1
拆分所有
1
字符.
拆分所有
1
Delim.
保留所有
2
行
提取
1
字符.所有
2
位置
开始位置
1
Swap
1
字符.所有
2
位置
开始位置
1
循环
1
将输出复制到剪贴板
复制输出到输入
全部清除
输入(原始值):
输出(转换值):
输出格式:
None
小数位填零

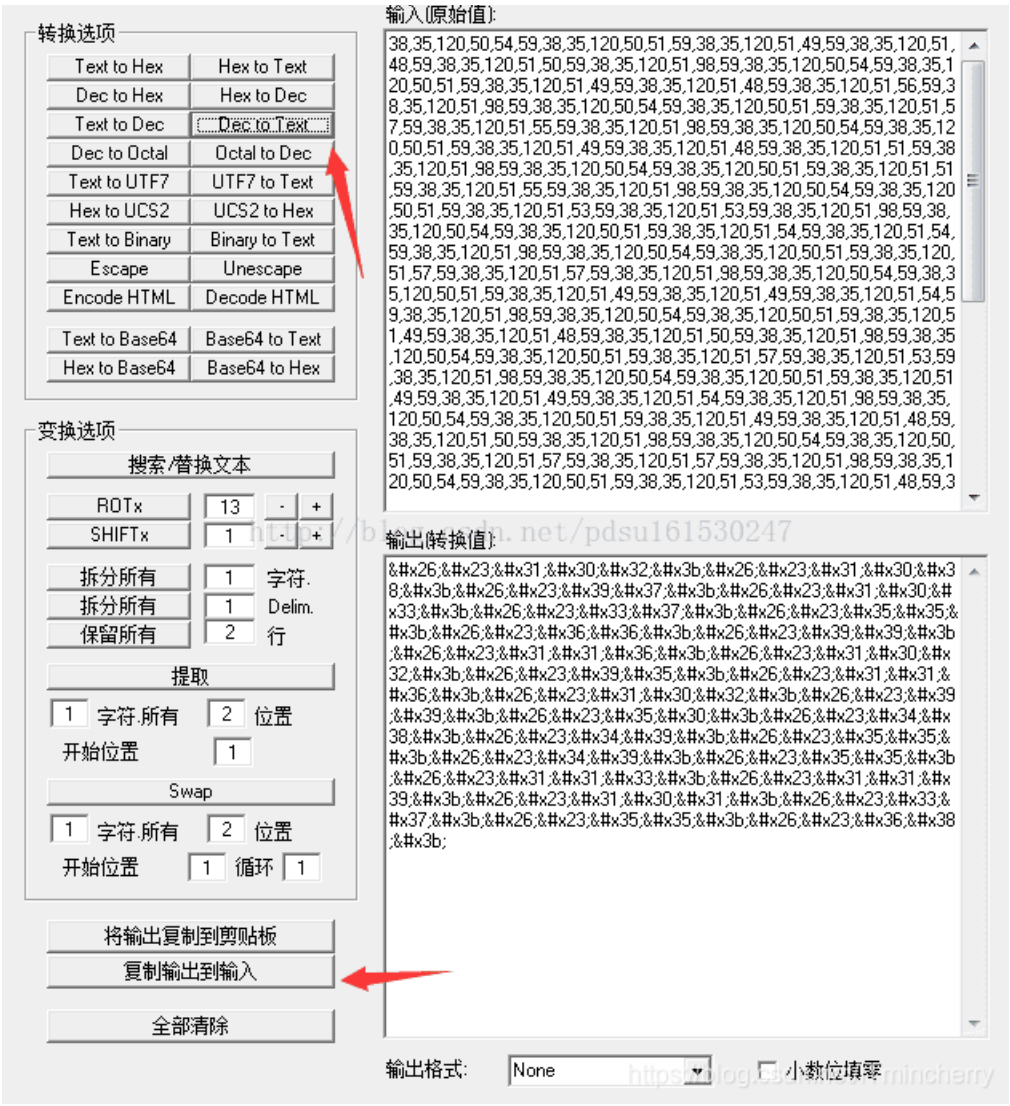

Converter
文件 复制/粘贴 过滤器 格式 统计 工具 扩展
转换选项
Text to Hex
Hex to Text
Dec to Hex
Hex to Dec
Text to Dec
Dec to Text
Dec to Octal
Octal to Dec
Text to UTF7
UTF7 to Text
Hex to UCS2
UCS2 to Hex
Text to Binary
Binary to Text
Escape
Unescape
Encode HTML
Decode HTML
Text to Base64
Base64 to Text
Hex to Base64
Base64 to Hex
变换选项
搜索/替换文本
ROTx
13
输入(原始值):

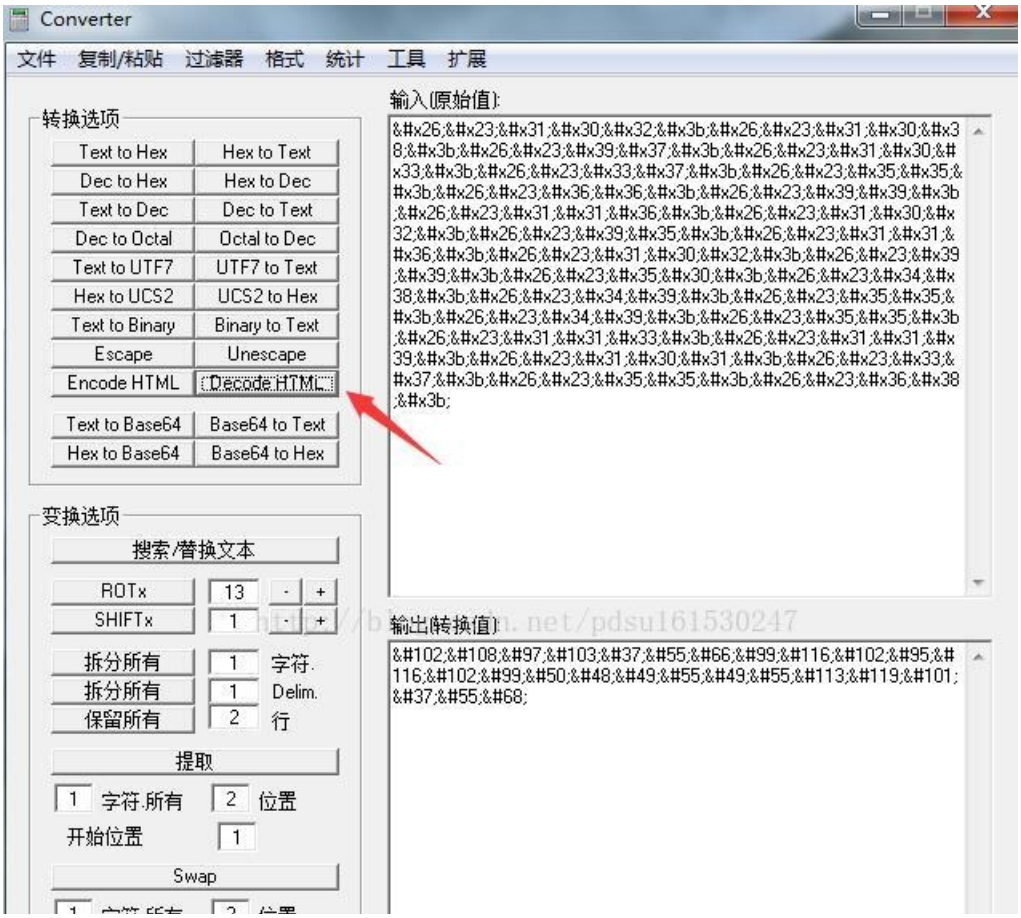


string.fromCharCode是JavaScript的函数，括号里面的参数

把输出中的参数复制到输入，




转换选项
Text to Hex
Hex to Text
Dec to Hex
Hex to Dec
Text to Dec
Dec to Text
Dec to Octal
Octal to Dec
Text to UTF7
UTF7 to Text
Hex to UCS2
UCS2 to Hex
Text to Binary
Binary to Text
Escape
Unescape
Encode HTML
Decode HTML
Text to Base64
Base64 to Text
Hex to Base64
Base64 to Hex
变换选项
搜索/替换文本
ROTx
13
SHIFTx
1
拆分所有
字符.
Delim.
保留所有
行
提取
字符.所有
位置
开始位置
Swap
循环
将输出复制到剪贴板
复制输出到输入
全部清除
输入(原始值):
输出(转换值):
输出格式:
None
小数位填零

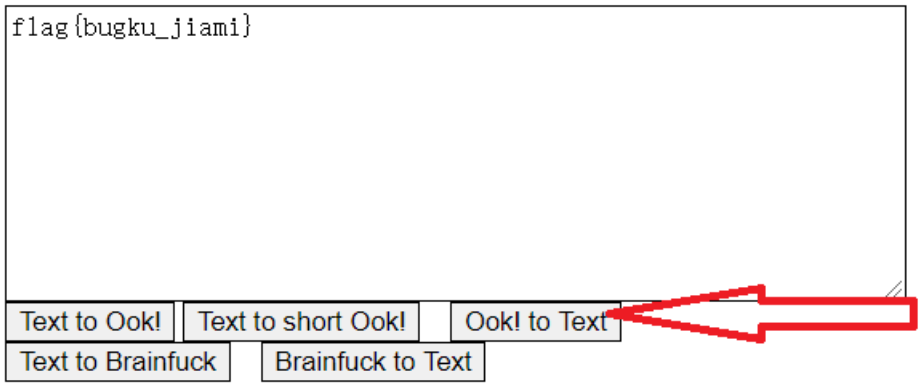

Converter
文件
复制/粘贴
过滤器
格式
统计
工具
扩展
转换选项
Text to Hex
Hex to Text
Dec to Hex
Hex to Dec
Text to Dec
Dec to Text
Dec to Octal
Octal to Dec
Text to UTF7
UTF7 to Text
Hex to UCS2
UCS2 to Hex
Text to Binary
Binary to Text
Escape
Unescape
Encode HTML
Decode HTML
Text to Base64
Base64 to Text
Hex to Base64
Base64 to Hex
变换选项
搜索/替换文本
ROTx
13
SHIFTx
1
拆分所有
字符.
Delim.
保留所有
行
提取
字符.所有
位置
开始位置
Swap
输入(原始值):
输出(转换值):

%7B和%7D是url编码，解出来就是{}

.!?

```
..... ..... ..... ..... !?!!. ?.... ..... ..... ..... .?.?! .?... .!...
..... ..... !.?.. ..... !?!!. ?!!!! !!?.? !.?!! !!!.. ..... ..... .!.?.
..... ...!? !!.?. ..... ..?.? !.?.. ..... .!.?. ..... ..... !?!!. ?!!!!
!!!!! !?.?! .?!.? ..... ....! ?!!.? ..... ...?. ?!.?. ..... !.?.. .....
!?!!. ?!!!! !!?.? !.?!! !!!!! !!!!. ..... ...!. ?.... ...!? !!.?. .....
?.?!. ?..!. ?.... ..... !?!!. ?!!!! !!!!? .?!.? !!!!! !!!!! !!!.? .....
..!?! !.?.. ....? .?!.? ....! .!!!. !!!!! !!!!! !!!!! !!.?. ..... .!?!!
.?... ...?. ?!.?. ..... !.!!! !!!!! !.?.. ..... ..!?! !.?.. ..... .?.?!
.?... ..... !.?.
```

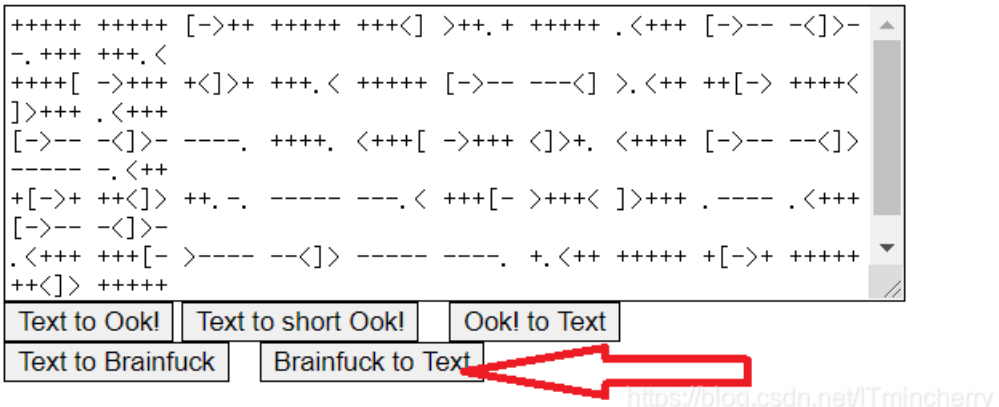
bugku工具解密一下就好了，附上链接：https://tool.bugku.com/brainfuck/



+[]-

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

同样还是bugku工具，



奇怪的密码

> 突然天上一道雷电 gndk€rlqhmtkwwp}z

脑洞！！！
g n d k四个字符的ASCII码是103 110 100 107，
f l a g四个字符的ASCII码是102 108 97 103
附上脚本：

```
secret='gndk€rlqhmtkwwp}z'
temp=1
for i in secret:
    i=ord(i)-temp
    print(chr(i),end="")
    temp+=1
```

得到flag₨lei_ci_jiami，其中Pts是bug，删掉就好了。

托马斯.杰斐逊

> 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：HCBTSXWCRQGLES
> flag格式 flag{你解密的内容}

看了大佬的wp，是**转轮加密**。

> 杰弗逊在1795年发明了一种加密装置叫做杰弗逊圆盘（Jefferson disk ），或叫杰弗逊转轮加密器（Jefferson wheelcipher ）。
> 这个装置有36片同样大小的木制转轮，套在一根铁杆上。每片转轮的圆周边缘上刻有乱序的26个英文字母其使用方法是：进行秘密通信的双方必须各自拥有完全一样转轮加密器。当一方要把一段文字（不超过36字）秘密通知身处异地的对方时，只需转动加密器上的各片转轮，使这段文字正好出现在同一行上，这时转轮上排列的其他25行都是无意义的乱码；再把其中任意一行的乱码抄下来交给信使。信使并不知道这段乱码文字的意义，只负责把它送交对方。对方收到乱码信后，只需拿出自己保存的同样的装置，转动上面各片转轮，让其中一行的排列和这段乱码同处在一行上，然后再查看其他25行上的内容，其中必然有一行显示出加密者要传达的信息，而其他行显示的都是乱码。

首先要根据密钥把14行字符串调整顺序，第一行是原来的第二行，第二行是原来的第五行，以此类推，得到

```
KPBELNACZDTRXMJQOYHGVSFUWI
IHFRLABEUOTSGJVDKCPMNZQWXY
ZWAXJGDLUBVIQHKYPNTCRMOSFE
BDMAIZVRNSJUWFHTEQGYXPLOCK
AMKGHIWPNYCJBFZDRUSLOQXVET
RPLNDVHGFCUKTEBSXQYIZMJWAO
QWATDSRFHENYVUBMCOIKZGJXPL
GWTHSPYBXIZULVKMRAFDCEONJQ
NOZUTWDCVRJLXKISEFAPMYGHBQ
XPHKZGJTDSENYVUBMLAOIRFCQW
WABMCXPLTDSRJQZGOIKFHENYVU
BMCSRFHLTDENQWAOXPYVUIKZGJ
XPLTDAOIKFZGHENYSRUBMCQWVJ
TDSWAYXPLVUBOIKZGJRFHENMCQ
```

接着，我们需要对密文调整每行字符串，
密文第一个字符H，第一行则变为：HGVSFUWIKPBELNACZDTRXMJQOY
密文第二个字符C，第二行则变为：CPMNZQWXYIHFRLABEUOTSGJVDK
以此类推…

```
HGVSFUWIKPBELNACZDTRXMJQOY
CPMNZQWXYIHFRLABEUOTSGJVDK
BVIQHKYPNTCRMOSFEZWAXJGDLU
TEQGYXPLOCKBDMAIZVRNSJUWFH
SLOQXVETAMKGHIWPNYCJBFZDRU
XQYIZMJWAORPLNDVHGFCUKTEBS
WATDSRFHENYVUBMCOIKZGJXPLQ
CEONJQGWTHSPYBXIZULVKMRAFD
RJLXKISEFAPMYGHBQNOZUTWDCV
QWXPHKZGJTDSENYVUBMLAOIRFC
GOIKFHENYVUWABMCXPLTDSRJQZ
LTDENQWAOXPYVUIKZGJBMCSRFH
ENYSRUBMCQWVJXPLTDAOIKFZGH
SWAYXPLVUBOIKZGJRFHENMCQTD
```

观察一下，倒数第六列就是flag了。改成小写提交flag{xsxsbugkuadmin}

zip伪加密

下载zip，用winHex打开，找到第二个PK，将09改成00（奇数表示加密，偶数表示未加密）

Offset	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
00000000	50	4B	03	04	14	00	09	00	08	00	50	A3	A5	4A	21	38	PK.....P£¥J!8
00000016	76	65	19	00	00	00	17	00	00	00	08	00	00	00	66	6C	ve.....fl
00000032	61	67	2E	74	78	74	4B	CB	49	4C	AF	76	4C	C9	35	F4	ag.txtKËIL¯vLÉ5ô
00000048	D3	75	32	72	D7	CD	0E	D5	0D	8E	F2	0C	A8	05	00	50	Óu2r×Í.Õ.ò.¨..P
00000064	4B	01	02	1F	00	14	00	09	00	08	00	50	A3	A5	4A	21	K.....P£¥J!
00000080	38	76	65	19	00	00	00	17	00	00	00	08	00	24	00	00	8ve.....$..
00000096	00	00	00	00	00	20	00	00	00	00	00	00	00	66	6C	61	....fla
00000112	67	2E	74	78	74	0A	00	20	00	00	00	00	00	01	00	18	g.txt.. .....
00000128	00	0F	F5	04	D5	9A	C5	D2	01	46	1F	CB	8A	9A	C5	D2	..õ.ÕÅÒ.F.ËÅÒ
00000144	01	46	1F	CB	8A	9A	C5	D2	01	50	4B	05	06	00	00	00	.F.ËÅÒ.PK.....
00000160	00	01	00	01	00	5A	00	00	00	3F	00	00	00	00	00		.....Z...?.....

https://blog.csdn.net/Hmincherry

告诉你个秘密(ISCCCTF)

> 636A56355279427363446C4A49454A7154534230526D6843
> 56445A31614342354E326C4B4946467A5769426961453067

1、查看给的字符串，字符只有A-F，而且上下两两结合对应的十进制数小于128；
2、将它们转化成ASCII对应的字母，得到cjV5RyBscDlJlEJqTSB0RmhCVDZ1aCB5N2lKlFFzWiBiaE0g
3、2中得到的字符串，像是经过了base64加密，解密得到`r5yG lp9I BjM tFhBT6uh y7iJ QsZ bhM`
4、看大佬wp，脑洞太大了吧！！！看键盘r5yG包围着T，以此类推得到TONGYUAN，直接提交就OK了。

下面是脚本：

```
#coding:utf-8
import base64
strs="636A5635527942736344C4A49454A7154534230526D684356445A31614342354E326C4B4946467A5769426961453067"
i = 0
strs_len = len(strs)
chs=[]
while(i < strs_len):
    chs.append(chr(int(strs[i:i+2],16)))
    i = i + 2
print("".join(chs))
print(base64.b64decode("".join(chs)))
```

这不是md5

> 666c61677b6165373335383762613536626165663 57d

16进制转字符串就OK了，转换地址：https://www.sojson.com/hexadecimal.html

贝斯家族

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

看了大佬wp，是base91编码，国内没有免费的base91在线解密网站，大佬fq找到了https://www.dcode.fr/base-91-encoding，不得不说tql。

富强民主

> 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

核心价值观编码https://aliyunvi.com/cvencode

python(N1CTF)

```
#challenge.py

from N1ES import N1ES
import base64
key = "wxy191iss00000000000cute"
n1es = N1ES(key)
flag = "N1CTF{****************************************}"
cipher = n1es.encrypt(flag)
print base64.b64encode(cipher)  # HRlgC2ReHW1/WRk2DikfNBo1dl1XZBJrRR9qECMNOjNHDktBJSxcI1hZIz07YjVx
```

```
#N1ES.py
# -*- coding: utf-8 -*-
def round_add(a, b):
    f = lambda x, y: x + y - 2 * (x & y)
    res = ''
    for i in range(len(a)):
        res += chr(f(ord(a[i]), ord(b[i])))
    return res

def permutate(table, block):
```

```
 return list(map(lambda x: block[x], table))

def string_to_bits(data):
    data = [ord(c) for c in data]
    l = len(data) * 8
    result = [0] * l
    pos = 0
    for ch in data:
        for i in range(0,8):
            result[(pos<<3)+i] = (ch>>i) & 1
        pos += 1
    return result

s_box = [54, 132, 138, 83, 16, 73, 187, 84, 146, 30, 95, 21, 148, 63, 65, 189, 188, 151, 72, 161, 116, 63, 161,
91, 37, 24, 126, 107, 87, 30, 117, 185, 98, 90, 0, 42, 140, 70, 86, 0, 42, 150, 54, 22, 144, 153, 36, 90, 149, 5
4, 156, 8, 59, 40, 110, 56,1, 84, 103, 22, 65, 17, 190, 41, 99, 151, 119, 124, 68, 17, 166, 125, 95, 65, 105, 13
3, 49, 19, 138, 29, 110, 7, 81, 134, 70, 87, 180, 78, 175, 108, 26, 121, 74, 29, 68, 162, 142, 177, 143, 86, 129
, 101, 117, 41, 57, 34, 177, 103, 61, 135, 191, 74, 69, 147, 90, 49, 135, 124, 106, 19, 89, 38, 21, 41, 17, 155,
 83, 38, 159, 179, 19, 157, 68, 105, 151, 166, 171, 122, 179, 114, 52, 183, 89, 107, 113, 65, 161, 141, 18, 121,
 95, 4, 95, 101, 81, 156, 17, 190, 38, 84, 9, 171, 180, 59, 45, 15, 34, 89, 75, 164, 190, 140, 6, 41, 188, 77, 1
65, 105, 5, 107, 31, 183, 107, 141, 66, 63, 10, 9, 125, 50, 2, 153, 156, 162, 186, 76, 158, 153, 117, 9, 77, 156
, 11, 145, 12, 169, 52, 57, 161, 7, 158, 110, 191, 43, 82, 186, 49, 102, 166, 31, 41, 5, 189, 27]

def generate(o):
    k = permutate(s_box,o)
    b = []
    for i in range(0, len(k), 7):
        b.append(k[i:i+7] + [1])
    c = []
    for i in range(32):
  pos = 0
  x = 0
  for j in b[i]:
   x += (j<<pos)
   pos += 1
  c.append((0x10001**x) % (0x7f))
    return c



class N1ES:
    def __init__(self, key):
        if (len(key) != 24 or isinstance(key, bytes) == False ):
            raise Exception("key must be 24 bytes long")
        self.key = key
        self.gen_subkey()

    def gen_subkey(self):
        o = string_to_bits(self.key)
        k = []
        for i in range(8):
         o = generate(o)
         k.extend(o)
         o = string_to_bits([chr(c) for c in o[0:24]])
        self.Kn = []
        for i in range(32):
            self.Kn.append(map(chr, k[i * 8: i * 8 + 8]))
        return

    def encrypt(self, plaintext):
```

```
    def encrypt(self, plaintext):
        if (len(plaintext) % 16 != 0 or isinstance(plaintext, bytes) == False):
            raise Exception("plaintext must be a multiple of 16 in length")
        res = ''
        for i in range(len(plaintext) / 16):
            block = plaintext[i * 16:(i + 1) * 16]
            L = block[:8]
            R = block[8:]
            for round_cnt in range(32):
                L, R = R, (round_add(L, self.Kn[round_cnt]))
            L, R = R, L
            res += L + R
        return res
```

猜测题目是让我们根据给的加密方式，写出解密脚本。难度较大，参考大佬wp：
注意运行环境是python2，可以在菜鸟工具在线运行。

```
# -*- coding: utf-8 -*-
import base64
def round_add(a,b):
    f = lambda x,y: x + y - 2 * (x & y)
    res = ''
    for i in range(len(a)):
        res += chr(f(ord(a[i]),ord(b[i])))
    return res

def permutate(table,block):
    return list(map(lambda x: block[x], table))

def string_to_bits(data):
    data = [ord(c) for c in data]
    l = len(data)*8
    result = [0] * l
    pos = 0
    for ch in data:
        for i in range(0,8):
            result[(pos<<3)+i] = (ch>>i) & 1
        pos += 1
    return result

s_box = [54, 132, 138, 83, 16, 73, 187, 84, 146, 30, 95, 21, 148, 63, 65, 189, 188, 151, 72, 161, 116, 63, 161,
91, 37, 24, 126, 107, 87, 30, 117, 185, 98, 90, 0, 42, 140, 70, 86, 0, 42, 150, 54, 22, 144, 153, 36, 90, 149, 5
4, 156, 8, 59, 40, 110, 56,1, 84, 103, 22, 65, 17, 190, 41, 99, 151, 119, 124, 68, 17, 166, 125, 95, 65, 105, 13
3, 49, 19, 138, 29, 110, 7, 81, 134, 70, 87, 180, 78, 175, 108, 26, 121, 74, 29, 68, 162, 142, 177, 143, 86, 129
, 101, 117, 41, 57, 34, 177, 103, 61, 135, 191, 74, 69, 147, 90, 49, 135, 124, 106, 19, 89, 38, 21, 41, 17, 155,
 83, 38, 159, 179, 19, 157, 68, 105, 151, 166, 171, 122, 179, 114, 52, 183, 89, 107, 113, 65, 161, 141, 18, 121,
 95, 4, 95, 101, 81, 156, 17, 190, 38, 84, 9, 171, 180, 59, 45, 15, 34, 89, 75, 164, 190, 140, 6, 41, 188, 77, 1
65, 105, 5, 107, 31, 183, 107, 141, 66, 63, 10, 9, 125, 50, 2, 153, 156, 162, 186, 76, 158, 153, 117, 9, 77, 156
, 11, 145, 12, 169, 52, 57, 161, 7, 158, 110, 191, 43, 82, 186, 49, 102, 166, 31, 41, 5, 189, 27]

def generate(o):
    k = permutate(s_box,o)
    b = []
    for i in range(0,len(k),7):
        b.append(k[i:i+7]+[1])
    c = []
    for i in range(32):
        pos = 0
        x = 0
        for j in b[i]:
```

```
            x += (j<<pos)
            pos += 1
        c.append((0x10001**x) % (0x7f))
    return c

class N1ES:
    def __init__(self,key):
        if (len(key) != 24 or isinstance(key,bytes) == False):
            raise Exception("key must be 24 bytes long")
        self.key = key
        self.gen_subkey()

    def gen_subkey(self):
        o = string_to_bits(self.key)
        k = []
        for i in range(8):
            o = generate(o)
            k.extend(o)
            o = string_to_bits([chr(c) for c in o[0:24]])
        self.Kn = []
        for i in range(32):
            self.Kn.append(map(chr,k[i*8: i*8+8]))
        return

    def decrypt(self,plaintext):
        res = ''
        for i in range(len(plaintext)/16):
            block = plaintext[i*16:(i + 1)*16]
            L = block[:8]
            R = block[8:]
            for round_cnt in range(32):
                L,R = R, (round_add(L, self.Kn[31-round_cnt]))
            L,R = R,L
            res += L + R
        return res


key = "wxy191iss00000000000cute"
nles = N1ES(key)
flag = base64.b64decode("HRlgC2ReHW1/WRk2DikfNBo1dl1XZBJrRR9qECMNOjNHDktBJSxcI1hZIz07YjVx")
flag = nles.decrypt(flag)
print flag
```

参考大佬博客：https://www.cnblogs.com/0yst3r-2046/p/12123653.html

进制转换

> 二进制、八进制、十进制、十六进制，你能分的清吗？
> 来源：第七届大学生网络安全技能大赛
> d87 x65 x6c x63 o157 d109 o145 b100000 d116 b1101111 o40 x6b b1100101 b1101100 o141 d105 x62 d101 b1101001 d46 o40 d71 x69 d118 x65 x20 b1111001 o157 b1110101 d32 o141 d32 d102 o154 x61 x67 b100000 o141 d115 b100000 b1100001 d32 x67 o151 x66 d116 b101110 b100000 d32 d102 d108 d97 o147 d123 x31 b1100101 b110100 d98 d102 b111000 d49 b1100001 d54 b110011 x39 o64 o144 o145 d53 x61 b1100010 b1100011 o60 d48 o65 b1100001 x63 b110110 d101 o63 b111001 d97 d51 o70 d55 b1100010 d125 x20 b101110 x20 b1001000 d97 d118 o145 x20 d97 o40 d103 d111 d111 x64 d32 o164 b1101001 x6d o145 x7e

这串字符有b二进制、o八进制、d十进制、x十六进制，先上脚本全部转化为十六进制：

```
s = ["d87", "x65", "x6c", "x63", "o157", "d109", "o145", "b100000", "d116", "b1101111", "o40", "x6b", "b1100101"
,
     "b1101100", "o141", "d105", "x62", "d101", "b1101001", "d46", "o40", "d71", "x69", "d118", "x65", "x20",
     "b1111001", "o157", "b1110101", "d32", "o141", "d32", "d102", "o154", "x61", "x67", "b100000", "o141", "d11
5",
     "b100000", "b1100001", "d32", "x67", "o151", "x66", "d116", "b101110", "b100000", "d32", "d102", "d108", "d
97",
     "o147", "d123", "x31", "b1100101", "b110100", "d98", "d102", "b111000", "d49", "b1100001", "d54", "b110011"
, "x39",
     "o64", "o144", "o145", "d53", "x61", "b1100010", "b1100011", "o60", "d48", "o65", "b1100001", "x63", "b1101
10",
     "d101", "o63", "b111001", "d97", "d51", "o70", "d55", "b1100010", "d125", "x20", "b101110", "x20", "b100100
0",
     "d97", "d118", "o145", "x20", "d97", "o40", "d103", "d111", "d111", "x64", "d32", "o164", "b1101001", "x6d"
,
     "o145", "x7e"]
s1 = ""
t = ""
t1 = ""
for i in s:
    s1 = i
    for j in range(1):
        if s1[0:1] == 'd':
            t = str(hex(int(s1[1:])))
            t = t[2:] + " "
            t1 = t1 + t
        if s1[0:1] == 'x':
            t = s1[1:] + " "
            t1 = t1 + t
        if s1[0:1] == 'o':
            t = str(hex(int(s1[1:], 8)))
            t = t[2:] + " "
            t1 = t1 + t
        if s1[0:1] == 'b':
            t = str(hex(int(s1[1:], 2)))
            t = t[2:] + " "
            t1 = t1 + t
print (t1)
```

得到一串16进制数，然后转字符，https://www.bejson.com/convert/ox2str/

```
57 65 6c 63 6f 6d 65 20 74 6f 20 6b 65 6c 61 69 62 65 69 2e 20 47 69 76 65 20 79 6f 75 20 61 20 66 6c 61 67 20 6
1 73 20 61 20 67 69 66 74 2e 20 20 66 6c 61 67 7b 31 65 34 62 66 38 31 61 36 33 39 34 64 65 35 61 62 63 30 30 35
 61 63 36 65 33 39 61 33 38 37 62 7d 20 2e 20 48 61 76 65 20 61 20 67 6f 6f 64 20 74 69 6d 65 7e
```

得到flag{1e4bf81a6394de5abc005ac6e39a387b}

affine

> y = 17x-8 flag{szzyfimhyzd}
> 答案格式：flag{*}
> 来源：第七届山东省大学生网络安全技能大赛

affine是仿射的意思，想到可能是仿射加密，y = 17x-8为加密函数，仿射加密知识参考https://blog.csdn.net/zz_Caleb/article/details/84184283
按照26个英文字母及其顺序进行解密。
下面是解密代码：

```
list = 'abcdefghijklmnopqrstuvwxyz'
newlist = ''

for i in range(len(list)):
    newlist += list[(17 * i - 8) % 26]

y = 'szzyfimhyzd'
x = ''
for j in y:
    x += list[newlist.find(j)]

print('flag{' + x + '}')
```

得到flag{affineshift}

Crack it

> 破解该文件，获得密码，flag格式为：flag{*}
> 来源：第七届山东省大学生网络安全技能大赛

下载下来是shadow文件，放到kali里面查看一下，

```
root@kali:~/文档/ctf# more shadow
root:$6$HRMJoyGA$26FIgg6CU0bGUOfqFB0Qo9AE2LRZxG8N3H.3BK8t49wGlYbkFbxVFtGOZqVIq3qQ
6k0oetDbn2aVzdhuVQ6US.:17770:0:99999:7:::
```

这是linux shadow文件，shadow文件是存放root密码的，具体介绍看这篇博客，用kali里面的john工具爆破，

```
root@kali:~/文档/ctf# john shadow
Created directory: /root/.john
Using default input encoding: UTF-8
Loaded 1 password hash (sha512crypt, crypt(3) $6$ [SHA512 256/256 AVX2 4x])
Cost 1 (iteration count) is 5000 for all loaded hashes
Proceeding with single, rules:Single
Press 'q' or Ctrl-C to abort, almost any other key for status
Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Warning: Only 4 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Warning: Only 2 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Warning: Only 7 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Warning: Only 2 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Almost done: Processing the remaining buffered candidate passwords, if any.
Warning: Only 5 candidates buffered for the current salt, minimum 8 needed for pe
rformance.
Proceeding with wordlist:/usr/share/john/password.lst, rules:Wordlist
hellokitty       (root)
1g 0:00:00:02 DONE 2/3 (2020-02-26 22:11) 0.3460g/s 1774p/s 1774c/s 1774C/s ilove
god..santiago
Use the "--show" option to display all of the cracked passwords reliably
Session completed
```

https://blog.csdn.net/ITmincherry

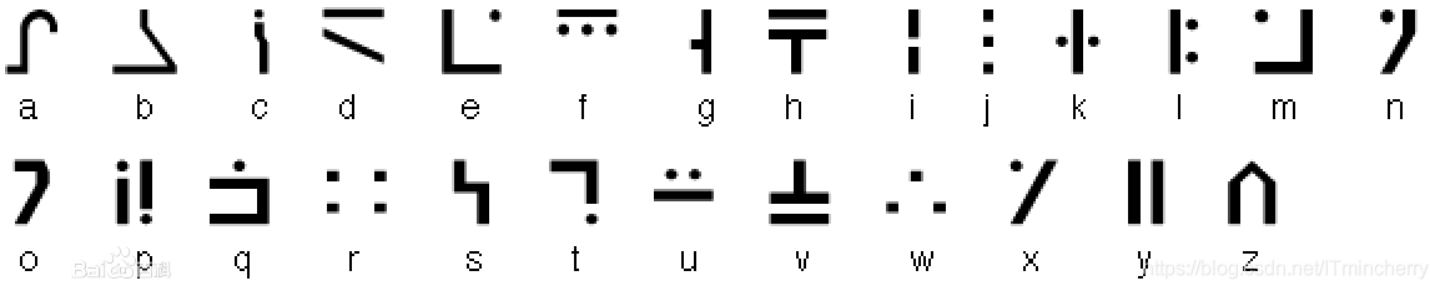
```
root@kali:~/文档/ctf# john --show shadow
root:hellokitty:17770:0:99999:7:::

1 password hash cracked, 0 left
```

得到flag:hellokitty

来自宇宙的信号

> 银河战队出击
> flag格式 flag{字母小写}

这是一种标注银河字母，百度百科介绍



对比得到flag{nopqrst}