HGAME 2021 Week2

MISC

Tools

F5 隐写

隐写密码在文件属性的详细信息的备注

```
!LyJJ9bi&M7E72*JyD
```

```
PS C:\Users\Director\Desktop\F5\F5-steganography-master> java Extract C:\Users\Director\Desktop\2021HGAME\misc_21\tools_
21d9ccfca5a4321d6256038d3e885b6d\Matryoshka.jpg -p '!LyJJ9bi&M7E72*JyD'
Huffman decoding starts
Permutation starts
577536 indices shuffled
Extraction starts
Length of embedded file: 18 bytes
(1, 127, 7) code used
```

得到压缩包的解压密码

```
e@317S*p1A4bIYIs1M
```

Steghide 隐写

隐写密码在文件属性的详细信息的备注

```
A7SL9nHRJXLh@$EbE8
```

```
Suggestion [3,General]: 找不到命令 steghide，但它确实存在于当前位置。默认情况下，Windows PowerShell 不会从当前位置加载命
令。如果信任此命令，请改为键入“.\steghide”。有关详细信息，请参阅 "get-help about_Command_Precedence"。
PS C:\Users\Director\Desktop\steghide\steghide> .\steghide.exe extract -sf C:\Users\Director\Desktop\2021HGAME\misc_21\t
ools_21d9ccfca5a4321d6256038d3e885b6d\F5\01.jpg -p 'A7SL9nHRJXLh@$EbE8'
wrote extracted data to "pwd.txt".
```

得到压缩包的解压密码

```
u0!FO4JUhl5!L55%$&
```

Outguess隐写

隐写密码在文件属性的详细信息的备注

```
z0GFieYAee%gdf0%lF
```

```
misc_21$ outguess -r 02.jpg 02.txt -k 'z0GFieYAee%gdf0%lF'
Reading 02.jpg....
Extracting usable bits:   4930 bits
Steg retrieve: seed: 184, len: 18
```

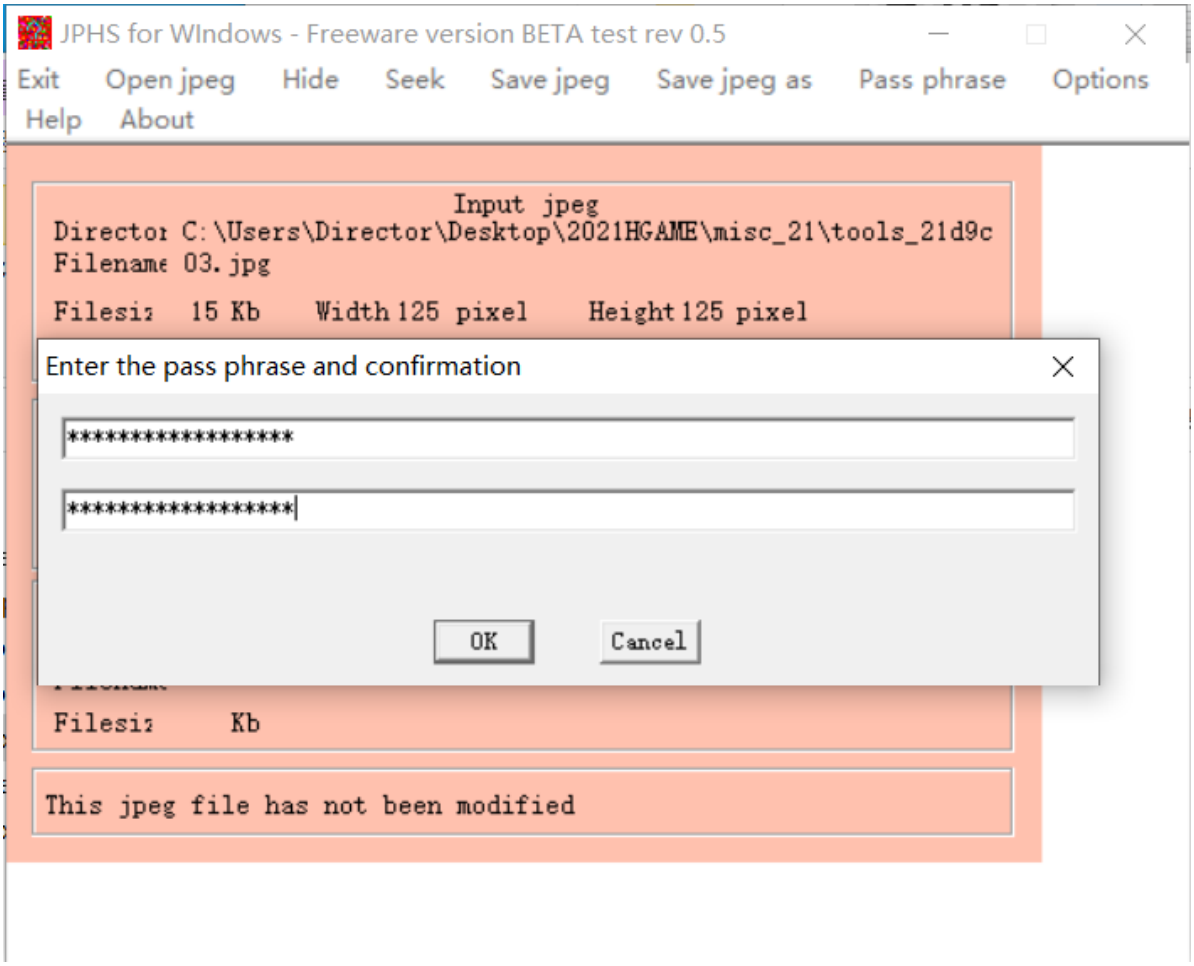
得到压缩包的解压密码

@UjXL93044V5zl2ZKI

JPHS隐写

隐写密码在文件属性的详细信息的备注

```
rFQmRoT5lze@4X4^@O
```



得到压缩包的解压密码

xSRejK1^Z1Cp9M!z@H

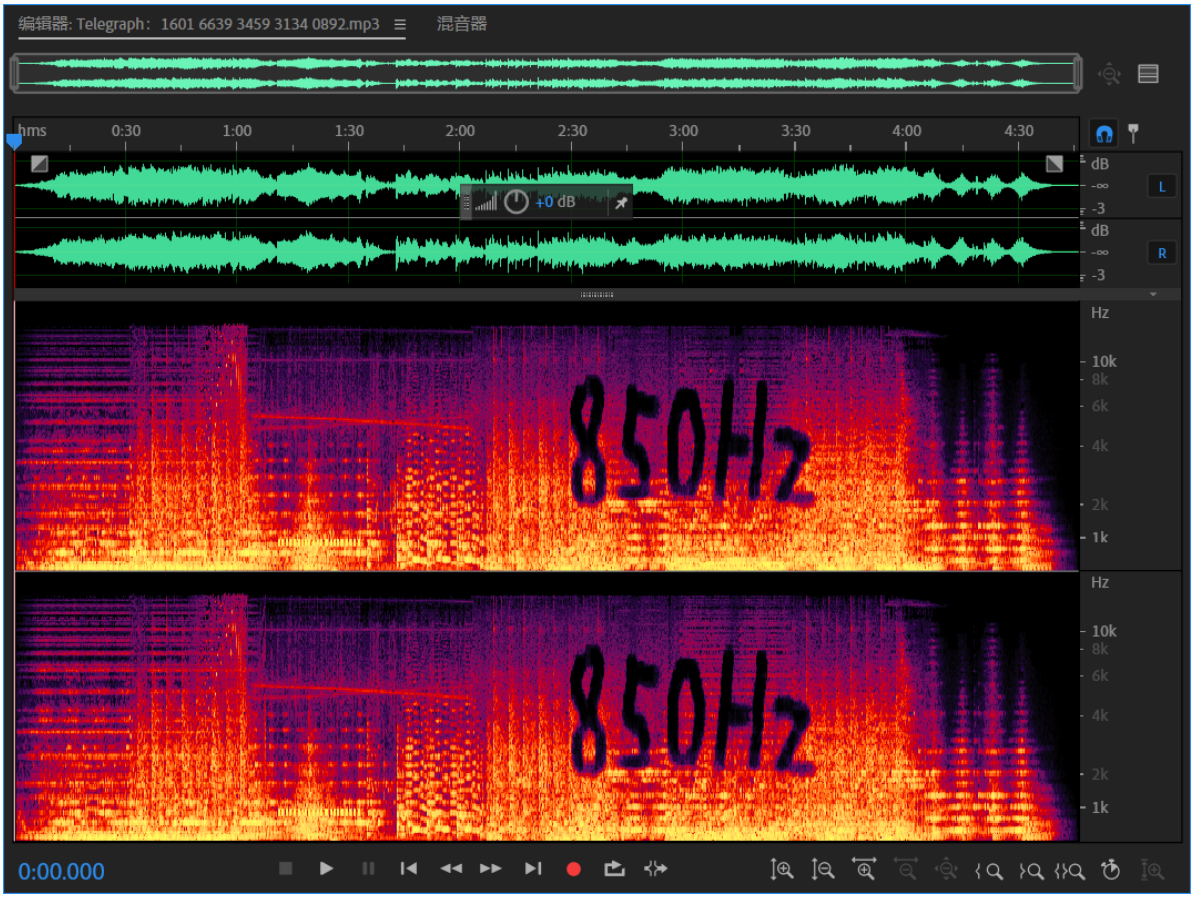
最后将四块二维码拼起来扫码

flag

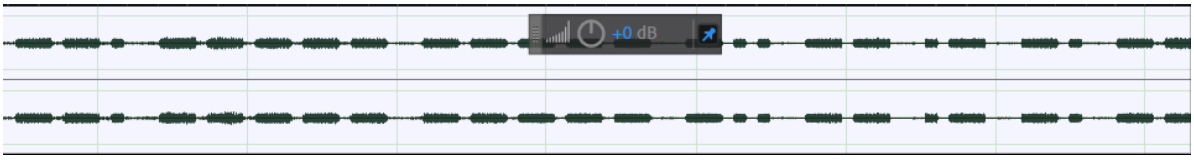
```
hgame{Taowa_is_NOT_gOOd_but_TOO1s_is_Useful}
```

Telegraph：1601 6639 3459 3134 0892

拿到音频文件，AU 打开，在频谱有提示



滤出 850Hz 的信息判断为摩斯密码




```
-.--/---/../.-./..-./.-../.-/--./../.../---.../..../--./-----/-----/-../.../---
--/-./--./-.../..-/-/-./-----/-/..../--./-----/-----/-../--/.-/-./-----/...--/--
--./...--/.----/-----/-.-/..
```

解密

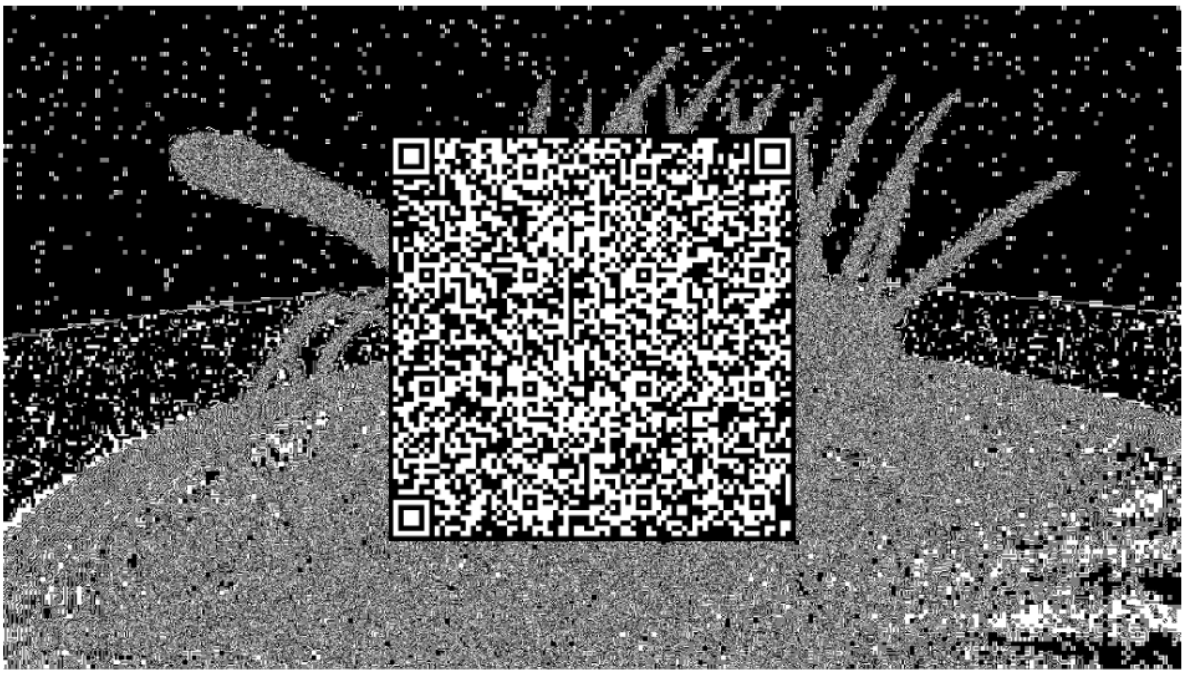
```
YOURFLAGIS:4GOODSONGBUTNOT4GOODMAN039310KI
```

flag

```
hgame{4GOODSONGBUTNOT4GOODMAN039310KI}
```

Hallucigenia

拿到一张图片，用Stegsolve得到二维码



扫码拿到 base64 数据

```
gmBCrkRORUkAAAAA+jrgsWajaqOBeC3IQhCEIQhCKZw1MxTzSlNKnmJpivW9IHVPrTjvkkuI3sP7bWAE
dIHWCbDsGsRkZ9IUJC9AhfZFbpqrmZBtI+ZvptWC/KCPrLOgFeRPOcI2WyqjndfUWlNj+dgWpe1qSTEc
durXzMRAc5EihsEflmIN8RzuguWq61JWRQpSI51/KHHT/6/ztPZJ33SSKbieTa1C5koONbLcf9aYmsVh
7RW6p3SpASnUSb3JuSvpUBKxscbyBjiOpOTq8jcdRsx5/IndXw3VgJV6iO1+6jl4gjVpWouViO6ih9Zm
ybSPkhaqyNUxVXpV5cYU+Xx5sQTfKystDLipmqaMhxIcgvplLqF/LWZzIS5PvwbqOvrSlNHVEYchCEIQ
ISICSZJijwu5OrRQHDyUpaFOy///p6FEDCCDFsuW7YFoVEFESTOBAACLgLOrAAAAAggUAAAAtAAAAFJE
SEkNAAAAChoKDUdOUIk=
```

解码得到的是乱码，但是乱码开头好像是 png 文件的二进制开头

```
f = open('3.txt','wb')
with open('2.txt','rb') as g:
    f.write(g.read()[::-1])
f.close()
```

脚本反转一下数据，得到 png

将图片上下左右翻转

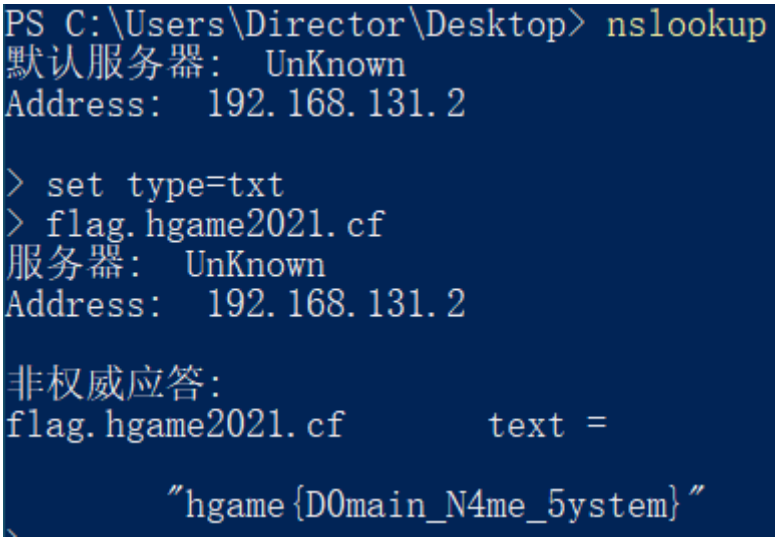
flag

```
hgame{tenchi_souzou_dezain_bu}
```

DNS

wireshark中在导出 HTTP 中看到提示

分组	主机名	内容类型	大小	文件名
77	flag.hgame2021.cf	text/html	186 bytes	\



flag

```
hgame{DOmain_N4me_5ystem}
```

Crypto

signin

```
from libnum import *
from Crypto.Util import number
from secret import FLAG

m = s2n(FLAG)
a = number.getPrime(1024)
p = number.getPrime(1024)

c = a ** p * m % p

print("a = {}".format(a))
print("p = {}".format(p))
print("c = {}".format(c))

# a =
14159823422000956332470397032491190513831048767850409666835790691667114636065952
91938837711127065661145277279716738666794508265581888963197916645062983217610874
73538120263961702829724300785225243215686348146495034373951340373078575478723977
59890540403957345153845948228154918661684022114196350552283989600467 7
```

```
# p =
120745143001485342228286688066218113539764532372807529699634885702123656125148512612668998248974189483480872353177449240892368719585177554625477351848580758754952603345237843942269236224446463745664144053139839533489035502279913759515711150885047415556931436301278062803808219912959682939484142475094684426943

# c =
736217079486249518574168109772532917121266830052752623066013039408539677619990423308721474307986351065202761027104796871331933073178742530631447392511315632654786625474017738921280697994936486122346845348756536175410553849250030401481374799037665162253973265304925764222972020548058566932319304946826096406 46
```

费马小定理：$a^{p-1} \equiv 1 \pmod{p}$

$c = a^p * m \pmod{p} \Rightarrow c = a * m \pmod{p}$

只要找到 a 对于 p 的逆元 d 即 $a * d \equiv 1 \pmod{p}$ //例如 $d = a^{p-2}$

$c * d = a * m * d \pmod{p} \Rightarrow c * d = m \pmod{p}$

Code:

```
from Crypto.Util import number
import gmpy2

a =
141598234220009563324703970324911905138310487678504096668357906916671146360659529193883771112706566114527727971673866679450826558188896319791664506298321761087473538120263961702829724300785225243215686348146495034373951340373078575478723977598905404039573451538459482281549186616840221141963505522839896004677
p =
120745143001485342228286688066218113539764532372807529699634885702123656125148512612668998248974189483480872353177449240892368719585177554625477351848580758754952603345237843942269236224446463745664144053139839533489035502279913759515711150885047415556931436301278062803808219912959682939484142475094684426943
c =
736217079486249518574168109772532917121266830052752623066013039408539677619990423308721474307986351065202761027104796871331933073178742530631447392511315632654786625474017738921280697994936486122346845348756536175410553849250030401481374799037665162253973265304925764222972020548058566932319304946826096406 46

d = int(gmpy2.invert(a, p))

plaintext = c * d % p
print (number.long_to_bytes(plaintext))
```

flag

```
hgame{MOdu1@r_m4th+1s^th3~ba5is-Of=cRyptO!!}
```

gcd or more?

```
from libnum import *
from secret import FLAG

p =
852285650211289018533149345831290834419890452250225412985505704493898396090 19
q =
111614714641364911312915294479850549131835378046002423977989457843071188836271
n = p * q

cipher = pow(s2n(FLAG), 2, n)
print(cipher)

#
766500368283066645619389449101598964164785482664717787314198410720209908147598 48
278060072878304728996168180809072766067444674534459089230549753936235095 39
```

p, q 为素数且均模 4 余 3， e = 2 可以判断为 Rabin 算法

1、运用广义 Euclid 除法，求出整数 s 和 t 使得 sp + tq = 1; //或者说 s 为 p 对于 q 的逆元，t 为 q 对于 q 的逆元

2、计算 $u \equiv c^{(p+1)/4} \pmod{p}$;

3、计算 $v \equiv c^{(q+1)/4} \pmod{q}$;

4、计算 $x \equiv (tqu + spv) \pmod{n}$;

5、计算 $y \equiv (tqu - spv) \pmod{n}$;

6、同余式 $x^2 \equiv c \pmod{n}$ 的四个根是 x, -x (mod n), y, -y (mod n);

Code:

```
from Crypto.Util import number
import gmpy2

p =
852285650211289018533149345831290834419890452250225412985505704493898396090 19
q =
111614714641364911312915294479850549131835378046002423977989457843071188836271
n = p * q
c =
766500368283066645619389449101598964164785482664717787314198410720209908147598 48
278060072878304728996168180809072766067444674534459089230549753936235095 39

u = pow(c, (p + 1) // 4, p)
v = pow(c, (q + 1) // 4, q)
s = int(gmpy2.invert(p, q))
t = int(gmpy2.invert(q, p))
x = (t * q * u + s * p * v) % n
y = (t * q * u - s * p * v) % n

print (number.long_to_bytes(x % n))
print (number.long_to_bytes(-x % n))
print (number.long_to_bytes(y % n))
print (number.long_to_bytes(-y % n))
```

flag

```
hgame{3xgCd~i5_re4l1y+e@sy^r1ght?}
```

WhitegiveRSA

```
N = 882564595536224140639625987659416029426239230804614613279163
e = 65537
c = 747831491353896780365654517748216624798517769637260742155527
```

简简单单 RSA N 分解因数得到 p, q

```
import gmpy2
from Crypto.Util import number

N = 882564595536224140639625987659416029426239230804614613279163
e = 65537
c = 747831491353896780365654517748216624798517769637260742155527

q = 857504083339712752489993810777
p = 1029224947942998075080348647219

d = int(gmpy2.invert(e, (p - 1) * (q - 1)))
m = pow(c, d, N)
m = number.long_to_bytes(m)

print(m)
```

flag

```
hgame{wOw~yOU_kNoW+R5@!}
```

The Password

```
    y_1=x_1⊕n_1⊕(x_1⋙7)⊕(x_1⋘3)
    y_2=x_2⊕n_2⊕(x_2⋙4)⊕(x_2⋘9)
    y_3=x_3⊕n_3⊕(x_3⋙2)⊕(x_3⋘5)
    y_4=x_4⊕n_4⊕(x_4⋙6)⊕(x_4⋘13)
    y_5=x_5⊕n_5⊕(x_5⋙8)⊕(x_5⋙16)
    y_6=x_6⊕n_6⊕(x_6⋙5)⊕(x_6⋘7)
    y_7=x_7⊕n_7⊕(x_7⋙2)⊕(x_7⋘5)
    (y_1,n_1) = (15789597796041222200,14750142427529922)
    (y_2,n_2) = (8279663441787235887,2802568775308984)
    (y_3,n_3) = (9666438290109535850,15697145971486341)
    (y_4,n_4) = (10529571502219113153,9110411034859362)
    (y_5,n_5) = (8020289479524135048,4092084344173014)
    (y_6,n_6) = (10914636017953100490,2242282628961085)
    (y_7,n_7) = (4622436850708129231,10750832281632461)
```

XOR-rot 加密

x 可以看作 64 阶单位矩阵与 64 位矩阵向量相乘，加密过程可以表示为 //以第一组为例

$$\overline{\mathrm{y}} = M\overline{\mathrm{x}} \oplus \overline{\mathrm{n}}$$

或者表示为

$$y_i = M_{ij}x^j \oplus n_i$$

$$M_{ij} = \delta_{ij} \oplus \delta_{(i+7)\ j} \oplus \delta_{(i-3)\ j}$$

$$\delta_{ij} = \begin{cases} 1, & if \quad i = j \\ 0, & if \quad i \neq j \end{cases}$$

因为 x 中的元素为 {0，1} 即模为 2，解密为 //M^{-1} 为 M 关于模为 2 的模逆矩阵

$$\overline{\mathrm{x}} = M^{-1}(\overline{\mathrm{y}} \oplus \overline{\mathrm{n}})$$

Code:

```
import gmpy2
import numpy as np
from Crypto.Util import number

cipher = [15789597796041222200, 8279663441787235887, 9666438290109535850,
10529571502219113153, 8020289479524135048, 10914636017953100490,
4622436850708129231]
key = [14750142427529922, 2802568775308984, 15697145971486341, 9110411034859362,
4092084344173014, 2242282628961085, 10750832281632461]
RO_R = [7, 4, 2, 6, 8, 5, 2]
RO_L = [3, 9, 5, 13, -16, 7, 5]
flag = b''

for count in range(7):
    x = [0] * 64
    y = [0] * 64
    cipher[count] = cipher[count] ^ key[count]
    arr1 = np.identity(64, dtype = 'int64')
    arr2 = np.zeros((64, 64), dtype = 'int64')
    arr3 = np.zeros((64, 64), dtype = 'int64')

    for i in range(64):
        for j in range(64):
            if ((i + RO_R[count]) % 64 == j):
                arr2[i][j] = 1
    for i in range(64):
        for j in range(64):
            if ((i - RO_L[count]) % 64 == j):
                arr3[i][j] = 1
    arr4 = arr1 ^ arr2 ^ arr3

    arr5 = np.zeros((64, 64), dtype = 'int64')
    for i in range(64):
        for j in range(64):
            arr = np.delete(arr4, i, 0)
            arr = np.delete(arr, j, 1)
            if (np.linalg.det(arr) > 0):
                M = np.linalg.det(arr) + 0.5
```

```
            else:
                M = np.linalg.det(arr) - 0.5
            arr5[i][j] =  M * pow(-1, i + j)
    k = int(gmpy2.invert(int(np.linalg.det(arr4)), 2))
    arr5 = (k * arr5.T) % 2

    for i in range(64):
        y[i] = cipher[count] % 2
        cipher[count] = cipher[count] // 2

    plain = 0
    for i in range(64):
        for j in range(64):
            x[i] += y[j] * arr5[i][j]
        x[i] = x[i] % 2
        plain = plain + int(x[i] * (2 ** i))
    flag += number.long_to_bytes(int(plain))

print(flag)
```

flag

```
hgame{l1neOr_a1gebrO&is@1mpor1On1^1n$crypto}
```