队伍名词：trash_fish

队伍ID：#0x00000c

队伍 Token: NYQkBpX0orAyC6xc2oudn

解出题目：

问卷：明年见！

CRYPTO：Ancient Recall，Intergalactic Bound，SPiCa

MISC：Invest in hints，Level 729 易画行，Computer cleaner plus

问卷

明年见！

填完问卷后拿到flag

CRYPTO

Ancient Recall

眼花缭乱的英文，直接丢给豆包解一下，然后不出意外的失败了，但豆包解Fortune_wheel的思路还是可以抄一下的

```
def de_Fortune_wheel(FATE):
    ol=[0 for i in range(5)]
    a_2=FATE[0]+FATE[2]+FATE[4]-FATE[1]-FATE[3]
    a=a_2//2
    ol[0]=a
    for i in range(1,5):
        ol[i]=FATE[i-1]-ol[i-1]
    return ol
```

接下来，因为正数^负数一定是负的，所以根据+-号可以判断是否是带re的（那个词我忘了……）

exp:

```
YOUR_final_Value=[253295195206629177489049836911419591724079470491821052057106708531147467501 9, 25329519520662917748904

Major_Arcana = ["The Fool", "The Magician", "The High Priestess","The Empress", "The Emperor", "The Hierophant","The L
wands = ["Ace of Wands", "Two of Wands", "Three of Wands", "Four of Wands", "Five of Wands", "Six of Wands", "Seven of
cups = ["Ace of Cups", "Two of Cups", "Three of Cups", "Four of Cups", "Five of Cups", "Six of Cups", "Seven of Cups",
swords = ["Ace of Swords", "Two of Swords", "Three of Swords", "Four of Swords", "Five of Swords", "Six of Swords", "S
pentacles = ["Ace of Pentacles", "Two of Pentacles", "Three of Pentacles", "Four of Pentacles", "Five of Pentacles", "
Minor_Arcana = wands + cups + swords + pentacles
tarot = Major_Arcana + Minor_Arcana
reversals = [0,-1]

def de_Fortune_wheel(FATE):
    ol=[0 for i in range(5)]
    a_2=FATE[0]+FATE[2]+FATE[4]-FATE[1]-FATE[3]
    a=a_2//2
    ol[0]=a
    for i in range(1,5):
        ol[i]=FATE[i-1]-ol[i-1]
    return ol
Value=YOUR_final_Value[:]
for i in range(250):
    Value=de_Fortune_wheel(Value)
```

```
ini=[]
for i in Value:
    if i<0:
        i=i^-1
        ini+=['re-'+tarot[i]]
    else:
        ini+=[tarot[i]]

print(ini)
#hgame{re-The_Moon&re-The_Sun&Judgement&re-Temperance&Six_of_Cups}
```

Intergalactic Bound

第一眼看到往Knapsack方向去想了，结果发现这题的加密就是这样的，和Knapsack没关系.....

此事在糖醋小鸡块的博客中亦有记载

https://tangcuxiaojikuai.xyz/post/689431.html

数据量比那题要小一点，所以在最后求解ECDLP的时候不用转换也可以解

exp:

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
import hashlib

p = 550990553680539486102767863 01
a = 390818107333806152607250351 89
P = (196634467629629276330379267 40, 350744124309156560717770153 20)
Q = (268051376735366358258843301 80, 263768331126093094759511868 83)
ciphertext=b"k\xe8\xbe\x94\x9e\xfc\xe2\x9e\x97\xe5\xf3\x04'\x8f\xb2\x01T\x06\x88\x04\xeb3Jl\xdd Pk$\x00:\xf5"


############################################################ part1 get d
d = (a*P[0]^3 + P[1]^3 + 1) * inverse(P[0]*P[1], p) % p
print(d)
############################################################ part2 dlp
R.<x,y,z> = Zmod(p)[]
cubic = a*x^3 + y^3 + z^3 - d*x*y*z
E = EllipticCurve_from_cubic(cubic,morphism=True)
P = E(P)
Q = E(Q)
#factor(Q.order())
r = 13812057089
#m = (r*Q).log(r*P)
x=discrete_log(Q,P,operation='+')
print(x)
key = hashlib.sha256(str(x).encode()).digest()
cipher = AES.new(key, AES.MODE_ECB)
plain=cipher.decrypt(ciphertext)
print(plain)
#b'hgame{N0th1ng_bu7_up_Up_UP!}\x04\x04\x04\x04'
```

SPiCa

在lazzaro的格那块找到了这是隐子集和问题(HSSP)

此事在糖醋小鸡块的博客中亦有记载

https://tangcuxiaojikuai.xyz/post/a02848e0.html

脚本只需要前面大半部分即可求解无需求出那个一维矩阵

exp:

```
from Crypto.Util.number import *

#Zmod(M): xA = B (x is binary Matrix)
B = ()#太长了，不展示

M = 24727704801291912268835129736340977567569865784366882566681759917843647658060231409536848349518003784121914409876941

n = 70     #列   x:1*31
m = 247     #行

########################################## part1 solve Orthogonal Lattice of B
if(1):
    BL = block_matrix(ZZ,[
        [M,0],
        [Matrix(ZZ,B).T,1]
    ])
    OL = BL.LLL()
    OL = Matrix(ZZ,OL[:m-n,1:])


########################################## part2 find kernel and reduce
Ker = OL.right_kernel().matrix()
Ker = Ker.BKZ()

########################################## part3 recover binary x(use greedy methods)
def check(v):
    if(all(i == 1 or i == 0 for i in v)):
        return v
    elif(all(i == -1 or i == 0 for i in v)):
        return -v

def find(Ker,x):
    x = [i for i in ini]
    while(1):
        for vi in x:
            for i in Ker:
                xi1 = check(i + vi)
                xi2 = check(i - vi)
                if xi1 and xi1 not in x:
                    x.append(xi1)
                    if(len(x) == n):
                        return Matrix(ZZ,x)
                if xi2 and xi2 not in x:
                    x.append(xi2)
                    if(len(x) == n):
                        return Matrix(ZZ,x)

ini = [check(vi) for vi in Ker if check(vi)]
x = find(Ker,ini)


for i in x:
    plain=long_to_bytes(int(''.join(map(str,i)),2))
    if b'hgame' in plain:
        print((plain))
#b'hgame{U_f0und_3he_5pec14l_0n3!}'
```

MISC

Invest in hints

挺有趣的misc题

Hint的01序列的1是和hint对应的，首先flag的长度就是Hint的长度(71)，然后把hint依次填入Hint中为1的地方，这样你就能得到一个残缺的flag，如：

```
Hint=010100111
hint=abcde
#那么你可以知道部分flag是*a*b**cde
```

所以当你得到了n个hint后，你就能拼凑出完整的flag

所以首先要找出最小的子集，使得我们能拼出完整的flag，然后填进去即可

注意不要买错位了！

注意不要买错位了！

注意不要买错位了！

当我发现程序在填入的时候报错了，一切都来不及了……

exp:

```
raw="""Hint 51: 0000110010100111101000000001001000111010000000000000000000110111100010
Hint 52: 0110100011101100000000010100010000100110110000000000100100011100110000000
Hint 53: 1010010000000101100011000100110100001000110101110101011000100000000000000
Hint 54: 0000101000001001000010011000010000001000010010110011100000101110000011100
Hint 55: 0111001010010010000000000000000001101011001100000111100010110000000001000
Hint 56: 0111010000100100001001011110111011101001000100010011001000010011100000000
Hint 57: 1000010101000000001100000110010100101011010000011011001000100010011000000
Hint 58: 0000011110100000100100000110010010000011000011000010100000110111010000000
Hint 59: 0100110100100100000000100100111010000000000000101100010001000010101010100
Hint 60: 1001001010011001101110001001100110010010000111001001010100100010000111100
Hint 61: 0100100010001100000100000000001101000111000100000010110000100010001010000
Hint 62: 0010100001000011100010111000001000100000000100011110001000110100100110100
Hint 63: 0100001011101000000001010000101000101100010010001000000000000000010000000
Hint 64: 0111011011001100000001000001100000000100000000000000111000000010000010001
Hint 65: 0110000000001100011000000001000100000000001100110000011001000101101000000
Hint 66: 0111001100100010100110000101100001101000000110001010000001101000000010000
Hint 67: 0011101100001100000010010010100010010010100001000110011100100010000010000
Hint 68: 0100011001010101110011010111001000111110001101000000010101010000000010010
Hint 69: 1111101011100011010001000000001000110111101001101000110000001100000010010
Hint 70: 0000001011010110010010001100101101100110000010001001111100001100000110100
Hint 71: 0000110000111010100001011100110001110010001001110000101000000000100001000
Hint 72: 0110000000001100100101110000010100011011100010110001010111100000101010000
Hint 73: 0000100000101001000000110101011011000011011101101110010101111001010110000
Hint 74: 0101001010000000011101111000100001011010000100011100110101010000000010000
Hint 75: 1101000001100001010000101000011101101010000111101010010010000011111011000"""
#25*71

raw="{'"+raw.replace(': ',"':'").replace('\n',"','")+"'}"
dic=eval(raw)
#print(dic)

inv=[0]*71

ready=[[] for i in range(71)]

key=list(dic.keys())

for i in range(71):
    for j in range(25):
        if dic[key[j]][i]=='1':
            ready[i]+=[j]

#print(ready)

count=[0]*25

for i in range(25):
    for j in ready:
        if i in j:
```

```
            count[i]+=1
#print(count)

def upd():
    global count,inv,ready,select
    most_i=count.index(max(count))
    select+=[most_i]
    for i in range(71):
        if dic[key[most_i]][i]=='1':
            inv[i]=1
            ready[i]=[]
    count=[0]*25

    for i in range(25):
        for j in ready:
            if i in j:
                count[i]+=1

select=[]
while 0 in inv:
    upd()

#这些是你要买的hint，注意第一个的索引是0
print(select)

inv=[0]*71

for i in select:
    c=dic[key[i]]
    for j in range(len(c)):
        if c[j]=='1':
            inv[j]=1
print(inv==[1]*71)

select.sort()
hint=['mMk3ACi7SCWyAq3C5wda42','{AuYoACLQa2zq3i691hNlCxrALma42','megk9CiLrKWyAqi9hN8rELm}','hgamgko9CLgQSyzti1Dlu8r2mD

flag=[0]*71

for i in range(len(select)):
    c=dic[key[select[i]]][::-1]
    k=0
    for j in range(len(c)):
        if c[j]=='1':
            flag[j]=hint[i][k]
            k+=1
print(''.join(flag))
#hgame{Aug5YMkf3o99ACi7Lr0gQSCKaWy2Azq3ti691DhNlCbxu8rR2mCAD5LEwLdmHa42}
```

你也可以只求解花括号中的内容，因为hgame{}是已知的，这样说不定能少买点

Level 729 易画行

刚开始一头雾水，给了个交易的.ts文件还以为是要我去交易一个NFT给它（然后查询一下记录还真有人这么做了）

后来说这题不需要用sepolia的测试ETH，再加上在题目发布前已经有过一次交易，以及后来交易的人只有那么一个，远少于解出题目的人

我们可以猜测应该是要我们去查看这个NFT的元数据

先查询交易对象的地址

查询

https://sepolia.etherscan.io/address/0x74520Ad628600F7Cc9613345aee7afC0E06EFd84#nfttransfers

这里我们可以看到最早的那次交易记录，直接点击交易的那个NFT可以看到详情，合约的地址（Contract Address）就在里面

再去查询合约的地址，在Transactions里面有多个记录，其中最下面有一条Mint NFT，这是铸造NFT的记录，最有可能包含合约的ipfs地址，点击交易的哈希进去查看，展开下面的Click to show more，然后View In Decoder，就可以看到ipfs了

`ipfs://QmUusCYT8GTNgbDk5WAHZsHmHSxqcxuHov94inyFcpPqM6`

再去查询这个ipfs

https://ipfs.io/ipfs/QmUusCYT8GTNgbDk5WAHZsHmHSxqcxuHov94inyFcpPqM6

《网站有风险》《坚持访问》

然后就可以看到flag了

`flag{Tr4d1ng_on_t3st_n3t}`

Computer cleaner plus

这题真是乱点点出来的了

网上搜出来有条linux命令ps可以查看进程，但是在这台虚拟机里调用一直permission denied，别的也试不出来，就想去看看这个ps（毕竟报错的时候路径都给了，这对于一个没接触过linux的能忍住不去看看？）

cd到ps那里，只会cat……那就cat一下吧

然后就能看到那个CTF风格的文件名了……

hgame{B4ck_D0_oR}