- Crypto
  - SPiCa
  - Ancient Recall
  - Intergalactic Bound
- Misc
  - Computer cleaner plus
  - Invest in hints

SeanDictionary#0002f9

Crypto

SPiCa

标准的HSSP问题，存个档

题目

```
from Crypto.Util.number import *
from secrets import flag
from sage.all import *

def derive_M(n):
    iota=0.035
    Mbits=int(2 * iota * n^2 + n * log(n,2))
    M = random_prime(2^Mbits, proof = False, lbound = 2^(Mbits - 1))
    return Integer(M)

m = bytes_to_long(flag).bit_length()
n = 70
p = derive_M(n)


F = GF(p)
x = random_matrix(F, 1, n)
A = random_matrix(ZZ, n, m, x=0, y=2)
A[randint(0, n-1)] = vector(ZZ, list(bin(bytes_to_long(flag))[2:]))
h = x*A

with open("data.txt", "w") as file:
    file.write(str(m) + "\n")
    file.write(str(p) + "\n")
    for item in h:
        file.write(str(item) + "\n")
```

exp

这是个标准的HSSP问题

参考Tover师傅的文章

```
def checkMatrix(M, wl=[-1, 1]):
    M = [list(i) for i in list(M)]
    ml = list(set(flatten(M)))
    return sorted(ml) == sorted(wl)

def hssp_solve(n,m,M,h):
    ge1 = [[0]*m for _ in range(m)]
    tmp = pow(h[0], -1, M)
    for i in range(1,m):
        ge1[i][0] = -h[i]*tmp
        ge1[i][i] = 1
    ge1[0][0] = M

    Ge1 = Matrix(ZZ,ge1)

    L1 = Ge1.BKZ()

    Lx_orthogonal = Matrix(ZZ, L1[:m-n])
    Lx = Lx_orthogonal.right_kernel(algorithm='pari').matrix()

    e = Matrix(ZZ, [1] * m)
    B = block_matrix([[-e], [2*Lx]])

    L2 = B.BKZ()
    assert checkMatrix(L2)

    E = matrix(ZZ, [[1]*L2.ncols() for _ in range(L2.nrows())])
    L2 = (L2 + E) / 2

    assert set(L2[0]) == {0}

    L2 = L2[1:]

    space = Lx.row_space()

    Lx2 = []
    e = vector(ZZ, [1] * m)
    for lx in L2:
        if lx in space:
            Lx2 += [lx]
            continue
        lx = e - lx
        if lx in space:
            Lx2 += [lx]
            continue
        return None

    Lx = matrix(Zmod(M), Lx2)
```

```
    vh = vector(Zmod(M), h)
    va = Lx.solve_left(vh)
    return Lx, va

with open("data.txt", "r") as file:
    n = 70
    m = int(file.readline())
    M = int(file.readline())
    h = list(map(int, file.readline()[1:-2].split(", ")))

A,a = hssp_solve(n,m,M,h)

for row in A:
    ans = "".join(str(i) for i in row)
    try:
        print(long_to_bytes(int(ans,2)).decode())
    except:
        None

# hgame{U_f0und_3he_5pec14l_0n3!}
```

Ancient Recall

很简单一个倒推，小学数学题

```
Major_Arcana = ["The Fool", "The Magician", "The High Priestess","The Empress",
"The Emperor", "The Hierophant","The Lovers", "The Chariot", "Strength","The
Hermit", "Wheel of Fortune", "Justice","The Hanged Man", "Death", "Temperance","The
Devil", "The Tower", "The Star","The Moon", "The Sun", "Judgement","The World"]

wands = ["Ace of Wands", "Two of Wands", "Three of Wands", "Four of Wands", "Five
of Wands", "Six of Wands", "Seven of Wands", "Eight of Wands", "Nine of Wands",
"Ten of Wands", "Page of Wands", "Knight of Wands", "Queen of Wands", "King of
Wands"]
cups = ["Ace of Cups", "Two of Cups", "Three of Cups", "Four of Cups", "Five of
Cups", "Six of Cups", "Seven of Cups", "Eight of Cups", "Nine of Cups", "Ten of
Cups", "Page of Cups", "Knight of Cups", "Queen of Cups", "King of Cups"]
swords = ["Ace of Swords", "Two of Swords", "Three of Swords", "Four of Swords",
"Five of Swords", "Six of Swords", "Seven of Swords", "Eight of Swords", "Nine of
Swords", "Ten of Swords", "Page of Swords", "Knight of Swords", "Queen of Swords",
"King of Swords"]
pentacles = ["Ace of Pentacles", "Two of Pentacles", "Three of Pentacles", "Four of
Pentacles", "Five of Pentacles", "Six of Pentacles", "Seven of Pentacles", "Eight
of Pentacles", "Nine of Pentacles", "Ten of Pentacles", "Page of Pentacles",
"Knight of Pentacles", "Queen of Pentacles", "King of Pentacles"]

Minor_Arcana = wands + cups + swords + pentacles

tarot = Major_Arcana + Minor_Arcana

YOUR_final_Value =
[25329519520662917748904983691141959172407947049182105205710670853114746750 19,
```

```
25329519520662917748903276660741003578980230131054431788812947003815097952 70,
25329519520662917748905544592872766049031303158592585441730683769670723357 30,
25329519520662917748908653282415328853915101626115345140144091742842991390 15,
25329519520662917748908306626081341560179463763099899341758339139211426093 34]

def re_Fortune_wheel(FATE):
    sums = sum(FATE)//2
    FATE_re = [sums-(FATE[(i+1)%5]+FATE[(i+3)%5]) for i in range(len(FATE))]
    return FATE_re

YOUR_initial_Value = YOUR_final_Value

for _ in range(250):
    YOUR_initial_Value = re_Fortune_wheel(YOUR_initial_Value)
print(YOUR_initial_Value)

YOUR_initial_FATE = []
for i in YOUR_initial_Value:
    k = -1 if i < 0 else 0
    index = i^k
    card = tarot[index]
    if card in Major_Arcana:
        if k == -1:
            YOUR_initial_FATE.append("re-"+card)
        else:
            YOUR_initial_FATE.append(card)
    else:
        YOUR_initial_FATE.append(card)

FLAG=("hgame{"+"&".join(YOUR_initial_FATE)+"}").replace(" ","_")
print(FLAG)

# hgame{re-The_Moon&re-The_Sun&Judgement&re-Temperance&Six_of_Cups}
```

Intergalactic Bound

Twisted Hessian Curves曲线，记录一下

一般方程

$$ax^3 + y^3 + 1 = dxy$$

加法

$$(x_1, y_1) + (x_2, y_2) = \left(\frac{x_1 - y_1^2 x_2 y_2}{a x_1 y_1 x_2^2 - y_2}, \frac{y_1 y_2^2 - a x_1^2 x_2}{a x_1 y_1 x_2^2 - y_2}\right)$$

乘法

$$2(x, y) = \left(\frac{x - y^3 x}{a x^3 y - y}, \frac{y^3 - a x^3}{a x^3 y - y}\right)$$

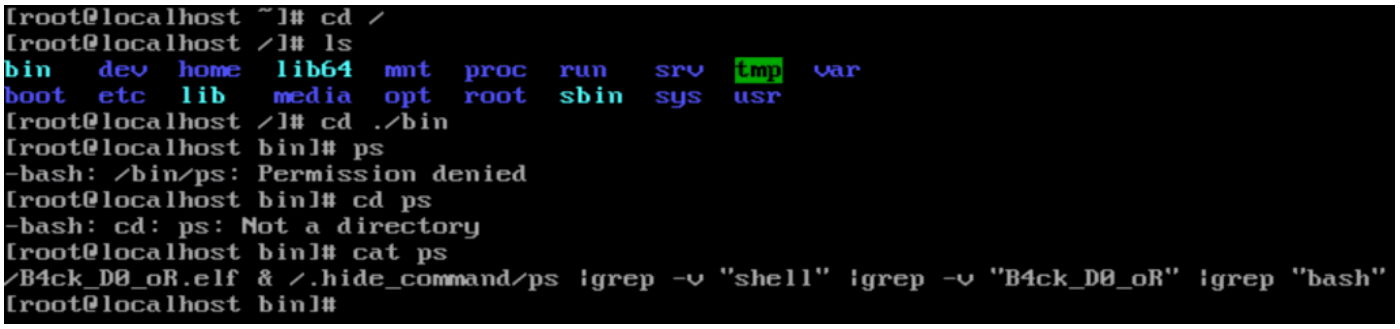
取反

$$-(x,y)=(\frac{x}{y},\frac{1}{y})$$

```
# HGEME 2025
# Intergalactic Bound

from Crypto.Util.number import *
from Crypto.Cipher import AES
import hashlib
p = 55099055368053948610276786301
G = (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\x0
4\xeb3Jl\xdd Pk$\x00:\xf5"

a = (G[0]*G[1]*(Q[1]^3+1) - Q[0]*Q[1]*(G[1]^3+1)) * inverse(G[0]*Q[0]*(G[0]^2*Q[1]
- Q[0]^2*G[1]),p)%p
d = (a*G[0]^3+G[1]^3+1)%p*inverse(G[0]*G[1],p)%p

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)
G = E(G)
Q = E(Q)
G_ord = G.order()

def Pohlig_Hellman(n,G,Q):
    factors, exponents = zip(*factor(n))
    primes = [factors[i] ^ exponents[i] for i in range(len(factors))]
    print(primes)
    dlogs = []
    for fac in primes:
        t = int(int(n) // int(fac))
        dlog = discrete_log(t*Q,t*G,operation="+")
        dlogs += [dlog]
        print("factor: "+str(fac)+", Discrete Log: "+str(dlog)) #calculates
discrete logarithm for each prime order
    num = crt(dlogs,primes)
    return num

x = Pohlig_Hellman(G_ord,G,Q)
print(x)

key = hashlib.sha256(str(x).encode()).digest()
cipher = AES.new(key, AES.MODE_ECB)
flag = cipher.decrypt(ciphertext)
print(flag)

# hgame{N0th1ng_bu7_up_Up_UP!}
```

Misc

Computer cleaner plus



Invest in hints

原题提到这一句话↓

每个 Hint 按原串顺序包含以下位（个位代表原串的第一个字符）

这句话的意思是按照flag的字符顺序，用01表示字符取舍，hint里面要用点数买对应取舍的字符串。并且个位代表第一个字符意味着01串是倒序的

所以就能跑脚本确定要开的最小个数

```
hint1 = '''
00001100101001111010000000100100011101000000000000000000011011110001000
01101000111011000000000101000100001001101100000000100100011100110000000
10100100000001011000110001001101000010001101011101010110001000000000000
00001010000010010000100110000100000010000100101100111000001011100000111
01110010100100100000000000000000011010110011000001111000101100000001000
01110100001001000010010111101111011101001000100010011001000010011100000
10000101010000000011000001100101001010110100000110110010001000100011000
00000111101000001001000001100100100000110000110000101000001101110100000
01001101001001000000001001001110100000000000010110001000100001010101010
10010010100110011011100010011001100100100001110010010101001000100001111
01001000100011000001000000000011010001110001000000101100001000100010100
00101000010000111000101110000010001000000001000111100010001101001001101
01000010111010000000010100001010001011000100100010000000000000001000000
01110110110011000000010000011000000010000000000000111000000010000010001
01100000000011000110000000010001000000000011001100000110010001011010000
01110011001000101001100001011000011010000001100010100000011010000001000
00111011000011000000100100101000100100101000010001100111001000100001000
01000110010101011100110101110010001111100011010000000101010100000010010
11111010111000110100010000000010001101111010011010001100000011000001001
00000010110101100100100011001011011001100000100010011111000011000001101
00001100001110101000010111001100011100100010011100001010000000001000010
```

```
011000000000110010010111000001010001101110001011000101011110000010101000
000010000010100100000011010101101100001101110110111001010111100101100000
010100101000000001110111100010000101101000100011100110101010000001000011
110100000110000101000010100001110110101000011110101001001000001111101100
'''.split("\n")[1:-1]

def count(lis:list):
    ans = [0]*71
    for i in lis:
        for j in range(71):
            if hint1[i][j] == "1":
                ans[j] += 1
    return ans

def check(ans):
    for i in ans[1:-6]:
        if i == 0:
            return False
    return True

for a in range(25-4):
    for b in range(a,25-3):
        for c in range(b,25-2):
            for d in range(c,25-1):
                for e in range(d,25):
                    ans = count([a,b,c,d,e])
                    if check(ans):
                        print([a,b,c,d,e])

# [0, 16, 19, 21, 22]
# [4, 6, 19, 21, 22]
# [5, 6, 19, 21, 22]
# [5, 7, 19, 21, 22]
# [5, 8, 19, 21, 22]
# ......
```

这里是从3开始都跑了一下发现最少要5个hint才能解出flag

所以开第1，17，20，22，23个hint然后写脚本

```
hint1 = '''
000011001010011111010000000010010001110100000000000000000001101111000100
001110110000110000001001001010001001001010000100011001110010001000010000
000000101101011001001000110010110110011000001000100111110000110000011101
011000000000110010010111000001010001101110001011000101011110000010101000
000010000010100100000011010101101100001101110110111001010111100101100000
'''.split("\n")[1:-1]

hint2 = '''
aAug5MkyAzq6Dr2mCALwmH
mgko99i7gayzt1hCuADLdHa4
ham5Yo99ACLQy2q3i61NluRCA5Ewd
aeAkf3o9Cr0QaWyAzi9Cbx82AD42
e{uYMkfo9i7L0gSCKWy3t69DNCbmDLH
```

```
'''.split("\n")[1:-1]

ans = [set() for _ in range(71)]
for i in range(5):
    index = 0
    for j in range(71):
        if hint1[i][::-1][j] == "1":
            ans[j].add(hint2[i][index])
            index += 1
for i in ans:
    print(*i if i != set() else " ", end = "")

# h ame{Aug5YMkf3o99ACi7Lr0gQSCKaWy2Azq3ti691DhNlCbxu8rR2mCAD5LEwLdmHa42
```

得到hgame{Aug5Y