hgame2025 WriteUp

签到

TEST NC

```
cat flag
hgame{YouR_cAn-ConN3Ct_to-TH3-R3MoTe-ENvIrOnMeNT-TO-gEt_f1@g0}
```

从这里开始的序章。

```
I am the flag!
hgame{Now-I-knOw-how-to-subm1t-my-fl4gs!}
```

也是我的终章😊

Crypto

suprimeRSA

先是一个小问题 a!+b!=a^b ,注意到右式增长速度远大于左式，a、b取值一定较小，试根得到a=2,b=2

根据源码：质数p=k*M+pow(e,a,M)，k和a随机生成，几乎无法爆破。

但是发现模数为96位大整数，可以通过二次筛法分解质因数，在msieve计算1h28min后得出结果：

```
Sat Feb 08 15:00:13 2025  p48 factor:
796688410951167236263039235508020180383351898113
Sat Feb 08 15:00:13 2025  p48 factor:
839777194079410698093279448382785381699926556673
Sat Feb 08 15:00:13 2025  elapsed time 01:28:00
```

```
from Crypto.Util.number import *

e = 0x10001
M = 2**130
enc = 
48720728317601882496526817230788824576381758387507100886937017256577723051437923673374284657584
9
n = 
66904075830415567557016782475969192110693575027076599713944685183048984473137372123329081625804
9
p = 796688410951167236263039235508020180383351898113
q = 839777194079410698093279448382785381699926556673

phi = (p - 1) * (q - 1)
d = inverse(e, phi)
m = pow(enc, d, n)
print(long_to_bytes(m))  # b'hgame{ROCA_ROCK_and_ROll!}'
```

之后我发现https://factordb.com这个网站把这个数的分解给秒了？？？
估计是算出来后给记下来了，题目附件也是及时换了，一看发现n有144位，原来考点不在分解因数上？

sieve

根据题意，需要算出715849728以内欧拉数之和，其中质数和1的欧拉数再加1。这里当然用筛法做：

欧拉数 $\phi(n) = n(1 - \frac{1}{p_1})(1 - \frac{1}{p_2})\dots(1 - \frac{1}{p_n}))$

先令 $\phi(n) = n$ ，每找到一个质数p，就将所有kp的欧拉数乘以 $(1 - \frac{1}{p})$ 这样对于任意一个数n，它的每个质数都能被筛一遍。

那怎么找质数呢？遍历 2~n，如果发现 $n = \phi(n)$，那就说明n没有被更小的素数筛过，那就是质数了。

由于本题要求质数和1的欧拉数再加1，所以代码中筛的时候从2*p开始筛，最后还要加1

这会就是C语言的魅力时刻了，python跑了2个小时左右，C语言半分钟都不用就出来了。

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <stdbool.h>

void get_phi(int n, int *phi)
{
    for (int i = 2; i <= n; i++)
    {
        phi[i] = i; // 初始化欧拉函数值
    }

    for (int p = 2; p <= n; p++)
    {
        if (phi[p] == p)
        {
            for (int multiple = 2 * p; multiple <= n; multiple += p)
            {
                phi[multiple] = phi[multiple] / p * (p - 1); // 更新欧拉函数值
            }
        }
    }
}

int main()
{
    int n = 715849728;
    long long sum = 0;
    int *phi = (int *)malloc((n + 1) * sizeof(int));

    get_phi(n, phi);

    for (int i = 2; i <= n; i++)
    {
        sum += phi[i];
    }
    printf("sum = %lld\n", sum + 1); // sum = 155763335447735055
```

```
    free(phi);
    return 0;
}
```

```
from Crypto.Util.number import *
from sympy import nextprime

p = q = nextprime(155763335447735055 << 128)
n = p * q
e = 65537
phi = (p - 1) * q
d = inverse(e, phi)
enc = 
24492940974747141365301400997845927327664444816652780380694844666655061539678510632094023360250654761726173765 46
m = pow(enc, d, n)
print(long_to_bytes(m))  # b'hgame{sieve_is_nOt_that_HArd}'
```

Reverse

Compress dot new

脚本语言是nushell，功能就是哈夫曼编码，解密脚本在此省略huffman_tree

```
def huffman_decode(encoded_data, huffman_tree):
    decoded_string = ""
    current_node = huffman_tree

    for bit in encoded_data:
        if bit == "0":
            current_node = current_node["a"]
        else:
            current_node = current_node["b"]

        try:
            current_node["s"]
            decoded_string += chr(current_node["s"])
            current_node = huffman_tree  # 重置到根节点
        except:
            pass

    return decoded_string
```

```
encoded_data =
"0001000111011111101001000001110001011100010011100011000010001011100111001001101l
0101011110111011001101000111011010011101111101110110110011101100111100111101101l1
011101101011001111011001111000111001101111000011001100001011011101100011100101001
110010111001111000011000101001010000000100101000100010011111110110010111010101000
1111010001101100011101010110100111111110011111110110101011000011011101011011111l0
100100111100100010110101111111111100110001010101101110010011111000110110101101111
010000011110100000110110101011000111111000110101001011100000110111100000010010100
010001011100011100111001011101011111000101010110101111000001100111100011100101110
101111110001011010111000001010000000101100011110111000111011111101010100100111010l1
100100011110010010110111101110111010111110110001111010101110010001011100100101110
001011010100001110101000101111010100110001110101011101100011011011000011010000001
0110001110111111111100010101011100000"
m = huffman_decode(encoded_data, huffman_tree)
print(m)
'''
hgame{Nu-Shell-scr1pts-ar3-1nt3r3st1ng-t0-wr1te-&-use!}
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Nulla nec ligula neque. Etiam et viverra nunc, vel bibendum risus. Donec.
'''
```

解出来后还有个彩蛋，知乎上的解释：https://www.zhihu.com/question/20133127?sort=created

Turtle

先手动UPX脱壳

找到源码，下面是我改过：

```
__int64 sub_401876()
{
    _BYTE v1[256];
    char flag[46];
    char key[8];
    char encrypted_flag[40] = {-8, -43, 98, -49, 67, -70, -62, 35, 21, 74, 81,
16, 39, 16, -79, -49, -60, 9, -2, -29, -97, 73, -121, -22, 89, -62, 7, 59, -87,
17, -63, -68, -3, 75, 87, -60, 126, -48, -86, 10};
    _BYTE encrypted_key[7] = {-51, -113, 37, 61, -31, 'Q', 'J'};
    char v8[7] = "yekyek";

    puts("plz input the key: ");
    scanf("%s", key);
    set_v1(v8, 6, v1);
    encrypt1(key, 7, v1);
    if (memcmp(key, encrypted_key, 7))
    {
        puts("key is wrong");
    }
    else
    {
        puts("plz input the flag:");
        scanf("%s", flag);
        set_v1(v8, 7, v1);
        encrypt2(flag, 40, v1);
        if (memcmp(flag, encrypted_flag, 40))
```

```
            puts("wrong, plz try again");
        else
            puts("Congratulate!");
    }
    return 0LL;
}
```

显然，关键函数set_v1、encrypt1、encrypt2，只要逆向后两个函数即可，注意到encrypt1是异或加密，直接可以当作解密函数用，而encrypt2跟对明文的变化只在于一行的 `-=` ，只要改为 `+=` 就能用了。

```
int main()
{
    _BYTE v1[256];
    char key[7] = {-51, -113, 37, 61, -31, 'Q', 'J'};
    char v8[7] = "yekyek";
    char v2[40] = {-8, -43, 98, -49, 67, -70, -62, 35, 21, 74, 81, 16, 39, 16,
-79, -49, -60, 9, -2, -29, -97, 73, -121, -22, 89, -62, 7, 59, -87, 17, -63, -68,
-3, 75, 87, -60, 126, -48, -86, 10};

    set_v1(v8, 6, v1);
    decrypt1(key, 7, v1);
    fwrite(key, sizeof(char), 7, stdout); // ecg4ab6
    putchar('\n');

    set_v1(key, 7, v1);
    decrypt2(v2, 40, v1);
    fwrite(v2, sizeof(char), 40, stdout); //
hgame{YOu'r3_re4l1y_g3t_OUt_of_th3_upX!}
    return 0;
}
```

Web

Pacman

翻源代码到/static/script/index.js，可以找到两组gift

```
here is your gift:aGFlcGFpZW1rc3ByZXRnbXtydGNfYWVfZWZjfQ==
here is your gift:aGFldTRlcGNhXzROcmdte19yX2Ftbm1zZXO=
```

base64解码：

```
haepaiemkspretgm{rtc_ae_efc}
haeu4epca_4trgm{_r_amnmse}
```

栅栏密码解密：

```
hgame{pratice_makes_perfect}
hgame{u_4re_pacman_m4ster}
```