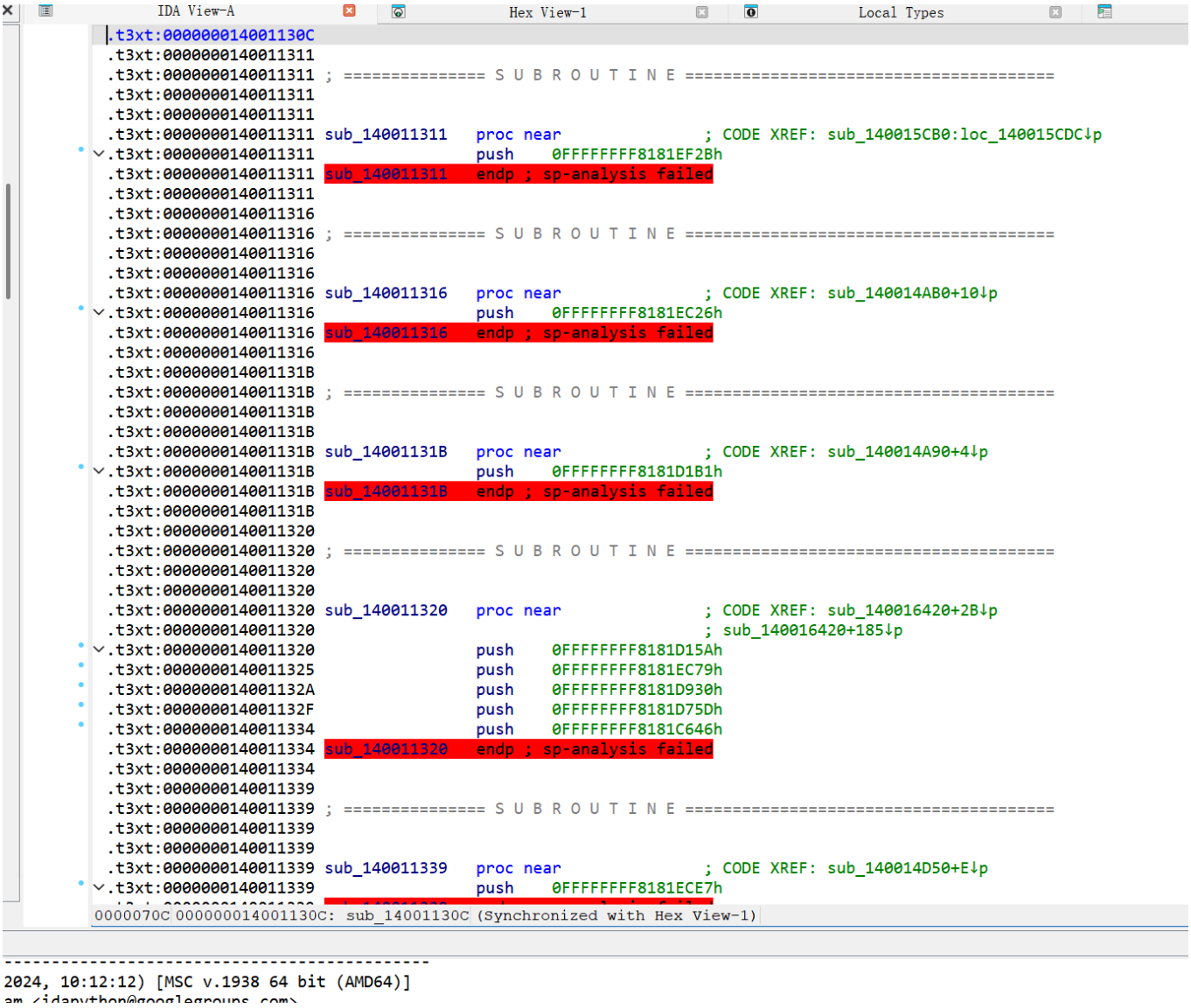
逆向wp-3

逆向题目wp

张程思

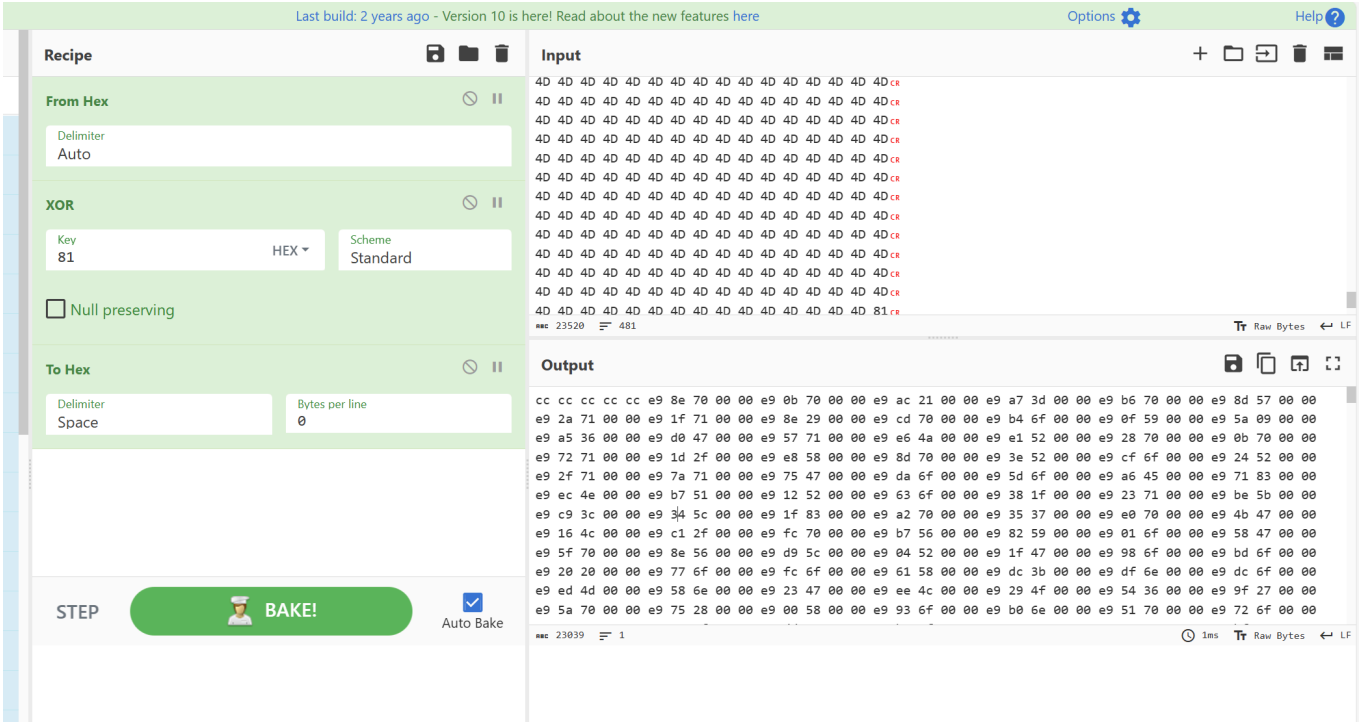
一,1z-RC4

1,ida打开发现一堆爆红,并且发现其16进制形式不太对劲

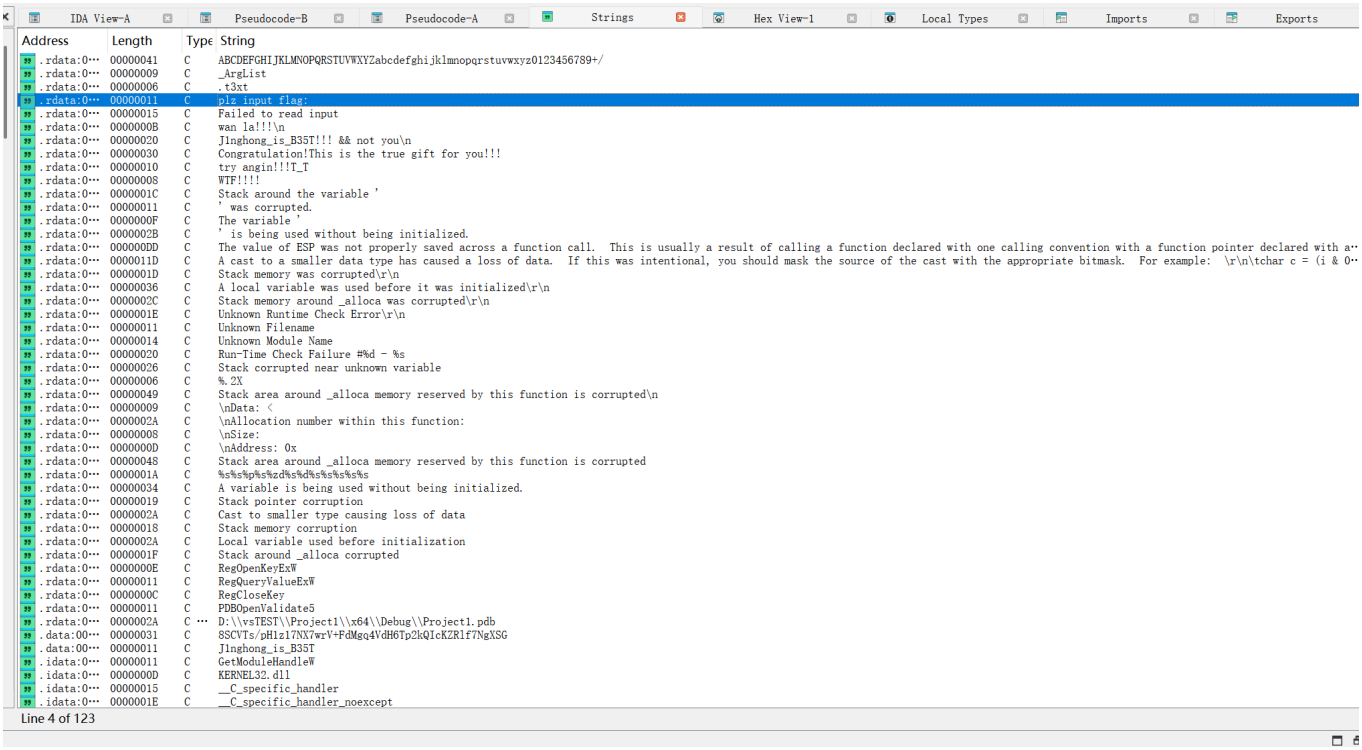


```
0140012BB0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012BC0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012BD0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012BE0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012BF0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C00  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C10  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C20  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C30  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C40  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C50  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C60  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C70  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C80  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012C90  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CA0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CB0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CC0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CD0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CE0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012CF0  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012D00  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012D10  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012D20  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012D30  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 4D  MMMMMMMMMMMMMMMM
0140012D40  4D 4D 4D 4D 4D 4D 4D 4D  4D 4D 4D 4D 4D 4D 4D 81  MMMMMMMMMMMMMMM.
0140012D50  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012D60  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012D70  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012D80  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012D90  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DA0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DB0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DC0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DD0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DE0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012DF0  81 81 81 81 81 81 81 81  81 81 81 81 81 81 81 81  ................
0140012E00  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0140012E10  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
0140012E20  ?? ?? ?? ?? ?? ?? ?? ??  ?? ?? ?? ?? ?? ?? ?? ??  ????????????????
```

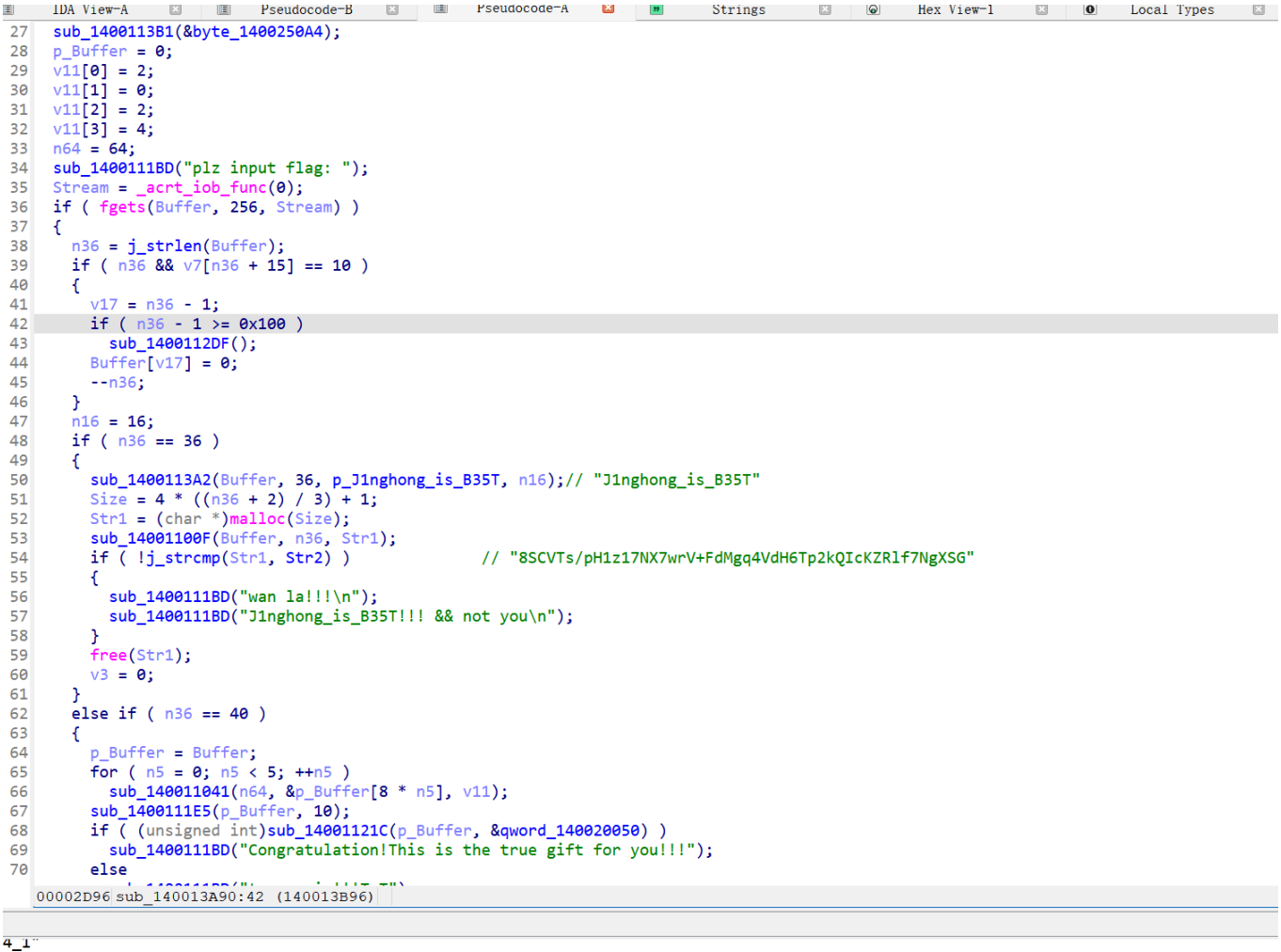
2,估计是需要单字节异或,先使用101打开,复制这些内容并改异或改成正确格式



3,现在就可以ida打开看到完整内容啦,先打开string,直接找到flag入口,点进去



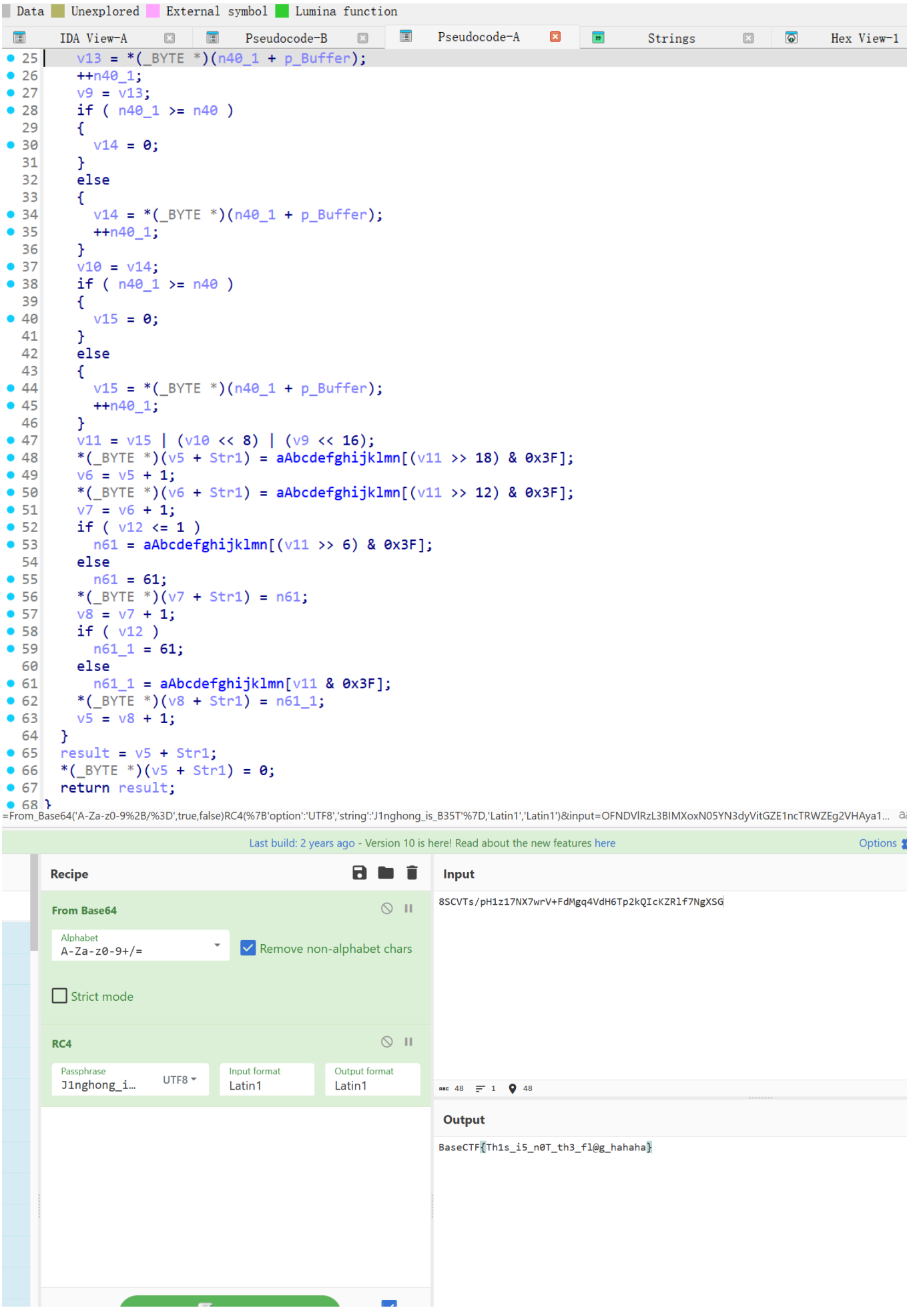
4,两部分,一个是假的,一个是真的flag



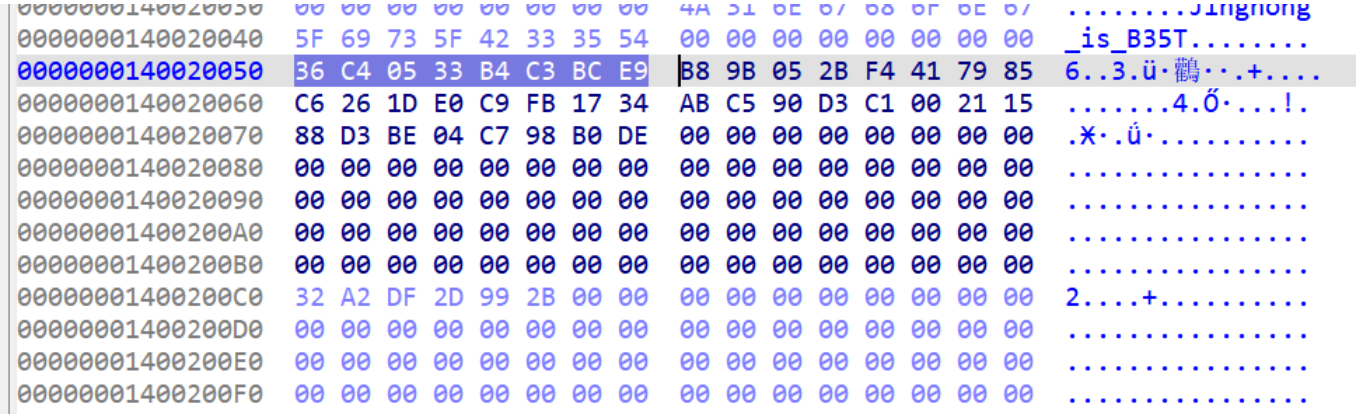
```
      v3 = 0;
    }
    else if ( n36 == 40 )
    {
      p_Buffer = Buffer;
      for ( n5 = 0; n5 < 5; ++n5 )
        sub_140011041(n64, &p_Buffer[8 * n5], v11);
      sub_1400111E5(p_Buffer, 10);
      if ( (unsigned int)sub_14001121C(p_Buffer, &qword_140020050) )
        sub_1400111BD("Congratulation!This is the true gift for you!!!");
      else
        sub_1400111BD("try angin!!!T_T");
      v3 = 0xFFFFFFFFLL;
    }
    else
    {
      sub_1400111BD("WTF!!!!");
      v3 = 0xFFFFFFFFLL;
    }
  }
  else
  {
    perror("Failed to read input");
    v3 = 1;
  }
  v4 = v3;
  sub_140011348(v6, &unk_14001CE10);
  return v4;
}
00002F45 sub_140013A90:76 (140013D45)
```

5,第一个if是rc4加上base64,解出来假的了

```
__int64 __fastcall sub_1400136E0(__int64 p_Buffer, unsigned int n36, __int64 p_J1ngho
{
  char *v4; // rdi
  __int64 n114; // rcx
  _BYTE v7[32]; // [rsp+0h] [rbp-20h] BYREF
  char v8; // [rsp+20h] [rbp+0h] BYREF
  _BYTE v9[276]; // [rsp+30h] [rbp+10h] BYREF
  int v10; // [rsp+144h] [rbp+124h]
  int v11; // [rsp+164h] [rbp+144h]
  int v12; // [rsp+184h] [rbp+164h]
  unsigned int n36_1; // [rsp+1A4h] [rbp+184h]
  char v14; // [rsp+1C4h] [rbp+1A4h]

  v4 = &v8;
  for ( n114 = 114; n114; --n114 )
  {
    *(_DWORD *)v4 = -858993460;
    v4 += 4;
  }
  sub_1400113B1(byte_1400250A4);
  sub_14001130C(v9, p_J1nghong_is_B35T, n16);
  v10 = 0;
  v11 = 0;
  for ( n36_1 = 0; n36_1 < n36; ++n36_1 )
  {
    v10 = (v10 + 1) % 256;
    v11 = ((unsigned __int8)v9[v10] + v11) % 256;
    v14 = v9[v10];
    v9[v10] = v9[v11];
    v9[v11] = v14;
    v12 = ((unsigned __int8)v9[v11] + (unsigned __int8)v9[v10]) % 256;
    *(_BYTE *)(p_Buffer + n36_1) ^= v9[v12];
  }
  return sub_140011348(v7, &unk_14001CD50);
}
```


Data Unexplored External symbol Lumina function
IDA View-A
Pseudocode-B
Pseudocode-A
Strings
Hex View-1
v13 = *(_BYTE *)(n40_1 + p_Buffer);
++n40_1;
v9 = v13;
if (n40_1 >= n40)
{
v14 = 0;
}
else
{
v14 = *(_BYTE *)(n40_1 + p_Buffer);
++n40_1;
}
v10 = v14;
if (n40_1 >= n40)
{
v15 = 0;
}
else
{
v15 = *(_BYTE *)(n40_1 + p_Buffer);
++n40_1;
}
v11 = v15 | (v10 << 8) | (v9 << 16);
*(_BYTE *)(v5 + Str1) = aAbcdefghijklmn[(v11 >> 18) & 0x3F];
v6 = v5 + 1;
*(_BYTE *)(v6 + Str1) = aAbcdefghijklmn[(v11 >> 12) & 0x3F];
v7 = v6 + 1;
if (v12 <= 1)
n61 = aAbcdefghijklmn[(v11 >> 6) & 0x3F];
else
n61 = 61;
*(_BYTE *)(v7 + Str1) = n61;
v8 = v7 + 1;
if (v12)
n61_1 = 61;
else
n61_1 = aAbcdefghijklmn[v11 & 0x3F];
*(_BYTE *)(v8 + Str1) = n61_1;
v5 = v8 + 1;
}
result = v5 + Str1;
*(_BYTE *)(v5 + Str1) = 0;
return result;
}
=From_Base64('A-Za-z0-9%2B/%3D',true,false)RC4(%7B'option':'UTF8','string':'J1nghong_is_B35T'%7D,'Latin1','Latin1')&input=OFNDVlRzL3BIMXoxN05YN3dyVitGZE1ncTRWZEg2VHAya1...
Last build: 2 years ago - Version 10 is here! Read about the new features here
Options
Recipe
From Base64
Alphabet
A-Za-z0-9+/=
Remove non-alphabet chars
Strict mode
RC4
Passphrase
J1nghong_i...
UTF8
Input format
Latin1
Output format
Latin1
Input
8SCVTs/pH1z17NX7wrV+FdMgq4VdH6Tp2kQIcKZRlf7NgXSG
48 1 48
Output
BaseCTF{Th1s_i5_n0T_th3_fl@g_hahaha}

6,第二个if是tea解码,这里把key方向的40字节告诉gpt,还有关键逻辑也扔给他就得出了flag



7,解题脚本如下

```
# We'll decrypt the 40-byte block you provided using the reverse of the TEA-
like algorithm
# described in the conversation. This code prints the recovered 40-byte flag
(both hex and UTF-8 if possible)
# and also shows the intermediate uint32 words for inspection.
import struct

# raw bytes from your memory dump lines (addresses + bytes)
hex_lines = [
    "36 C4 05 33 B4 C3 BC E9 B8 9B 05 2B F4 41 79 85",
    "C6 26 1D E0 C9 FB 17 34 AB C5 90 D3 C1 00 21 15",
    "88 D3 BE 04 C7 98 B0 DE 00 00 00 00 00 00 00 00"
]

# build bytearray
b = bytearray()
for line in hex_lines:
    parts = line.split()
    for p in parts:
        b.append(int(p, 16))

# verify length
assert len(b) == 40, f"expected 40 bytes, got {len(b)}"

# parse as 10 little-endian uint32 words
words_le = [struct.unpack_from("<I", b, i*4)[0] for i in range(10)]

def byteswap32(x):
    return ((x & 0xFF) << 24) | ((x & 0xFF00) << 8) | ((x & 0xFF0000) >> 8) |
((x >> 24) & 0xFF)

# apply byteswap to each word (this reverses the byteswap done in the program)
words_after_byteswap = [byteswap32(w) for w in words_le]
```

```
# TEA-like decryption parameters (derived from the disassembly)
DELTA = 1640531527  # 0x61C886F7
ROUNDS = 64
KEY = [2, 0, 2, 4]

mask32 = 0xFFFFFFFF

def decrypt_block(v0, v1, rounds=ROUNDS):
    # v0,v1 are unsigned 32-bit integers (the encrypted outputs after the
forward algorithm)
    # Start v7 = -DELTA * rounds  (mod 2^32)
    v7 = (-DELTA * rounds) & mask32
    # Work on local copies
    v5 = v0 & mask32
    v6 = v1 & mask32
    for _ in range(rounds):
        # undo v6 += (...) ^ (...) ^ 0x99  (note order in original)
        idx = ((v7 >> 11) & 3)
        part = ((KEY[idx] + v7) & mask32) ^ ( (v5 + (((v5 >> 5) ^ (16 * v5)) &
mask32)) & mask32) ^ 0x99
        v6 = (v6 - part) & mask32
        # undo v7 -= DELTA  -> v7 = v7 + DELTA
        v7 = (v7 + DELTA) & mask32
        # undo v5 += (...) ^ (...) ^ 0x88
        idx2 = (v7 & 3)
        part2 = ((v7 + (KEY[idx2] & mask32) + 102) & mask32) ^ ( (v6 + (((v6
>> 5) ^ (16 * v6)) & mask32)) & mask32) ^ 0x88
        v5 = (v5 - part2) & mask32
    return v5, v6

# Decrypt all 5 blocks (each block = two uint32 words)
plaintext_words = []
for i in range(0, 10, 2):
    enc_v0 = words_after_byteswap[i]
    enc_v1 = words_after_byteswap[i+1]
    dec_v0, dec_v1 = decrypt_block(enc_v0, enc_v1)
    plaintext_words.extend([dec_v0, dec_v1])

# pack plaintext words back into bytes little-endian
plaintext_bytes = bytearray()
for w in plaintext_words:
    plaintext_bytes += struct.pack("<I", w & mask32)

# Output results
print("Input bytes (40):", b.hex())
```

```
print("Words (little-endian read from memory):")
for i, w in enumerate(words_le):
    print(f" w[{i:02}] = 0x{w:08X}")
print("\nAfter applying byteswap (to undo program's byteswap):")
for i, w in enumerate(words_after_byteswap):
    print(f" wbs[{i:02}] = 0x{w:08X}")

print("\nDecrypted words (uint32 little-endian):")
for i, w in enumerate(plaintext_words):
    print(f" p[{i:02}] = 0x{w:08X}")

print("\nRecovered 40 bytes (hex):", plaintext_bytes.hex())
# Try to decode as utf-8 / printable ASCII
try:
    s = plaintext_bytes.decode('utf-8')
    print("\nRecovered as UTF-8 string:")
    print(s)
except Exception as e:
    # show printable representation
    printable = ''.join(chr(c) if 32 <= c < 127 else f"\\x{c:02x}" for c in
plaintext_bytes)
    print("\nRecovered bytes contain non-UTF8/Non-printable bytes. Printable
view:")
    print(printable)

# Also show as possible flag candidate with escape sequences
print("\nPrintable/escaped:", repr(plaintext_bytes))
```

8,flag如下

```
BaseCTF{Th1s_1s_A_g1ft_fr0m_NshIdE_4U}~~
```

这后面做几个简单的,尝试一下不用ai写脚本(随便找的简单题)

二,exe1

1,无壳,ida查看分析

IDA View-A | Pseudocode-A | Hex View-1 | Local

```
1  int __fastcall main(int argc, const char **argv, const char **envp)
2  {
3    int n3; // [rsp+2Ch] [rbp-4h]
4
5    sub_401800(argc, argv, envp);
6    puts("<---  moectf2021  --->");
7    puts(" [CheckIn] Welcome to moectf2021.");
8    puts("This is a really eazy chall. I believe it is easy for you to get the flag");
9    puts("Now input your flag and I will check it : ");
10   scanf("%100s", &unk_407040);
11   for ( n3 = 0; n3 <= 3; ++n3 )
12     ((void (__fastcall *)(void *))funcs_4016DE[n3])(&unk_407040);
13   if ( byte_403020 )
14   {
15     puts("Congratulations!!!");
16     puts("See you next chall!!!");
17   }
18   else
19   {
20     puts("Something went wrong!   QwQ");
21     puts("Try againt!!!");
22   }
23   puts(&Buffer_);
24   getchar();
25   getchar();
26   return 0;
27 }
```

2,点开exe试一下,发现提示有长度不对,且告诉了我们flag头

```
C:\Users\zhang\Desktop\第三
<---  moectf2021  --->
 [CheckIn] Welcome to moectf2021.
This is a really eazy chall. I believe it is easy for you to get the flag
Now input your flag and I will check it :
123
Wrong length!
In this game, every flag begin with moectf{
where is your '}' ???
Wrong flag!
Something went wrong!   QwQ
Try againt!!!
按任意键以继续
```

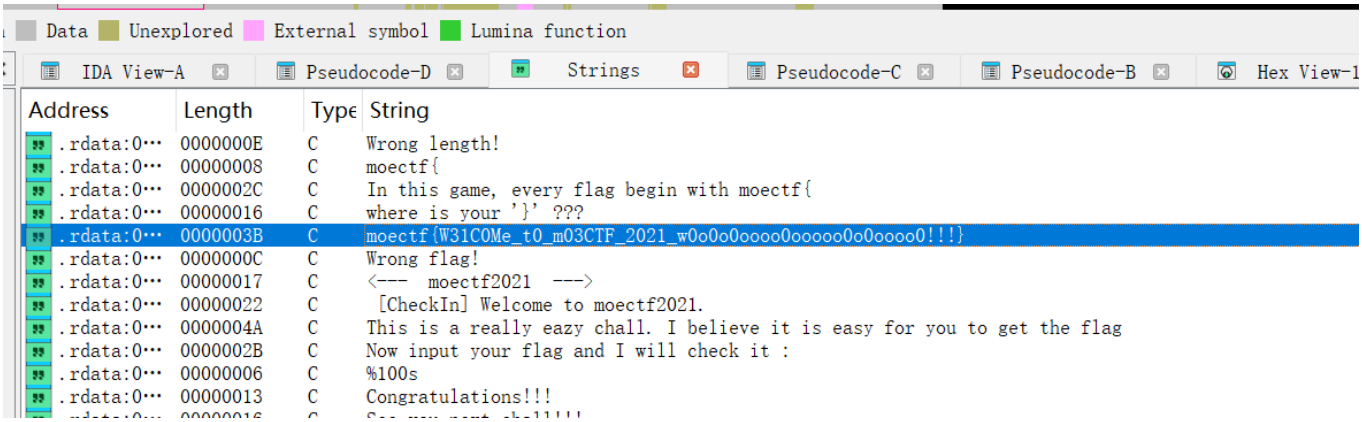
3,查看关键函数,发现需要长度为58

```
.data:0000000000403028 off_403028      dq offset aMoectfW31c0meT
.data:0000000000403028                                          ; DATA XREF: sub_401606:loc_40161E
.data:0000000000403028                                          ; "moectf{W31C0Me_t0_m03CTF_2021_w
.data:0000000000403030                 align 20h
.data:0000000000403040 funcs_4016DE    dq offset sub_401550     ; DATA XREF: main+66↑o
.data:0000000000403040                                          ; main+6D↑r
.data:0000000000403048                 dq offset sub_401585
.data:0000000000403050                 dq offset sub_4015C5
.data:0000000000403058                 dq offset sub_401606
```

IDA View-A | Pseudocode-C | Pseudocode-B

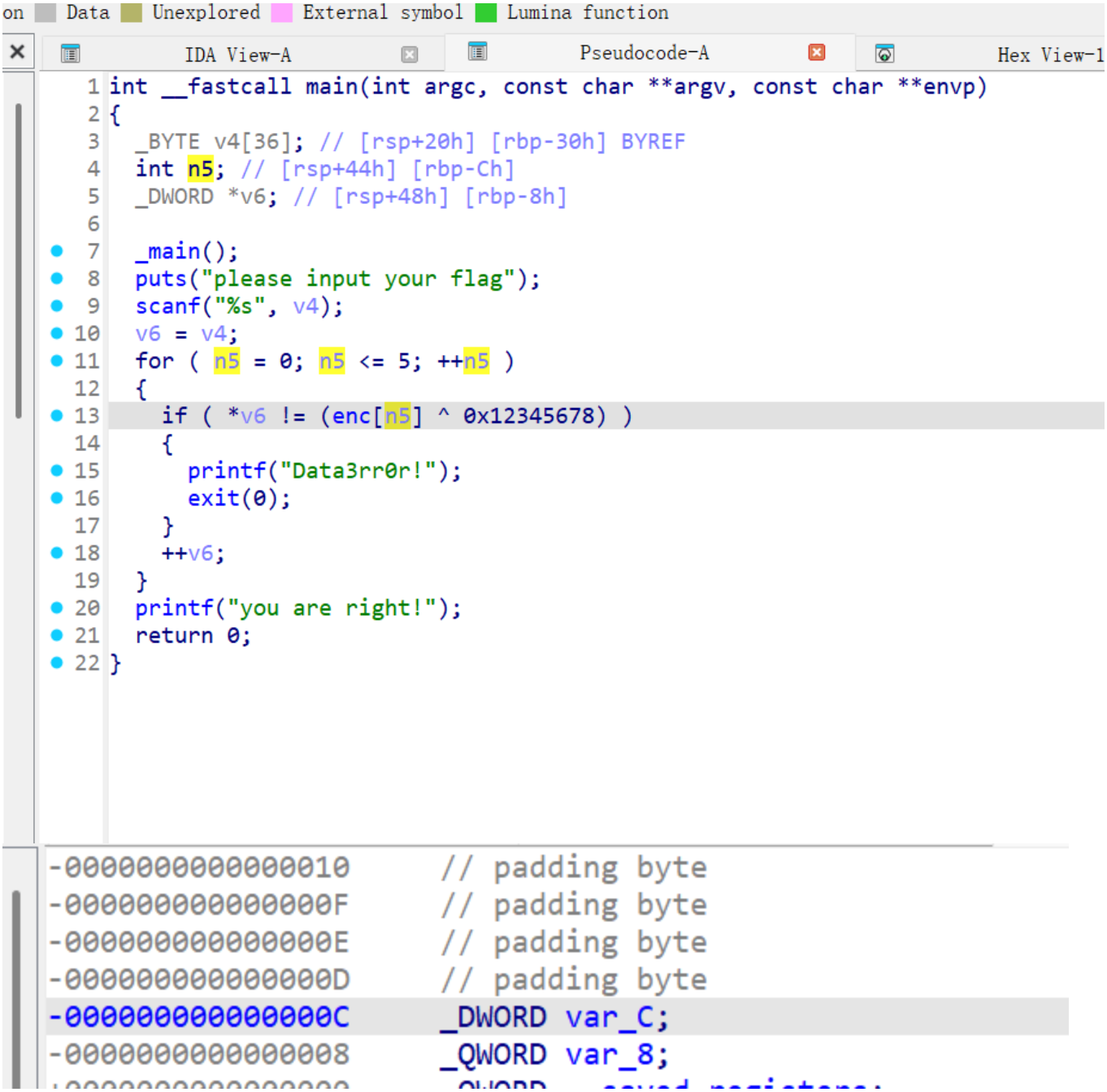
```
1  int __fastcall sub_401550(const char *a1)
2  {
3    size_t n58; // rax
4
5    n58 = strlen(a1);
6    if ( n58 != 58 )
7    {
8      LODWORD(n58) = puts("Wrong length!");
9      byte_403020 = 0;
10   }
11   return n58;
12 }
```

4,闹麻了,我是傻逼,查看string页面直接出来flag了



三,exe2

1,无壳,ida查看分析,单纯的异或,循环五次,异或五次,注意数组是dword形式,

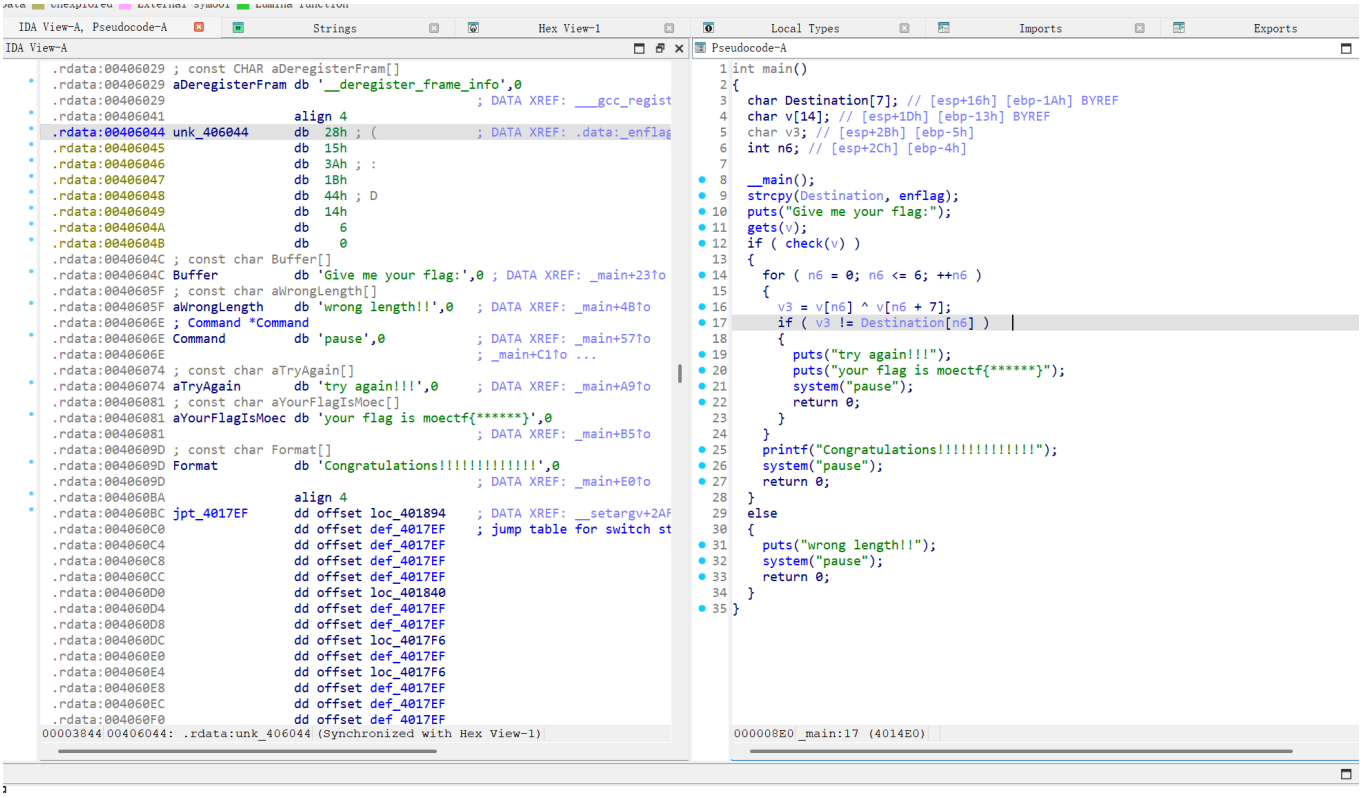


2,脚本如下,这个还挺好写的喵

```
enc=[0x51670536, 0x5E4F102C, 0x7E402211, 0x7C71094B, 0x7C553F1C,0x6F5A3816]
b=""
for i in range(len(enc)):
    a= enc[i] ^ 0x12345678
    b=a.to_bytes(4,byteorder='little').decode()
    print(b)
```

四,exe3

1,无壳,ida查看分析,提取出来密文,关键逻辑还是循环异或

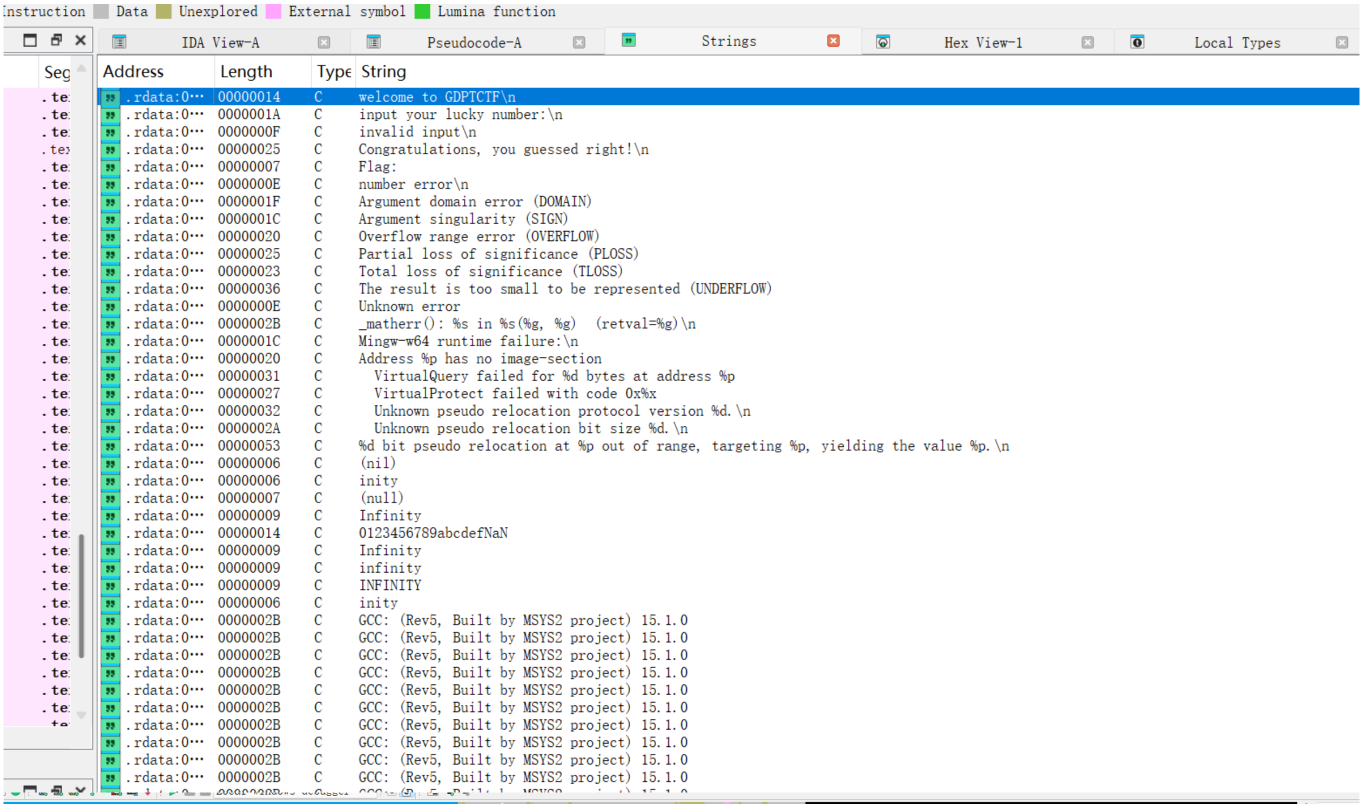


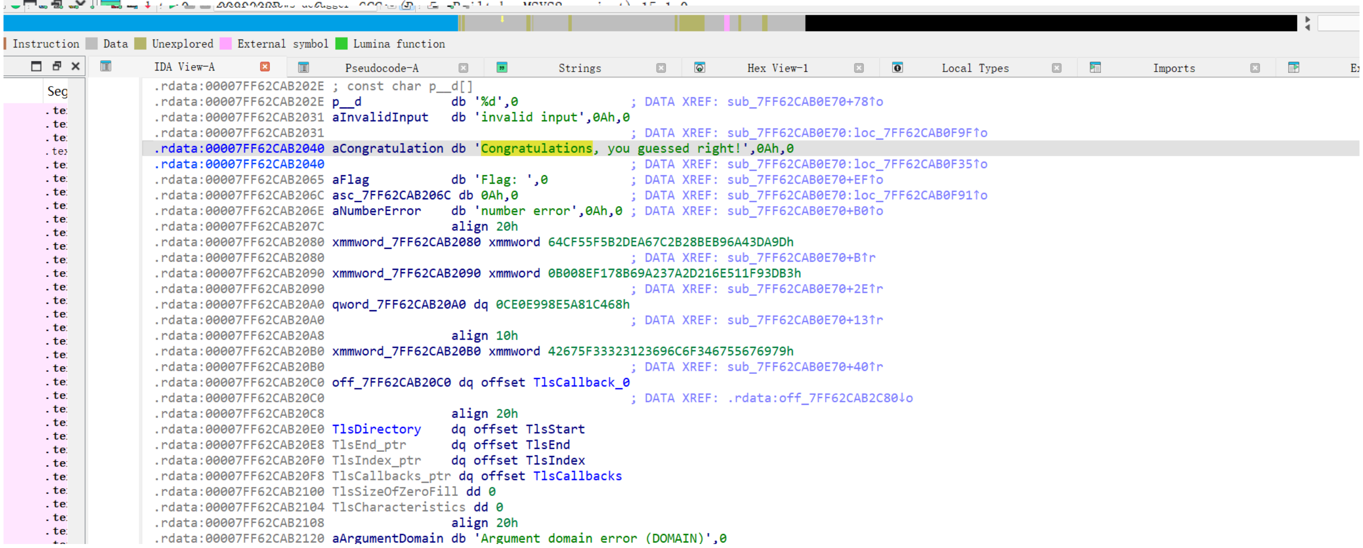
2,脚本如下

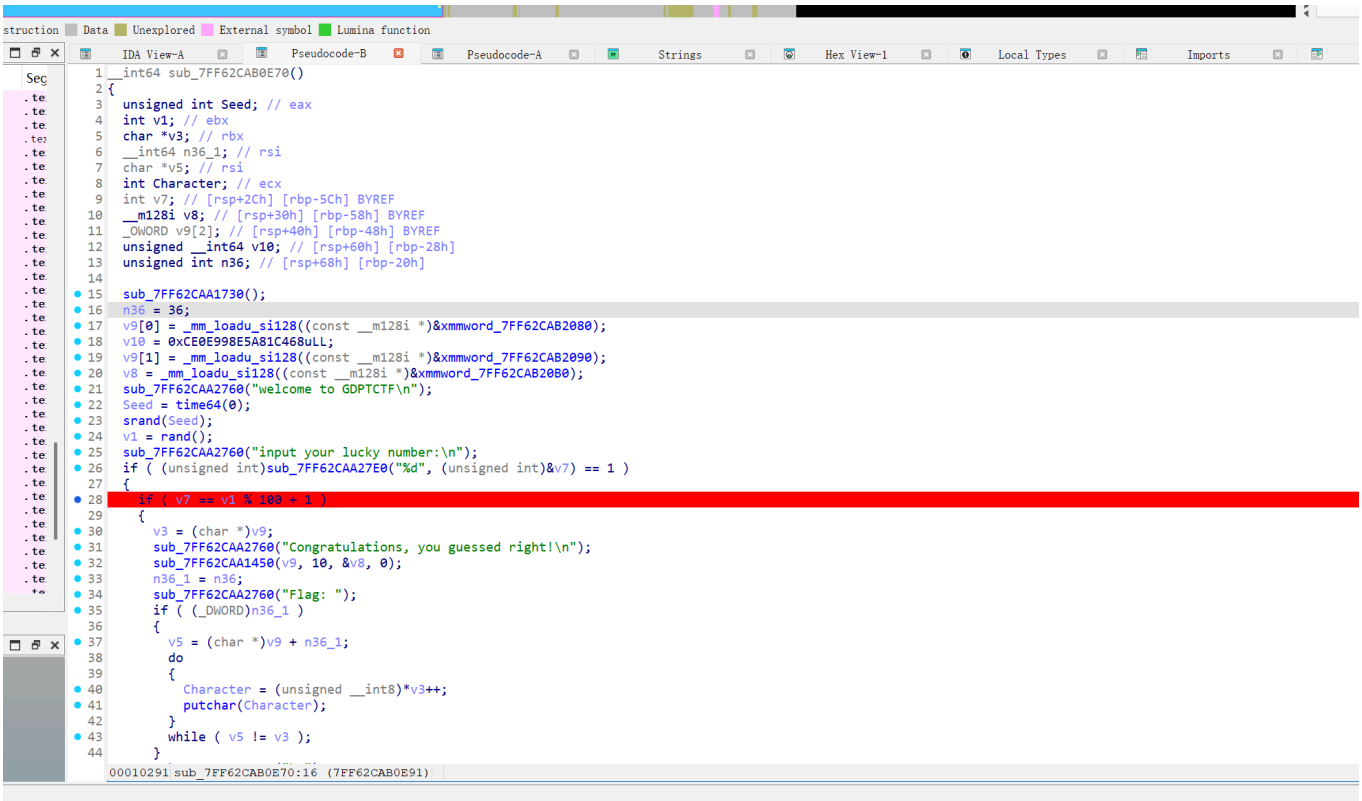
```
flag='moectf{'
enc=[0x28,0x15,0x3A,0x1B,0x44,0x14,0x6]
a=[]
for i in range(len(flag)):
    a.append(ord(flag[i]))

for i in range(len(enc)):
    a.append(enc[i])
print(a)
c=''
for i in range(len(a)):
    try:
        d=a[i]^a[i+7]
        c+=chr(d)
    except:
        break

print(flag+c)
```

四,调试1

1,无壳,ida查看发现没有main函数,于是去string里找一下,找到了但是发现需要动调,先设一个断点

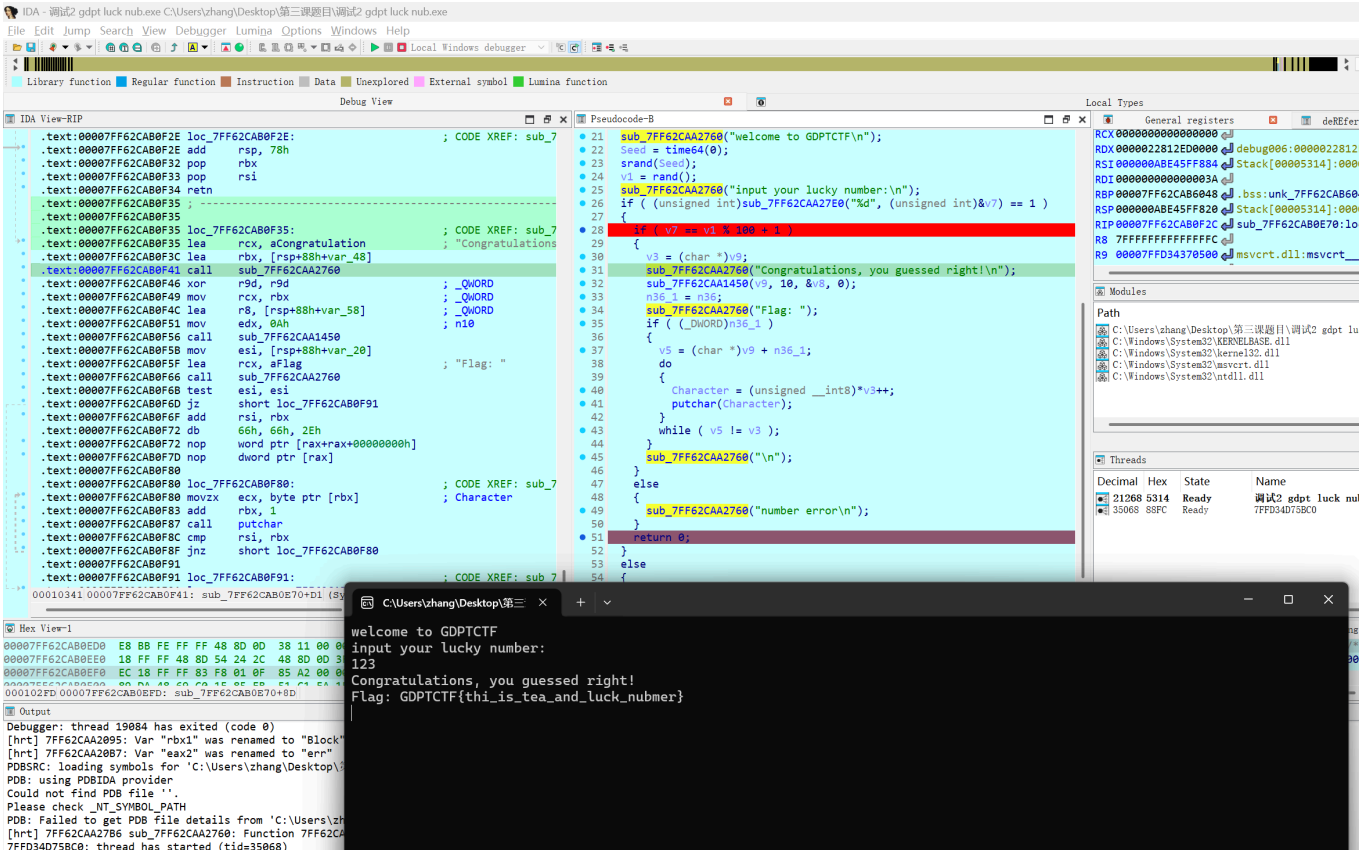






```
__int64 sub_7FF62CAB0E70()
{
  unsigned int Seed; // eax
  int v1; // ebx
  char *v3; // rbx
  __int64 n36_1; // rsi
  char *v5; // rsi
  int Character; // ecx
  int v7; // [rsp+2Ch] [rbp-5Ch] BYREF
  __m128i v8; // [rsp+30h] [rbp-58h] BYREF
  _OWORD v9[2]; // [rsp+40h] [rbp-48h] BYREF
  unsigned __int64 v10; // [rsp+60h] [rbp-28h]
  unsigned int n36; // [rsp+68h] [rbp-20h]

  sub_7FF62CAA1730();
  n36 = 36;
  v9[0] = _mm_loadu_si128((const __m128i *)&xmmword_7FF62CAB2080);
  v10 = 0xCE0E998E5A81C468uLL;
  v9[1] = _mm_loadu_si128((const __m128i *)&xmmword_7FF62CAB2090);
  v8 = _mm_loadu_si128((const __m128i *)&xmmword_7FF62CAB20B0);
  sub_7FF62CAA2760("welcome to GDPTCTF\n");
  Seed = time64(0);
  srand(Seed);
  v1 = rand();
  sub_7FF62CAA2760("input your lucky number:\n");
  if ( (unsigned int)sub_7FF62CAA27E0("%d", (unsigned int)&v7) == 1 )
  {
    if ( v7 == v1 % 100 + 1 )
    {
      v3 = (char *)v9;
      sub_7FF62CAA2760("Congratulations, you guessed right!\n");
      sub_7FF62CAA1450(v9, 10, &v8, 0);
      n36_1 = n36;
      sub_7FF62CAA2760("Flag: ");
      if ( (_DWORD)n36_1 )
      {
        v5 = (char *)v9 + n36_1;
        do
        {
          Character = (unsigned __int8)*v3++;
          putchar(Character);
        }
        while ( v5 != v3 );
      }
```

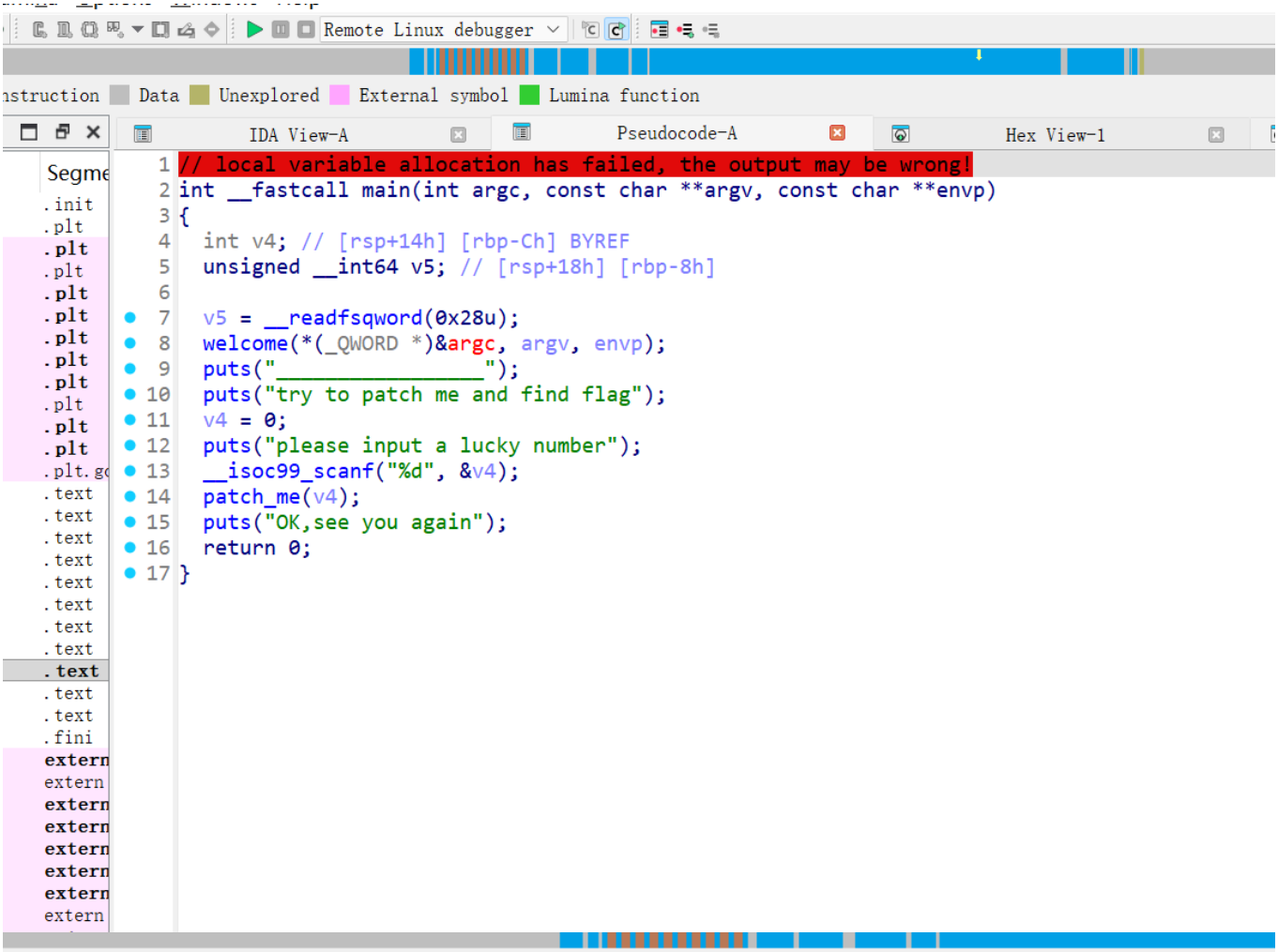
2,本地调试之后,改个ip,设置俩断点就行,但是改ip弄了一会,动调还是不熟练

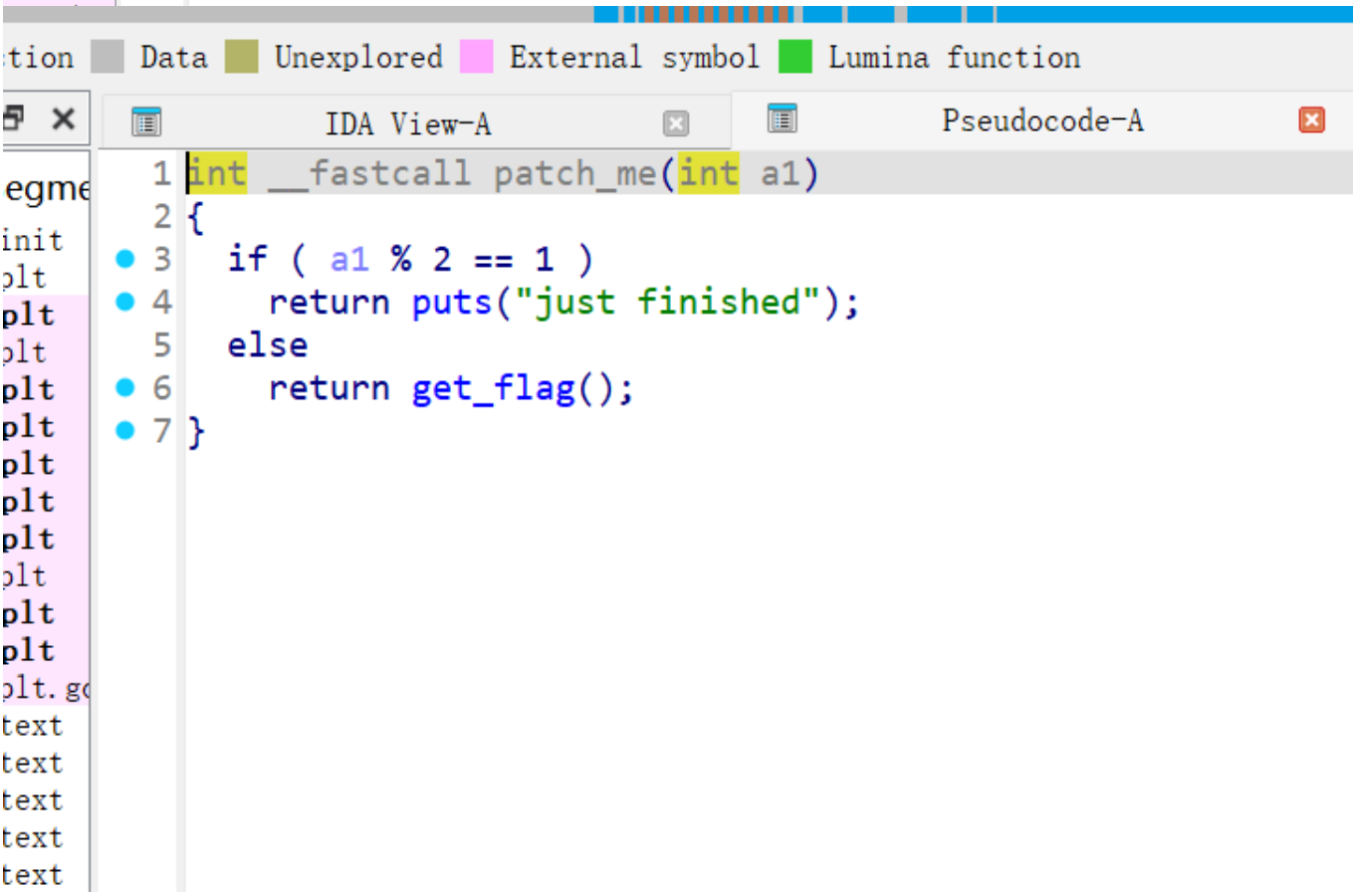


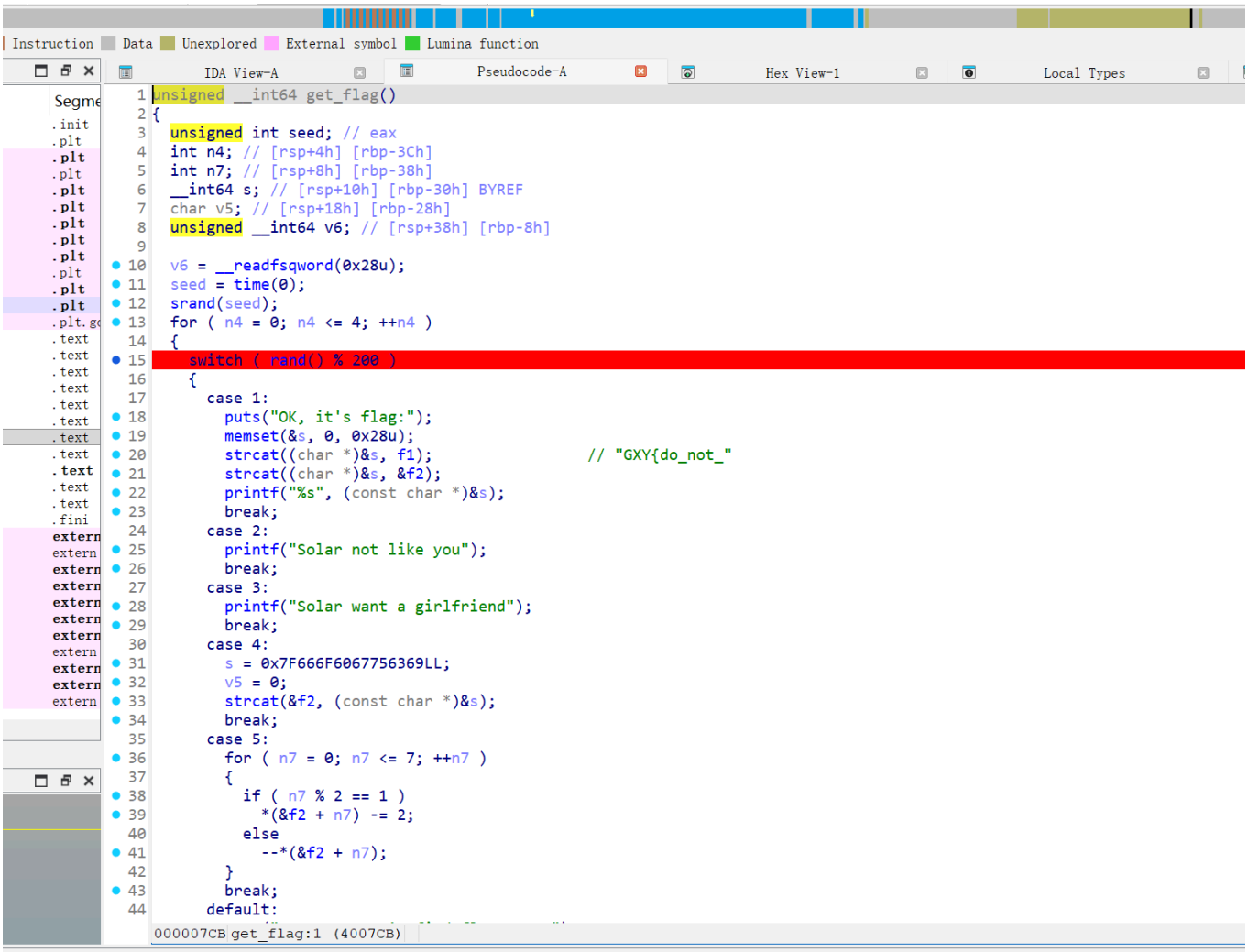
```
welcome to GDPTCTF
input your lucky number:
123
Congratulations, you guessed right!
Flag: GDPTCTF{thi_is_tea_and_luck_nubmer}
```

五,调试2

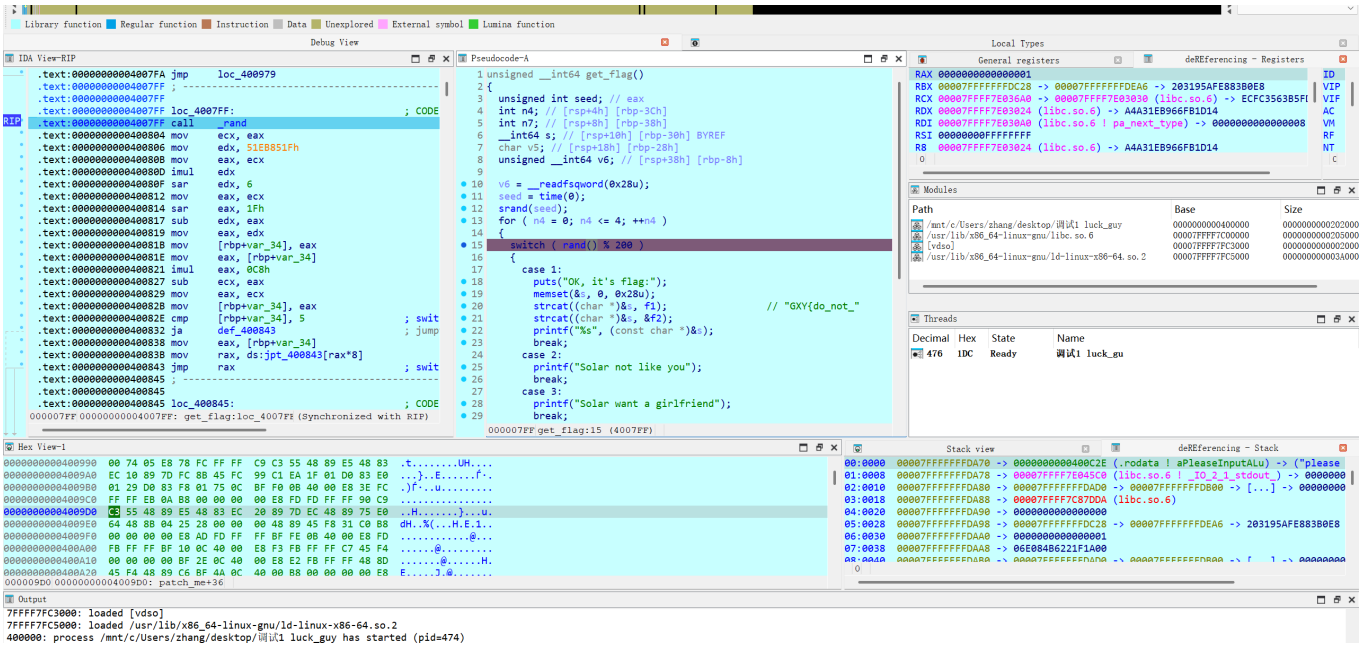
1,无壳,ida查看,还是要调试,由于是linux程序,我用wsl进行调试

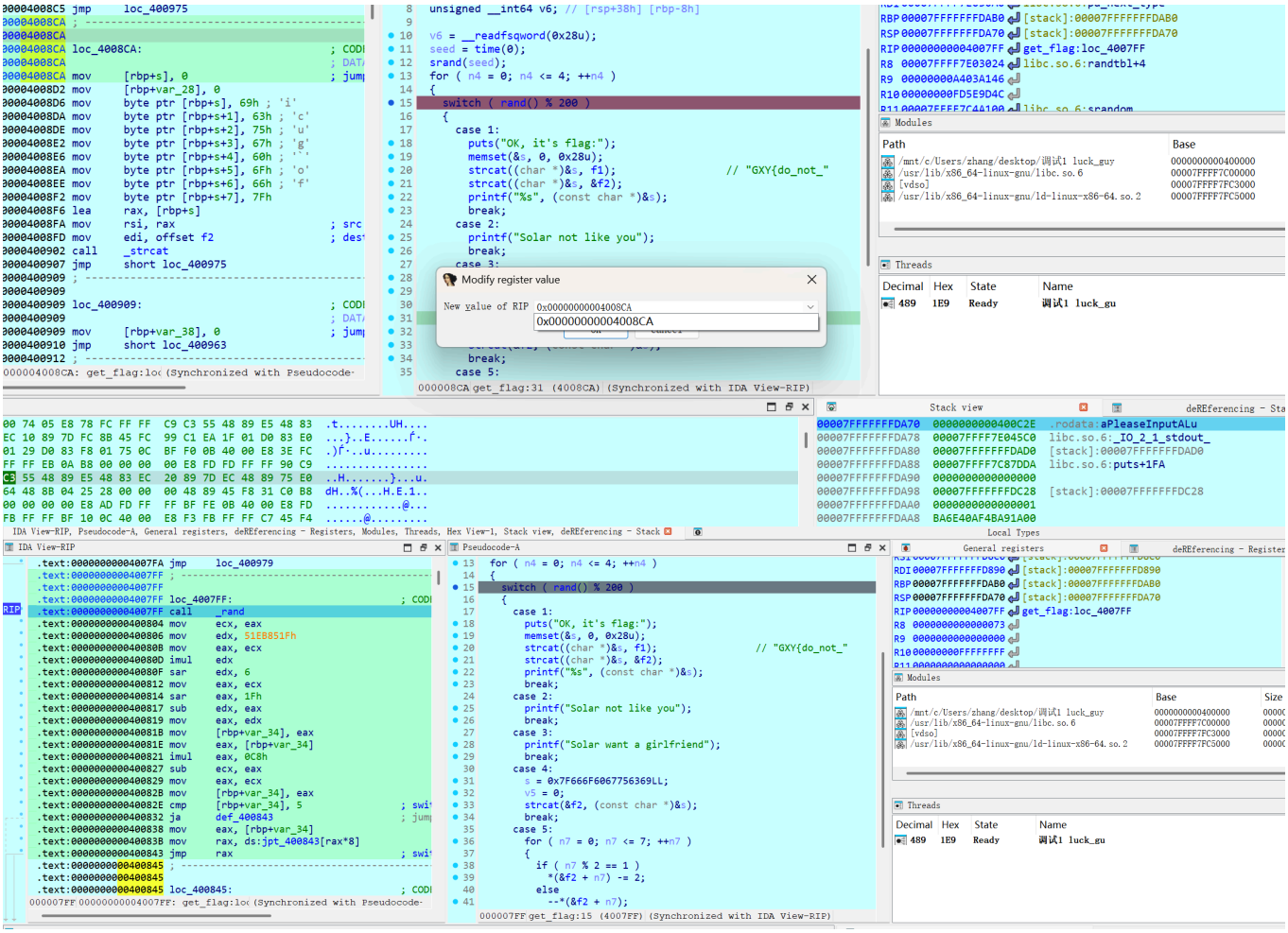






2,下断点,改ip就可以,改三次





3,flag出来啦

```
please input a lucky number
2
Looking for GNU DWARF file at "/usr/lib/debug/.build-id/38/f6b7066f73e3b41dca3ab5b6da405f8edf2ec5.debug"... no.
OK, it's flag:
GXY{do_not_hate_me}emmm,you can't find flag 23333
emmm,you can't find flag 23333
OK,see you again
2025-11-09 14:05:59 [6] Closing connection from ::ffff:172.24.48.1...
```