

逆向wp-5

逆向题目wp

张程思

—,maze

1,根据wasd这四个字母可以判断出来是迷宫问题

```

eg( 28     }
eg( 29     else if ( (unsigned __int8)buf_[i] > 0x64u )
eg( 30     {
eg( 31         if ( n119 == 's' )
eg( 32         {
eg( 33             if ( n209 > '\xD1' )
eg( 34                 goto LABEL_20;
eg( 35             n209 += 15;
eg( 36         }
eg( 37         else
eg( 38         {
eg( 39             if ( n119 != 'w' )
eg( 40             {
eg( 41 LABEL_21:
eg( 42                 j_puts(aInvalidInput);           // "Invalid input!"
eg( 43                 return -1;
eg( 44             }
eg( 45             if ( n209 <= 14 )
eg( 46                 goto LABEL_20;
eg( 47             n209 -= 15;
eg( 48         }
eg( 49     }
eg( 50     else
eg( 51     {
eg( 52         if ( n119 != 'a' )
eg( 53             goto LABEL_21;
eg( 54         if ( !(n209 % 15) )
eg( 55         {
eg( 56 LABEL_20:
eg( 57             j_puts(aInvalidMoveOut);           // "Invalid move: out of bounds!"
eg( 58             return -1;
eg( 59         }
eg( 60         --n209;
eg( 61     }
eg( 62     if ( asc_403020[n209] == 36 )                // "xxxxxxxxxxxx&&&&xxxxxxxx&&&&xxxxxxxx&&&&"
eg( 63     {
eg( 64         j_puts(aInvalidMoveHit);               // "Invalid move: hit a wall!"
eg( 65         return -1;
eg( 66     }
eg( 67     if ( asc_403020[n209] == 'y' )              // "xxxxxxxxxxxxxxxx&&&&xxxxxxxx&&&&xxxxxxxx&&&&"
eg( 68     {
eg( 69         j_puts(aYouWin);                       // "You win!"
eg( 70         j_puts(aPlzBasectfFlowe);             // "plz BaseCTF{lower.MD5{your path}} by 32bit"
eg( 71     }

```

```
appnetp.dll
nsvcrtd.dll
16)
24)
```

2,看一下string这里,找到了迷宫的大致

[illegible]

3,提取一下,自己在整理一下

 Export Plus: Export Data

Selected address

seg000:0000000000403020

Selected Length

0xE2

Selected Data

26 26 26 26 26 26 26 26 26 79 00

Data Type

String literal

Export As

String

Export Format:

Export Window

```
x$$$$$$$$$$$$$$$$&&&&&&$$$$$$$$$$$$&&$&$&&&&&$
&&$&$&$&&$&$&$&$&$&&$&$&$&$&$&$&$&$&$&$&$
&&$&$&$&&$&$&&&&&&&&&$&$&$&$&$&$&&&&$&$&$&$
&&$&$&$&$&$&$&&&$&&&$&$&$&&&&$&$&$&$&$&$&$
&&$&$&$&$&$&$&$&$&$&$&$&$&&&&&&y\x00
```

Line:1 Column:1

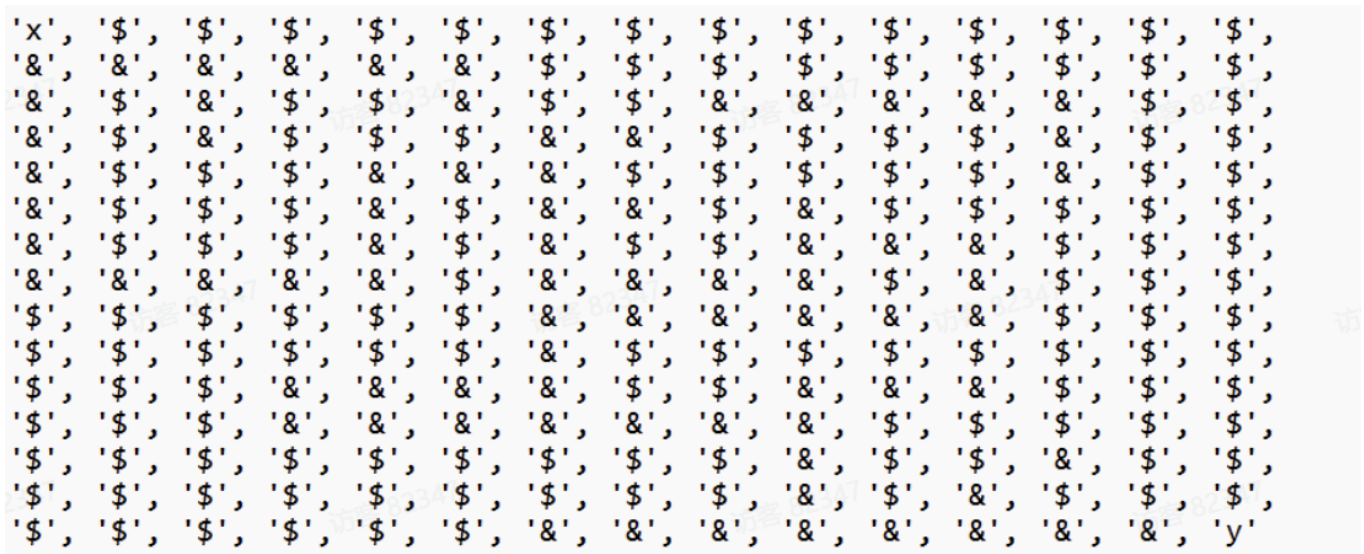
Export File Path

ez_maze (1)\export_results.txt

...

Export

Cancel



4,这里手动走一下也行，就是最短路径了，ssssssdddddwwddssssssdddddssssddddd
5,md5解一下就出来了

Last build: 2 years ago - Version 10 is here! Read about the new features here

Recipe

MD5

Input

ssssssdddddwwddssssssdddddssssddddd

Output

131b7d6e60e8a34cb01801ae8de07efe

STEP

BAKE!

Auto Bake

二,re1

1,这次找的题目好像有点水啊,我随便找的比赛

```
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     int p_n7; // [rsp+Ch] [rbp-34h] BYREF
4     char s[48]; // [rsp+10h] [rbp-30h] BYREF
5
6     strcpy(s, "flag{c0ngr@tul@ti0n_f0r_luck_numb3r}");
7     p_n7 = 0;
8     _isoc99_scanf(&unk_2004, &p_n7);
9     if ( p_n7 == 7 )
10         return puts(s);
11     else
12         return puts("sorry!");
13 }
```

三,re2

1,还原原本的字符串

```
IDA View-A Pseudocode-B Pseudocode-A Hex View-1 Local
1 v10[17] = 26;
2 v10[18] = 20;
3 v10[19] = 59;
4 v10[20] = 4;
5 v10[21] = 0;
6 printf("plz enter the flag:");
7 while ( 1 )
8 {
9     v7 = getch();
10    v12[v11] = v7;
11    if ( !v7 || v12[v11] == '\r' )
12        break;
13    if ( v12[v11] == '\b' )
14    {
15        printf("\b\b");
16        --v11;
17    }
18    else
19    {
20        printf("%c", (char)v12[v11++]);
21    }
22 }
23 v9 = 0;
24 for ( n17 = 0; n17 < 17; ++n17 )
25 {
26     if ( (char)v12[n17] != byte_415768[v10[n17]] )
27         v9 = 1;
28 }
29 if ( v12[17] != 49 || v12[18] != 48 || v12[19] != 50 || v12[20] != 52 || v12[21] != 125 )
30     v9 = 1;
31 v12[v11] = 0;
32 printf("\r\n");
33 if ( v9 )
34 {
35     printf("u r wrong\r\n\r\n");
36     main(argc_1, argv_1, envp_1);
37 }
38 else
39 {
40     printf("u r right!\r\n");
41 }
42 system("pause");
43 return 0;
44 }
```

000009AA _main_0:59 (4115AA)

```
a:00415767 align 4
a:00415768 ; char byte_415768[]
a:00415768 byte_415768 db 73h ; DATA XREF: _main_0+1E3↑r
a:00415769 aKfxEeftFGyrygt db 'KfxEeft}{gyrYgthtyhifsjei53UUrre_t2cdsef66246087138\0087138',0
a:004157A6 db 0
a:004157A7 db 0
a:004157A8 db 0
a:004157A9 db 0
a:004157AA db 0
a:004157AB db 0
a:004157AC db 0
a:004157AD db 0
a:004157AE db 0
a:004157AF db 0
a:004157B0 db 0
----- .. -
```

2,

```
data = "eKfxEeft}{gyrYgthtyhifsjei53UUrre_t2cdsef66246087138\0087138"
```

```
v10 = [1,4,14,10,5,36,23,42,13,19,28,13,27,39,48,41,42]
```

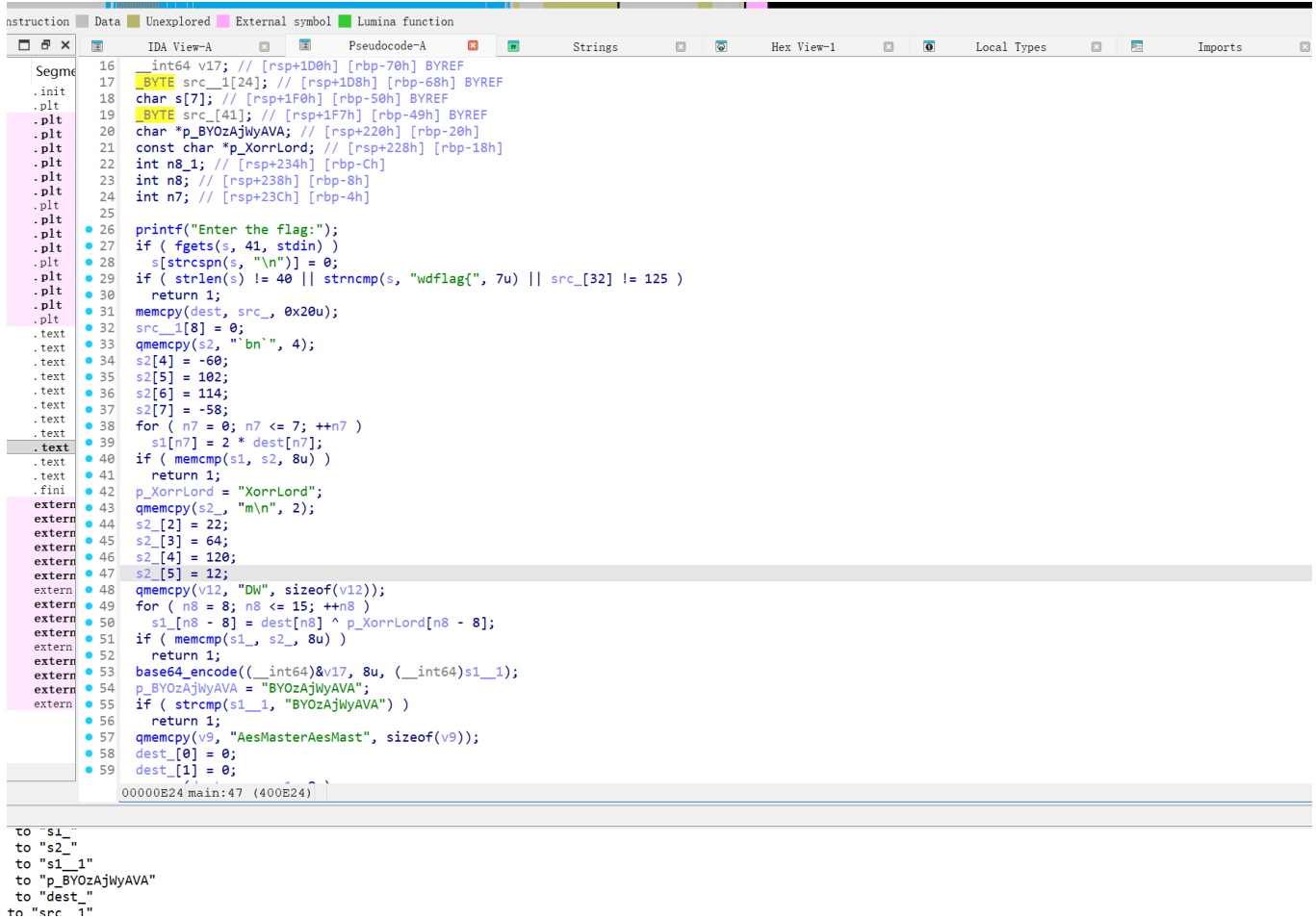
```
flag_part = ''.join([data[i] for i in v10])
```

```
full_flag = (f"{flag_part}1024}")
```

```
print(full_flag)
```

四,re3

1,题目给的提示是要闯关四个护卫,看了一下ida,应该就是四个加密,一个乘法,一个异或,一个base64,一个是AES加密,



```
16  int64 v17; // [rsp+1D0h] [rbp-70h] BYREF
17  BYTE src_1[24]; // [rsp+1D8h] [rbp-68h] BYREF
18  char s[7]; // [rsp+1F0h] [rbp-50h] BYREF
19  BYTE src_1[41]; // [rsp+1F7h] [rbp-49h] BYREF
20  char *p_BYOzAjWyAVA; // [rsp+220h] [rbp-20h]
21  const char *p_XorrLord; // [rsp+228h] [rbp-18h]
22  int n8_1; // [rsp+234h] [rbp-Ch]
23  int n8; // [rsp+238h] [rbp-8h]
24  int n7; // [rsp+23Ch] [rbp-4h]
25
26  printf("Enter the flag:");
27  if ( fgets(s, 41, stdin) )
28  {
29      s[strlen(s)] = 0;
30      if ( strlen(s) != 40 || strncmp(s, "wdf", 7) || src_1[32] != 125 )
31          return 1;
32      memcpy(dest, src_1, 0x20u);
33      src_1[8] = 0;
34      memcpy(s2, "\n", 4);
35      s2[4] = -60;
36      s2[5] = 102;
37      s2[6] = 114;
38      s2[7] = -58;
39      for ( n7 = 0; n7 <= 7; ++n7 )
40          s1[n7] = 2 * dest[n7];
41      if ( memcmp(s1, s2, 8u) )
42          return 1;
43      p_XorrLord = "XorrLord";
44      memcpy(s2, "m\n", 2);
45      s2[2] = 22;
46      s2[3] = 64;
47      s2[4] = 120;
48      s2[5] = 12;
49      memcpy(v12, "DW", sizeof(v12));
50      for ( n8 = 8; n8 <= 15; ++n8 )
51          s1[n8 - 8] = dest[n8] ^ p_XorrLord[n8 - 8];
52      if ( memcmp(s1, s2, 8u) )
53          return 1;
54      base64_encode((__int64)&v17, 8u, (__int64)s1_1);
55      p_BYOzAjWyAVA = "BYOzAjWyAVA";
56      if ( strcmp(s1_1, "BYOzAjWyAVA") )
57          return 1;
58      memcpy(v9, "AesMasterAesMast", sizeof(v9));
59      dest[0] = 0;
60      dest[1] = 0;
61
62  00000E24 main:47 (400E24) ^`
to "s1_"
to "s2_"
to "s1_1"
to "p_BYOzAjWyAVA"
to "dest_"
to "src_1"
```

2,第一二步脚本附上

```
memcpy(dest, v51, 0x20uLL);
```

```
v49 = 0;
```

```
s2 = 96;
```

```
v38 = 98;
```

```
v39 = 110;
```

```
v40 = 96;
```

```
v41 = -60;
```

```
v42 = 102;
```

```
v43 = 114;
```

```
v44 = -58;
```

```
for ( i = 0; i <= 7; ++i )
```

```
    s1[i] = 2 * dest[i];
```

```
96
```

```
=>
```

```
0x60
```

```
| 0x60 / 2 = 0x30 = 48 (ASCII '0')
```

```
98
```

```
=>
```

```
0x62
```

```
| 0x62 / 2 = 0x31 = 49 (ASCII '1')
```

```
110 =>
```

```
0x6E
```

```
| 0x6E / 2 = 0x37 = 55 (ASCII '7')
```

```
96
```

```
=>
```

```
0x60
```

```
| 0x60 / 2 = 0x30 = 48 (ASCII '0')
```

```
-60 =>
```

```
0xC4
```

```
| 0xC4 / 2 = 0x62 = 98
```

(ASCII 'b')

102 =>

0x66

| 0x66 / 2 = 0x33 = 51 (ASCII '3')

114 =>

0x72

| 0x72 / 2 = 0x39 = 57 (ASCII '9')

-58 =>

0xC6

| 0xC6 / 2 = 0x63 = 99

(ASCII 'c')

3,第三步用赛博厨子

ps://cyberchef.org/#recipe=From_Base64('CDEFGHIJKLMNOPQRSTUVWXYZABcdefghijklmnopqrstuvwxyz0123456789%2B/',true,false)&input=QlIPekFqV3lBVkE

sponsored by DEF24.com Last build: 2 years ago - Version 10 is here! Read about the new features here Options

Recipe

From Base64

Alphabet
CDEFGHIJKLMNOPQRST...
☒ Remove non-alphabet chars
☐ Strict mode

STEP

BAKE!

Auto Bake

Input

BYOzAjWYAVA

Output

ec3b52a6

4,第四步还是写脚本

```
qmemcpy(v9, "AesMasterAesMast", sizeof(v9));
```

```
v8[0] = 0LL;

v8[1] = 0LL;

memcpy(v8, v18, 8uLL);

for ( k = 8; k <= 15; ++k )

*((_BYTE *)v8 + k) = 8;

AES_set_encrypt_key(v9, 128LL, v6);

AES_encrypt(v8, v7, v6);

hex_to_string(v7, 16LL, v5);

v4[0] = -10;

v4[1] = -85;

v4[2] = 71;

v4[3] = -66;

v4[4] = 113;

v4[5] = -28;

v4[6] = 1;

v4[7] = -36;

v4[8] = 3;

v4[9] = 48;

v4[10] = -97;

v4[11] = -15;

v4[12] = 67;

v4[13] = -15;

v4[14] = -45;
```

```
v4[15] = 102;

if ( memcmp(v7, v4, 0x10uLL) )

return 1;

puts("Correct Flag!");

return 0;

}
```

然后再exp

```
from Crypto.Cipher import AES

from Crypto.Util.Padding import unpad

# 提供的数据 v4

v4 = bytes([0xF6, 0xAB, 0x47, 0xBE, 0x71, 0xE4, 0x01, 0xDC, 0x03, 0x30,

0x9F, 0xF1, 0x43, 0xF1, 0xD3, 0x66])

# 使用"AesMasterAesMast"作为密钥，编码为字节

key = "AesMasterAesMast".encode()

# 初始化AES解密器，选择ECB模式

cipher = AES.new(key, AES.MODE_ECB)

# 解密数据

decrypted_data = cipher.decrypt(v4)

# 去除填充

try:

unpadded_data = unpad(decrypted_data, AES.block_size)

# 输出解密后的数据，假设是ASCII字符串

print("Decrypted data:", unpadded_data.decode('utf-8'),
```

```

errors='ignore'))

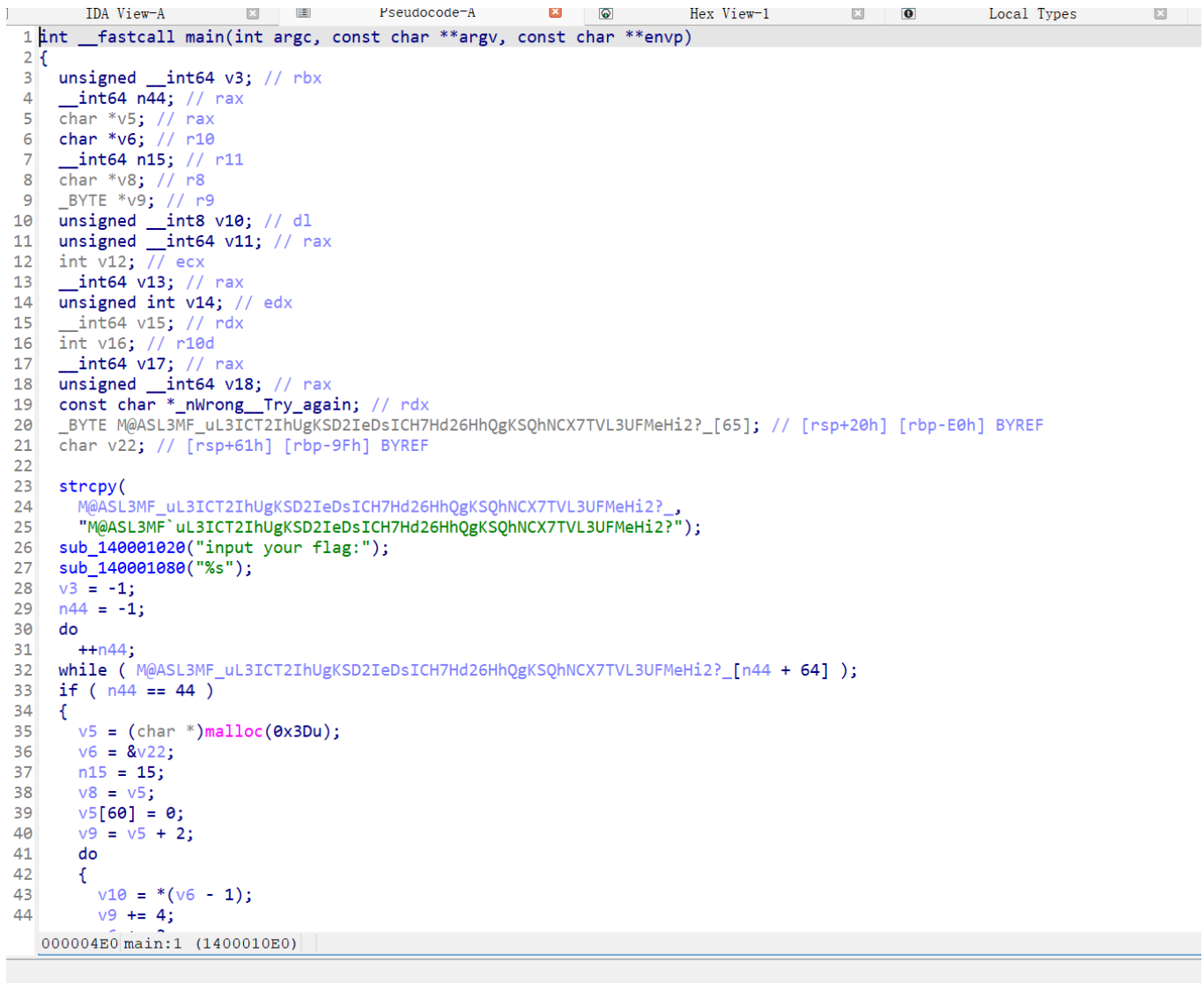
except ValueError as e:

print(f"Padding error or decryption issue: {e}")

```

五, re4

1, 先异或再base64换表



```

IDA View-A Pseudocode-A Hex View-1 Local Types
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     unsigned __int64 v3; // rbx
4     __int64 n44; // rax
5     char *v5; // rax
6     char *v6; // r10
7     __int64 n15; // r11
8     char *v8; // r8
9     _BYTE *v9; // r9
10    unsigned __int8 v10; // dl
11    unsigned __int64 v11; // rax
12    int v12; // ecx
13    __int64 v13; // rax
14    unsigned int v14; // edx
15    __int64 v15; // rdx
16    int v16; // r10d
17    __int64 v17; // rax
18    unsigned __int64 v18; // rax
19    const char *nWrong_Try_again; // rdx
20    _BYTE M@ASL3MF_uL3ICT2IhUgKSD2IeDsICH7Hd26HhQgKSQhNCX7TVL3UFMeHi2?_[65]; // [rsp+20h] [rbp-E0h] BYREF
21    char v22; // [rsp+61h] [rbp-9Fh] BYREF
22
23    strcpy(
24        M@ASL3MF_uL3ICT2IhUgKSD2IeDsICH7Hd26HhQgKSQhNCX7TVL3UFMeHi2?_,
25        "M@ASL3MF_uL3ICT2IhUgKSD2IeDsICH7Hd26HhQgKSQhNCX7TVL3UFMeHi2?");
26    sub_140001020("input your flag:");
27    sub_140001080("%s");
28    v3 = -1;
29    n44 = -1;
30    do
31        ++n44;
32    while ( M@ASL3MF_uL3ICT2IhUgKSD2IeDsICH7Hd26HhQgKSQhNCX7TVL3UFMeHi2?_[n44 + 64] );
33    if ( n44 == 44 )
34    {
35        v5 = (char *)malloc(0x3Du);
36        v6 = &v22;
37        n15 = 15;
38        v8 = v5;
39        v5[60] = 0;
40        v9 = v5 + 2;
41        do
42        {
43            v10 = *(v6 - 1);
44            v9 += 4;

```

000004E0 main:1 (1400010E0)

missions, class, and name

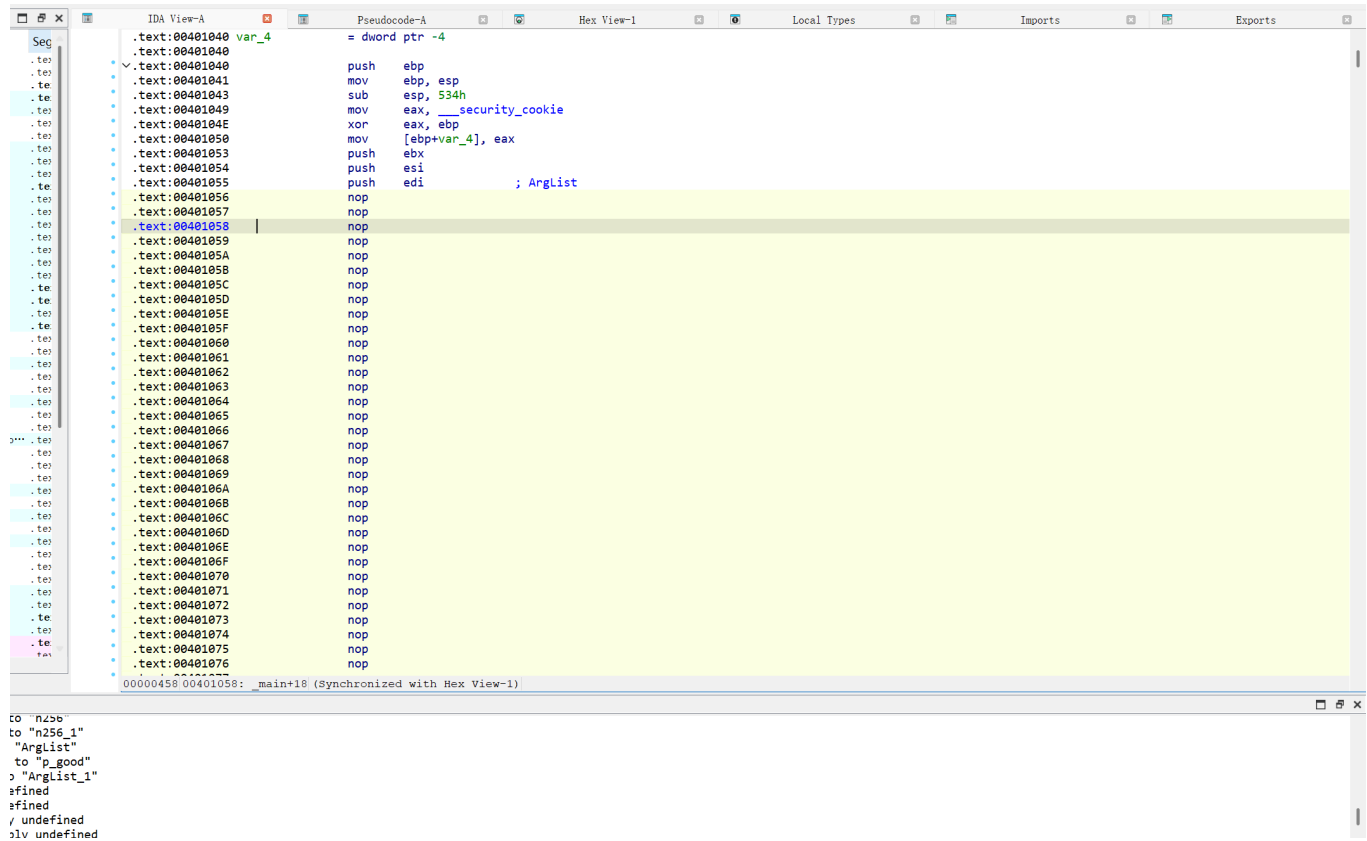

```
standard_b64 += '=' * (4 - len(standard_b64) % 4)
```

```
flag = base64.b64decode(standard_b64).decode()
```

```
print("Flag:", flag)
```

六,re5

1,初步打开发现无法反编译,我去了一下花就好了



```
Data Unexplored External symbol Lumina function
IDA View-A Pseudocode-B Pseudocode-A Hex View-1 Local Types

1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3     int v3; // ecx
4     const char *v4; // esi
5     unsigned int v5; // esi
6     unsigned int v6; // kr00_4
7     int n256; // ecx
8     int n256_1; // edi
9     int v9; // ebx
10    unsigned __int8 v10; // d1
11    int v11; // edi
12    int v12; // ebx
13    unsigned __int8 v13; // d1
14    int v14; // ecx
15    unsigned int i; // ecx
16    char *p_good; // eax
17    char ArgList; // [esp+0h] [ebp-540h]
18    char ArgList_1; // [esp+0h] [ebp-540h]
19    _BYTE v20[44]; // [esp+Ch] [ebp-534h]
20    int v21; // [esp+38h] [ebp-508h]
21    _BYTE v22[512]; // [esp+3Ch] [ebp-504h]
22    _BYTE v23[256]; // [esp+23Ch] [ebp-304h] BYREF
23    char v24[512]; // [esp+33Ch] [ebp-204h] BYREF
24
25    v5 = (unsigned int)&v4[strlen(v4) + 1 - v3];
26    v6 = strlen(v24);
27    memset(v23, 0, sizeof(v23));
28    for ( n256 = 0; n256 < 256; ++n256 )
29    {
30        v24[n256 + 256] = n256;
31        v23[n256] = v24[n256 % v6];
32    }
33    n256_1 = 0;
34    v9 = 0;
35    do
36    {
37        v10 = v24[n256_1 + 256];
38        v9 = (v9 + (char)v23[n256_1] + v10) % 256;
39        v24[n256_1++ + 256] = v24[v9 + 256];
40        v24[v9 + 256] = v10 ^ 0x37;
41    }
42    while ( n256_1 < 256 );
43    sub_401010("\n\n", ArgList);
44    v11 = 0;
45    return 0;
46 }
```

```
1"
2"
3od"
```

2,一个rc4,但是中间有异或,应该是魔改rc4

```
27 memset(v23, 0, 256 * (v25 / 256));
28 for ( n256 = 0; n256 < 256; ++n256 )
29 {
30     v24[n256 + 256] = n256;
31     v23[n256] = v24[n256 % v6];
32 }
33 n256_1 = 0;
34 v9 = 0;
35 do
36 {
37     v10 = v24[n256_1 + 256];
38     v9 = (v9 + (char)v23[n256_1] + v10) % 256;
39     v24[n256_1++ + 256] = v24[v9 + 256];
40     v24[v9 + 256] = v10 ^ 0x37;
41 }
42 while ( n256_1 < 256 );
43 sub_401010("\n\n", ArgList);
44 v11 = 0;
45 v21 = 0;
46 v12 = 0;
47 if ( v5 )
48 {
49     do
50     {
51         v11 = (v11 + 1) % 256;
52         v13 = v24[v11 + 256];
53         v12 = (v13 + v12) % 256;
54         v24[v11 + 256] = v24[v12 + 256];
55         v24[v12 + 256] = v13;
56         v14 = v21;
57         v22[v21] ^= v24[(unsigned __int8)(v13 + v24[v11 + 256]) + 256];
58         v21 = v14 + 1;
59     }
60     while ( v14 + 1 < v5 );
61     v12 = 0;
62 }
63 for ( i = 0; i < v5; ++i )
64     v12 = v22[i] == v20[i];
65 p_good = (char *)&unk_402184;
66 if ( v12 == 1 )
67     p_good = aGood;
68 sub_401010(p_good, ArgList_1);
69 return 0;
70 }
```

0000054D_main:27 (40114D)

3.patch一下

Pseudocode-C

```
22 int v23; // [esp+38h] [ebp-508h]
23 char v24[512]; // [esp+3Ch] [ebp-504h] BYREF
24 char v25[256]; // [esp+23Ch] [ebp-304h] BYREF
25 char tallmewhy[256]; // [esp+33Ch] [ebp-204h] BYREF
26 _BYTE v27[256]; // [esp+43Ch] [ebp-104h] BYREF
27
28 memset(v27, 0, sizeof(v27));
```

Execute script

Snippet list

Name

Default snippet

Line 1 of 1

Please enter script body

```
1 add=0x58F6B0
2 enc=[0xF5, 0x8C, 0x8D, 0xE4, 0x9F, 0xA5, 0x28,
3 0xEB, 0xD3, 0x24, 0xA9, 0x91, 0x1A, 0x6F, 0xD4,
4 0x6A, 0xD7,
5 0x0B, 0x8D, 0xE8, 0xB8, 0x83, 0x4A, 0x5A, 0x6E,
6 0xBE, 0xCB,
7 0xF4, 0x4B, 0x99, 0xD6, 0xE6, 0x54, 0x7A, 0x4F,
```

Line:1 Column:13

Scripting language

Python

Tab size

4

Run

Export

Import

61 (Synchronized with Pseudocode-B)

```
45 v27[n256] = n256;
46 v25[n256] = tallmewhy[n256 % v4];
47 }
```

n256_1 = 0

0000050F_main:39 (CB110F)

4,最后的循环之前下个断点,找到密文flag就在旁边了

The screenshot displays the IDA Pro interface with three main panels:

- Stack View-EIP:** Shows a list of stack frames. The frame at address 00005FF4 is expanded, showing various data types (db, dw, dd) and their values. The current instruction pointer (EIP) is 00005F52B, which is synchronized with the pseudocode.
- Pseudocode-A:** Displays the assembly code for the current function. The code includes a loop that increments a counter and checks a condition. The current instruction is at address 00000670, which is labeled as `_main:79 (CB1270)`.
- Hex View-1:** Shows the raw hex data of the memory. The data is organized into columns, with the first column showing the address (e.g., 00CB1000, 00CB1010, etc.) and the second column showing the hex bytes (e.g., B8 80 33 CB 00 C3 CC CC). The data is also displayed in a human-readable format (e.g., ..3....., U....u.j.....).

The pseudocode shows a loop that increments a counter and checks a condition. The current instruction is at address 00000670, which is labeled as `_main:79 (CB1270)`.