WOW

```
.text:00561BCC
.text:00561BCC loc_561BCC:
.text:00561BCC cmp     [ebp+var_4], 4
.text:00561BD0 jge     short loc_561C18
```

循环 4 次，每次取 8 位
FLAG 长度 32

加密语句

```
sub_F31AC0(a40s, (char)v10);
for ( i = 0; i < 4; ++i )
{
  sub_F31850(&v10[8 * i], v5);
  sub_F31410(v5, v7);
  sub_F318D0(v7, &Buf2[i]);
}
sub_F319B0((char *)&loc_F31C73 + 51);
sub_F319B0(&loc_F31C73);
if ( !memcmp(&unk_F34D40, Buf2, 0x20u) )
  sub_F31A50(aYouWin, v4[0]);
else
  sub_F31A50(aError, v4[0]);
```

For 循环有三个加密函数
第一个

```
int __cdecl sub_F31850(int a1, int a2)
{
  int result; // eax
  int j; // [esp+4h] [ebp-8h]
  int i; // [esp+8h] [ebp-4h]

  for ( i = 0; i < 8; ++i )
  {
    for ( j = 0; j < 8; ++j )
      *(_BYTE *)(a2 + j + 8 * i) = ((*(char *)(i + a1) >> j) & 1) != 0;
    result = i + 1;
  }
  return result;
}
```

第二个

```
for ( i = 0; i < 64; ++i )
{
  v4[i + 32] = *(_BYTE *)(a1 + dword_F34820[i] - 1);
  result = i + 1;
}
for ( i = 0; i < 32; ++i )
{
  v4[i + 96] = v4[i + 32];
  result = i + 1;
}
for ( i = 0; i < 32; ++i )
{
  result = i;
  v5[i] = v4[i + 64];
}
for ( j = 0; j < 16; ++j )
{
  for ( i = 0; i < 32; ++i )
    v3[i] = v5[i];
  result = sub_F31000(v5, &byte_F35200[48 * j], v4);
  for ( i = 0; i < 32; ++i )
  {
    v7 = v4[i + 96] != v4[i];
    v6 = v7;
    result = i;
    v5[i] = v6;
  }
  for ( i = 0; i < 32; ++i )
  {
    result = i;
    v4[i + 96] = v3[i];
```

```
  }
  for ( i = 0; i < 32; ++i )
  {
    v4[i + 32] = v5[i];
    result = i + 1;
  }
  for ( i = 0; i < 32; ++i )
  {
    result = i;
    v4[i + 64] = v4[i + 96];
  }
  for ( i = 0; i < 64; ++i )
  {
    result = v4[dword_F34C40[i] + 31];
    *(_BYTE *)(i + a2) = result;
  }
  return result;
}
```

第二个加密函数内部还有一个加密函数

```
  for ( i = 0; i < 48; ++i )
  {
    v4[i] = *(_BYTE *)(a1 + dword_F34A60[i] - 1);
    result = i + 1;
  }
  for ( i = 0; i < 48; ++i )
  {
    v9 = v4[i] != *(_BYTE *)(i + a2);
    v8 = v9;
    v4[i] = v8;
    result = i + 1;
  }
  for ( i = 0; i < 8; ++i )
  {
    v12 = 6 * i;
    v6 = (unsigned __int8)v4[6 * i + 5] + 2 * (unsigned __int8)v4[6 * i];
    v5 = (unsigned __int8)v4[6 * i + 4]
       + 4 * (unsigned __int8)v4[6 * i + 2]
       + 8 * (unsigned __int8)v4[6 * i + 1]
       + 2 * (unsigned __int8)v4[6 * i + 3];
    result = v5;
    v10 = dword_F34020[64 * i + 16 * v6 + v5];
    v12 = 4 * i;
    for ( j = 3; j >= 0; --j )
    {
      v7 = v10 % 2 != 0;
      v4[j + 48 + v12] = v7;
      v10 /= 2;
      result = j - 1;
    }
  }
  for ( i = 0; i < 32; ++i )
  {
    result = dword_F34920[i];
    *(_BYTE *)(i + a3) = v4[result + 47];
  }
  return result;
}
```

第三个

```
  for ( i = 0; i < 8; ++i )
  {
    for ( j = 0; j < 8; ++j )
      *(_BYTE *)(i + a2) |= *(_BYTE *)(a1 + j + 8 * i) << j;
    result = i + 1;
  }
  return result;
}
```

编写c代码

```
#include<stdio.h>
#include<math.h>
void dealv4(int k);
int v4[128], v5[32], v3[32];

int data2[48] =
{ 0x20,0x1,0x2,0x3,0x4,0x5,0x4,0x5,0x6,0x7,0x8,0x9,0x8,0x9,0x0A,0x0B,0x0C,0x0D,0x0C,0x0
D,0x0E,0x0F,0x10,0x11,0x10,0x11,0x12,0x13,0x14,0x15,0x14,0x15,0x16,0x17,0x18,0x19,0x18,
0x19,0x1A,0x1B,0x1C,0x1D,0x1C,0x1D,0x1E,0x1F,0x20,0x1 };
int data3[768];
int data4[512] =
{ 0x0E,0x4,0x0D,0x1,0x2,0x0F,0x0B,0x8,0x3,0x0A,0x6,0x0C,0x5,0x9,0x0,0x7,0x0,0x0F,0x7,0x
4,0x0E,0x2,0x0D,0x1,0x0A,0x6,0x0C,0x0B,0x9,0x5,0x3,0x8,0x4,0x1,0x0E,0x8,0x0D,0x6,0x2,0x
0B,0x0F,0x0C,0x9,0x7,0x3,0x0A,0x5,0x0,0x0F,0x0C,0x8,0x2,0x4,0x9,0x1,0x7,0x5,0x0B,0x3,0x
0E,0x0A,0x0,0x6,0x0D,0x0F,0x1,0x8,0x0E,0x6,0x0B,0x3,0x4,0x9,0x7,0x2,0x0D,0x0C,0x0,0x5,0
x0A,0x3,0x0D,0x4,0x7,0x0F,0x2,0x8,0x0E,0x0C,0x0,0x1,0x0A,0x6,0x9,0x0B,0x5,0x0,0x0E,0x7,
0x0B,0x0A,0x4,0x0D,0x1,0x5,0x8,0x0C,0x6,0x9,0x3,0x2,0x0F,0x0D,0x8,0x0A,0x1,0x3,0x0F,0x4
,0x2,0x0B,0x6,0x7,0x0C,0x0,0x5,0x0E,0x9,0x0A,0x0,0x9,0x0E,0x6,0x3,0x0F,0x5,0x1,0x0D,0x0
C,0x7,0x0B,0x4,0x2,0x8,0x0D,0x7,0x0,0x9,0x3,0x4,0x6,0x0A,0x2,0x8,0x5,0x0E,0x0C,0x0B,0x0
F,0x1,0x0D,0x6,0x4,0x9,0x8,0x0F,0x3,0x0,0x0B,0x1,0x2,0x0C,0x5,0x0A,0x0E,0x7,0x1,0x0A,0x
0D,0x0,0x6,0x9,0x8,0x7,0x4,0x0F,0x0E,0x3,0x0B,0x5,0x2,0x0C,0x7,0x0D,0x0E,0x3,0x0,0x6,0x
9,0x0A,0x1,0x2,0x8,0x5,0x0B,0x0C,0x4,0x0F,0x0D,0x8,0x0B,0x5,0x6,0x0F,0x0,0x3,0x4,0x7,0x
2,0x0C,0x1,0x0A,0x0E,0x9,0x0A,0x6,0x9,0x0,0x0C,0x0B,0x7,0x0D,0x0F,0x1,0x3,0x0E,0x5,0x2,
0x8,0x4,0x3,0x0F,0x0,0x6,0x0A,0x1,0x0D,0x8,0x9,0x4,0x5,0x0B,0x0C,0x7,0x2,0x0E,0x2,0x0C,
0x4,0x1,0x7,0x0A,0x0B,0x6,0x8,0x5,0x3,0x0F,0x0D,0x0,0x0E,0x9,0x0E,0x0B,0x2,0x0C,0x4,0x7
,0x0D,0x1,0x5,0x0,0x0F,0x0A,0x3,0x9,0x8,0x6,0x4,0x2,0x1,0x0B,0x0A,0x0D,0x7,0x8,0x0F,0x9
,0x0C,0x5,0x6,0x3,0x0,0x0E,0x0B,0x8,0x0C,0x7,0x1,0x0E,0x2,0x0D,0x6,0x0F,0x0,0x9,0x0A,0x
4,0x5,0x3,0x0C,0x1,0x0A,0x0F,0x9,0x2,0x6,0x8,0x0,0x0D,0x3,0x4,0x0E,0x7,0x5,0x0B,0x0A,0x
0F,0x4,0x2,0x7,0x0C,0x9,0x5,0x6,0x1,0x0D,0x0E,0x0,0x0B,0x3,0x8,0x9,0x0E,0x0F,0x5,0x2,0x
8,0x0C,0x3,0x7,0x0,0x4,0x0A,0x1,0x0D,0x0B,0x6,0x4,0x3,0x2,0x0C,0x9,0x5,0x0F,0x0A,0x0B,0
x0E,0x1,0x7,0x6,0x0,0x8,0x0D,0x4,0x0B,0x2,0x0E,0x0F,0x0,0x8,0x0D,0x3,0x0C,0x9,0x7,0x5,0
x0A,0x6,0x1,0x0D,0x0,0x0B,0x7,0x4,0x9,0x1,0x0A,0x0E,0x3,0x5,0x0C,0x2,0x0F,0x8,0x6,0x1,0
x4,0x0B,0x0D,0x0C,0x3,0x7,0x0E,0x0A,0x0F,0x6,0x8,0x0,0x5,0x9,0x2,0x6,0x0B,0x0D,0x8,0x1,
0x4,0x0A,0x7,0x9,0x5,0x0,0x0F,0x0E,0x2,0x3,0x0C,0x0D,0x2,0x8,0x4,0x6,0x0F,0x0B,0x1,0x0A
,0x9,0x3,0x0E,0x5,0x0,0x0C,0x7,0x1,0x0F,0x0D,0x8,0x0A,0x3,0x7,0x4,0x0C,0x5,0x6,0x0B,0x0
,0x0E,0x9,0x2,0x7,0x0B,0x4,0x1,0x9,0x0C,0x0E,0x2,0x0,0x6,0x0A,0x0D,0x0F,0x3,0x5,0x8,0x2
,0x1,0x0E,0x7,0x4,0x0A,0x8,0x0D,0x0F,0x0C,0x9,0x0,0x3,0x5,0x6,0x0B };
int data5[32] =
{ 0x10,0x7,0x14,0x15,0x1D,0x0C,0x1C,0x11,0x1,0x0F,0x17,0x1A,0x5,0x12,0x1F,0x0A,0x2,0x8,
0x18,0x0E,0x20,0x1B,0x3,0x9,0x13,0x0D,0x1E,0x6,0x16,0x0B,0x4,0x19 };
int main()
{
```

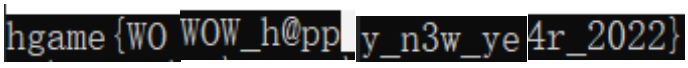
```
    int buf2[32] =
{ 0x29,0x20,0x98,0x62,0x0FC,0x46,0x79,0x56,0x0A5,0x90,0x4A,0x0F6,0x0A3,0x70,0x0D9,0x1B,
0x34,0x2,0x1B,0x50,0x65,0x86,0x33,0x35,0x0BB,0x0A5,0x9F,0x96,0x0BB,0x2B,0x0E3,0x2A };
    int input[8] = { 0 };
    int n;
    int output1[64];
    int boo;
    int data0[64] =
{ 0x3A,0x32,0x2A,0x22,0x1A,0x12,0x0A,0x2,0x3C,0x34,0x2C,0x24,0x1C,0x14,0x0C,0x4,0x3E,0x
36,0x2E,0x26,0x1E,0x16,0x0E,0x6,0x40,0x38,0x30,0x28,0x20,0x18,0x10,0x8,0x39,0x31,0x29,0
x21,0x19,0x11,0x9,0x1,0x3B,0x33,0x2B,0x23,0x1B,0x13,0x0B,0x3,0x3D,0x35,0x2D,0x25,0x1D,0
x15,0x0D,0x5,0x3F,0x37,0x2F,0x27,0x1F,0x17,0x0F,0x7 };
    int data1[64] =
{ 0x28,0x8,0x30,0x10,0x38,0x18,0x40,0x20,0x27,0x7,0x2F,0x0F,0x37,0x17,0x3F,0x1F,0x26,0x
6,0x2E,0x0E,0x36,0x16,0x3E,0x1E,0x25,0x5,0x2D,0x0D,0x35,0x15,0x3D,0x1D,0x24,0x4,0x2C,0x
0C,0x34,0x14,0x3C,0x1C,0x23,0x3,0x2B,0x0B,0x33,0x13,0x3B,0x1B,0x22,0x2,0x2A,0x0A,0x32,0
x12,0x3A,0x1A,0x21,0x1,0x29,0x9,0x31,0x11,0x39,0x19 };
    int i, j;
    int  output0[64];
    int A[768] =
{ 0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,1,1,1,1,1,1,0,0,0,1,0,1,1,1,1,1,1,1,1,1,1,0,1,0,0,0,0,0,0
,0,0,0,0,0,0,0,0,1,0,0,0,0,1,0,1,1,1,1,1,1,1,1,0,1,0,0,0,1,0,1,0,1,0,0,0,0,0,1,0,0,1,0,1,
0,1,0,0,0,0,0,0,0,0,0,1,1,0,1,0,1,0,0,1,1,0,1,0,1,1,0,1,0,1,1,0,1,0,1,0,0,0,0,0,0,0,0,0
,1,1,1,1,1,1,1,1,1,1,1,0,0,1,0,1,0,0,0,0,0,0,0,1,1,1,1,1,1,1,0,0,0,0,0,0,1,1,0,1,0,1,1,0,
1,0,1,1,0,1,1,1,1,1,0,0,0,0,0,0,0,1,0,1,1,0,1,0,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,1,0,0,1,0,1
,0,1,1,0,0,0,0,1,1,1,0,1,0,0,1,0,1,0,1,1,1,1,1,0,1,0,1,0,1,1,1,1,1,0,1,1,0,1,0,1,1,0,1,0,1,
0,0,1,0,1,0,0,0,1,1,1,1,0,0,0,0,0,0,1,0,0,0,0,0,1,0,1,0,0,1,0,0,0,1,0,1,1,1,0,1,0,1,0,1
,1,1,1,1,1,1,0,0,1,0,1,1,0,1,1,1,1,1,0,0,0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,0,1,1,0,0,0,0,0,1,0,
1,0,1,1,0,0,1,0,1,1,0,0,1,0,1,0,1,1,0,1,0,1,0,1,1,1,1,1,0,0,1,0,1,1,0,0,0,0,0,0,1,1,1,1
,1,0,1,0,0,0,0,0,0,0,1,0,1,0,0,1,1,0,1,0,1,0,0,1,0,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,0,0,0,1,1,
0,1,0,0,0,0,0,1,1,0,1,0,1,0,1,1,1,1,0,0,1,1,1,1,0,1,0,0,0,0,1,1,0,1,0,1,0,0,0,0,0,0,1,0
,1,1,0,1,0,1,1,0,1,0,0,0,0,1,0,1,1,0,0,1,0,1,0,1,0,0,0,0,0,1,1,0,1,0,0,1,0,1,0,1,0,0,0,
0,0,0,0,0,0,1,0,1,1,1,1,1,1,1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,1,0,1,0,1,0,0,0,0,0,0,0,1,0
,1,1,1,1,0,1,0,0,0,0,1,0,0,0,0,0,0,0,0,1,0,0,1,0,1,1,0,1,1,1,1,1,0,0,0,1,0,1,0,1,1,1,1,1,
0,1,1,1,1,1,1,0,0,1,1,1,1,0,0,0,0,1,0,0,1,0,0,0,0,0,0,0,1,0,1,1,0,0,0,0,0,1,1,1,1,1,1,1,0
,0,1,0,1,1,1,0,1,0,1,1,0,1,1,1,1,1,1,1,0,0,0,0,0,1,0,1,0,0,1,1,0,1,0,0,0,1,1,1,1,0,0,1,
1,1,1,0,1,0,1,0,1,1,1,1,0,1,0,1,1,1,0,1,0,0,1,0,0,0,1,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1
,0,0,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,1,1,1,1,1,1,0,1,0,0,1,0,1 };
    int F[768] =
{ 1,1,1,1,0,1,0,0,1,0,1,1,1,1,1,1,1,1,0,0,1,1,1,1,1,1,0,0,1,1,0,0,0,1,1,0,1,1,0,0,0,1
,1,1,1,0,0,0,1,0,1,0,0,0,1,1,1,1,1,1,0,0,0,0,1,1,0,0,0,0,0,1,0,0,1,1,1,0,0,1,1,1,0,0,1,0,
1,1,0,0,1,1,1,1,0,0,1,0,0,0,1,0,1,1,1,0,0,1,1,0,1,0,1,1,1,0,1,0,1,1,0,0,0,1,1,0,0,1,0,1
,0,0,0,1,0,1,0,1,0,0,1,1,0,1,1,1,0,1,0,1,0,1,1,1,1,1,1,1,0,1,0,1,1,1,1,1,1,1,1,0,0,0,1,0,1,0,
0,1,0,0,1,1,0,1,0,1,1,0,1,0,1,1,0,0,0,0,1,1,0,1,0,0,1,1,0,0,0,1,0,0,1,1,1,1,0,0,0,0,0,1,0
```

```
, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 1,
0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 1
, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0,
1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0
, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0,
0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0
, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0,
1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1
, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1,
1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0
, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 0, 0, 0, 1, 1,
1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1
, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 1 };
    for (i = 0; i < 768; i++)
    {
        data3[i] = A[i] ^ F[i];
    }
    n = 0;

        for (i = 0; i < 8; ++i)//第三个函数逆向
        {
            for (j = 0; j < 8; ++j)
            {
                output1[j + 8 * i] = ((buf2[i+8 * n] & (unsigned char)pow(2, j)) >> j);
            }
        }

        for (i = 63; i >= 0; --i)//第二个函数逆向
        {
            v4[data1[i] + 31] = output1[i];
        }
        for (i = 31; i >= 0; --i)
        {
            v4[i + 96] = v4[i + 64];
        }
        for (i = 31; i >= 0; --i)
        {
            v5[i] = v4[i + 32];
        }
        for (j = 15; j >= 0; --j)
        {
            for (i = 31; i >= 0; --i)
            {
                v3[i] = v4[i + 96];
```

```
            }
            dealv4(j);
            for (i = 0; i < 32; ++i)
            {
                v4[i + 96] = v5[i] ^ v4[i];
            }

            for (i = 31; i >= 0; --i)
            {
                v5[i] = v3[i];
            }
        }
        for (i = 31; i >= 0; --i)
        {
            v4[i + 64] = v5[i];
        }
        for (i = 31; i >= 0; --i)
        {
            v4[i + 32] = v4[i + 96];
        }
        for (i = 63; i >= 0; --i)
        {
            output0[data0[i] - 1] = v4[i + 32];
        }

        for (i = 0; i < 8; ++i)//第一个函数逆向
        {
            for (j = 0; j < 8; ++j)
            {
                input[i] |= output0[j + 8 * i] << j;
            }
        }


        for (i = 0; i < 8; i++)
        {
            printf("%c", input[i]);
        }

    return 0;
}


void dealv4(int k)
```

```
{
    int v4_1[80];
    int v5_1, v6, v7, v8, v9, v10, v12, i, j;
    for (i = 0; i < 48; ++i)
    {
        v4_1[i] = v3[data2[i] - 1];
    }
    for (i = 0; i < 48; ++i)
    {
        v9 = v4_1[i] != data3[i + 48 * k];
        v4_1[i] = v9;
    }
    for (i = 0; i < 8; ++i)
    {
        v12 = 6 * i;
        v6 = v4_1[6 * i + 5] + 2 * v4_1[6 * i];
        v5_1 = v4_1[6 * i + 4] + 2 * v4_1[6 * i + 3] + 4 * v4_1[6 * i + 2] + 8 * v4_1[6
* i + 1];
        v10 = data4[64 * i + 16 * v6 + v5_1];
        v12 = 4 * i;
        for (j = 3; j >= 0; --j)
        {
            v7 = v10 % 2 != 0;
            v4_1[j + 48 + v12] = v7;
            v10 /= 2;
        }
    }
    for (i = 0; i < 32; ++i)
    {
        v4[i] = v4_1[data5[i] + 47];
    }
}
```

依次对 n 赋值 0，1，2，3



得到 flag

hgame{WOWOW_h@ppy_n3w_ye4r_2022}

ezvm

查看伪代码

```
while ( 1 )
{
  v3 = dword_49F020[dword_49F020[0] + 0x6D];
  if ( v3 == -1 )
    return 0;
  switch ( v3 )
  {
    case 0:
      sub_401770(&dword_49F020[3], &dword_49F020[2]);
      break;
    case 1:
      sub_401775(&dword_49F020[2]);
      break;
    case 2:
      sub_401779(&dword_49F020[2]);
      break;
    case 3:
      sub_40178C(&dword_49F020[3], &dword_49F020[7]);
      break;
    case 4:
      sub_401798();
      break;
    case 5:
      sub_4017BB();
      break;
    case 6:
      sub_4017DE();
      break;
    case 7:
      sub_401825();
      break;
    case 8:
      sub_40183B();
```

```
41        break;
42      case 9:
43        sub_401851();
44        break;
45      case 10:
46        sub_401867();
47        break;
48      case 11:
49        sub_40187D();
50        break;
51      case 12:
52        sub_4018BE((unsigned int)dword_49F020[6]);
53        break;
54      case 13:
55        sub_4018CE((unsigned int)dword_49F020[6]);
56        break;
57      case 14:
58        sub_401791((unsigned int)dword_49F020[2]);
59        break;
60      case 15:
61        sub_401893((unsigned int)dword_49F020[3], (unsigned int)dword_49F020[5]);
62        break;
63      case 16:
64        sub_4018DE(&dword_49F020[3]);
65        break;
66      case 17:
67        sub_4018F3(&dword_49F020[3]);
68        break;
69      case 18:
70        sub_401904();
71        break;
72      case 19:
73        sub_40193A();
74        break;
75      case 20:
76        sub_401953();
```

```
77        break;
78      case 21:
79        sub_40177D();
80        break;
81      default:
82        break;
83    }
84    ++dword_49F020[0];
85  }
86}
```

编写 c 逆向

```
#include<stdio.h>
int main()
{
    //data0[2]统计输入的字符的数量
    int data0[8] = { 0,-1,0,0,0,0,0,0 };//dword_49F020[0-7]
    int data1[66] =
```

```
{ 0x12,0x8,0x12,0x9,0x10,0x4,0x1,0x0F,0x0D,0x2,0x12,0x8,0x12,0x9,0x0,0x4,0x0F,0x0D,0x12
,0x9,0x12,0x0A,0x13,0x12,0x0B,0x15,0x3,0x14,0x1,0x0,0x0F,0x0D,0x12,0x0A,0x12,0x12,0x12,
0x8,0x13,0x0F,0x7,0x4,0x9,0x0D,0x9,0x8,0x5,0x6,0x4,0x1,0x0,0x0F,0x0D,0x12,0x9,0x12,0x8,
0x12,0x0A,0x12,0x7,0x0F,0x0C,0x11,0x0E,-1 };
    int data2[83] = { 0x0A,-5,0x20,0x2F,-
10,0x0,0x5E,0x46,0x61,0x43,0x0E,0x53,0x49,0x1F,0x51,0x5E,0x36,0x37,0x29,0x41,0x63,0x3B,
0x64,0x3B,0x15,0x18,0x5B,0x3E,0x22,0x50,0x46,0x5E,0x35,0x4E,0x43,0x23,0x60,0x3B,0x0,-
17,0x15,0x8E,0x88,0x0A3,0x99,0x0C4,0x0A5,0x0C3,0x0DD,0x19,0x0EC,0x6C,0x9B,0x0F3,0x1B,0x
8B,0x5B,0x3E,0x9B,0x0F1,0x86,0x0F3,0x0F4,0x0A4,0x0F8,0x0F8,0x98,0x0AB,0x86,0x89,0x61,0x
22,0x0C1,0x2,0x0,-6,0x73,0x75,0x63,0x63,0x65,0x73,0x73 };
    int input[320] = { 0 };//dword_49F020[? + 9]
    int v3;
    int iii = 0;
    int i;
    while (1)
    {
        v3 = data1[data0[0]];
        if (v3 == -1)
        {
            return 0;
        }
        switch (v3)
        {
        case 0:
            data0[3]=data0[2];
            break;
        case 1:
            data0[2]++;
            break;
        case 2:
            data0[2]--;
            break;
        case 3:
            data0[3]^=data0[7];
            break;
        case 4:
            data0[1]++;
            input[data0[1]] = data0[3];
            break;
        case 5:
            data0[1]++;
            input[data0[1]] = data0[5];
            break;
        case 6:
```

```
        data0[1]++;
        input[data0[1]] = data0[6];
        break;
    case 7:
        if (data0[1] != -1)
        {
            --data0[1];
            data0[3] = input[data0[1]+1];
        }
        break;
    case 8:
        if (data0[1] != -1)
        {
            --data0[1];
            data0[5] = input[data0[1]+1];
        }
        break;
    case 9:
        if (data0[1] != -1)
        {
            --data0[1];
            data0[6] = input[data0[1]+1];
        }
        break;
    case 10:
        if (data0[1] != -1)
        {
            --data0[1];
            data0[2] = input[data0[1]+1];
        }
        break;
    case 11:
        if (data0[1] != -1)
        {
            --data0[1];
            data0[7] = input[data0[1]+1];
        }
        break;
    case 12:
        if (!iii)
            data0[0] += data0[6];
        break;
    case 13:
        if (iii)
```

```
        data0[0] += data0[6];
    break;
case 14:
    data0[0] += data0[2];
    break;
case 15:
    if (data0[3] == data0[5])
    {
        iii = 0;
    }
    else if (data0[3] >= data0[5])
    {
        if (data0[3] > data0[5])
            iii = 1;
    }
    else
    {
        iii = -1;
    }
    break;
case 16:
    data0[3]=getchar();
    break;
case 17:
    putchar(data0[3]);
    break;
case 18:
    int v0;
    v0 = data0[4]++;
    ++data0[1];
    input[data0[1]] = data2[v0];
    break;
case 19:
    data0[3] = input[data0[2]];
    break;
case 20:
    input[data0[2]] = data0[3];
    break;
case 21:
    data0[3] *= 2;
    break;
default:
    break;
}
```

```
        printf("v3 = %2d||", v3);
        for (i = 0; i <= 7; i++)
        {
            printf(" %2x ", data0[i]);
        }
        printf("\n");

        ++data0[0];
    }

}
/*输出格式为 v3 data0[0] data0[1] data0[2] data0[3] data0[4] data0[5] data0[6] data0[7]*/
```

分析输出

```
v3 =  8|  b  42  42   a   3  20  fffffffb   0
v3 = 18|  c  43  42   a   4  20  fffffffb   0
v3 =  9|  d  42  42   a   4  20  2f   0
v3 =  0|  e  42  42  42   4  20  2f   0
v3 =  4|  f  43  42  42   4  20  2f   0
v3 = 15| 10  43  42  42   4  20  2f   0
v3 = 13| 40  43  42  42   4  20  2f   0
```

2f 表明输入的长度位 0x20,及 32 位

输入 32 位后发现循环

```
v3 = 11| 18  21   0  30   7  20  fffffff6  5e
v3 = 21| 19  21   0  60   7  20  fffffff6  5e
v3 =  3| 1a  21   0  3e   7  20  fffffff6  5e
v3 = 20| 1b  21   0  3e   7  20  fffffff6  5e
v3 =  1| 1c  21   1  3e   7  20  fffffff6  5e
v3 =  0| 1d  21   1   1   7  20  fffffff6  5e
v3 = 15| 1e  21   1   1   7  20  fffffff6  5e
v3 = 13| 15  21   1   1   7  20  fffffff6  5e
v3 = 19| 16  21   1  31   7  20  fffffff6  5e
v3 = 18| 17  22   1  31   8  20  fffffff6  5e
v3 = 11| 18  21   1  31   8  20  fffffff6  46
v3 = 21| 19  21   1  62   8  20  fffffff6  46
v3 =  3| 1a  21   1  24   8  20  fffffff6  46
v3 = 20| 1b  21   1  24   8  20  fffffff6  46
v3 =  1| 1c  21   2  24   8  20  fffffff6  46
v3 =  0| 1d  21   2   2   8  20  fffffff6  46
v3 = 15| 1e  21   2   2   8  20  fffffff6  46
v3 = 13| 15  21   2   2   8  20  fffffff6  46
v3 = 19| 16  21   2  32   8  20  fffffff6  46
v3 = 18| 17  22   2  32   9  20  fffffff6  46
```

分析这个循环后

```
v3 = 11|| 18  21   0  30   7  20  fffffff6  5e//为 data0[7]赋值
v3 = 21|| 19  21   0  60   7  20  fffffff6  5e//输入的单个字符 asc 码乘以 2
v3 =  3|| 1a  21   0  3e   7  20  fffffff6  5e//将乘 2 后的数与 data0[7]异或
v3 = 20|| 1b  21   0  3e   7  20  fffffff6  5e//用处理后的字符将原字符替换
v3 =  1|| 1c  21   1  3e   7  20  fffffff6  5e//为下一个字符的处理做准备
v3 =  0|| 1d  21   1   1   7  20  fffffff6  5e//为下一个字符的处理做准备
v3 = 15|| 1e  21   1   1   7  20  fffffff6  5e//比较是否超出了 32 位
v3 = 13|| 15  21   1   1   7  20  fffffff6  5e//是否退出
v3 = 19|| 16  21   1  31   7  20  fffffff6  5e 为将下一个字符赋值给 data0[3]
v3 = 18|| 17  22   1  31   8  20  fffffff6  5e//为 data0[7]赋值
```

所有字符处理完成后执行下列语句

```
v3 = 10|| 21  21   0  20  27  20  fffffff6  3b
v3 = 18|| 22  22   0  20  28  20  fffffff6  3b
v3 = 18|| 23  23   0  20  29  20  fffffff6  3b
v3 = 18|| 24  24   0  20  2a  20  fffffff6  3b
v3 =  8|| 25  23   0  20  2a  8e  fffffff6  3b
v3 = 19|| 26  23   0  3e  2a  8e  fffffff6  3b
v3 = 15|| 27  23   0  3e  2a  8e  fffffff6  3b
v3 =  7|| 28  22   0  15  2a  8e  fffffff6  3b
v3 =  4|| 29  23   0  15  2a  8e  fffffff6  3b
v3 =  9|| 2a  22   0  15  2a  8e  15  3b
v3 = 13|| 40  22   0  15  2a  8e  15  3b
```

经过简单的分析后编写 c 逆向

```c
#include<stdio.h>
int main()
{
    int s1[32] =
{ 0x5E,0x46,0x61,0x43,0x0E,0x53,0x49,0x1F,0x51,0x5E,0x36,0x37,0x29,0x41,0x63,0x3B,0x64,
0x3B,0x15,0x18,0x5B,0x3E,0x22,0x50,0x46,0x5E,0x35,0x4E,0x43,0x23,0x60,0x3B };
    int s2[32] =
{ 0x8E,0x88,0x0A3,0x99,0x0C4,0x0A5,0x0C3,0x0DD,0x19,0x0EC,0x6C,0x9B,0x0F3,0x1B,0x8B,0x5
B,0x3E,0x9B,0x0F1,0x86,0x0F3,0x0F4,0x0A4,0x0F8,0x0F8,0x98,0x0AB,0x86,0x89,0x61,0x22,0x0
C1 };
    int i;
    for (i = 0; i < 32; i++)
    {
        s1[i] ^= s2[i];
        s1[i] /= 2;
        printf("%c", s1[i]);
    }
```

```
    return 0;
}
```

得到 flag

```
hgame{Ea$Y-Vm-t0-PrOTeCT_cOde!!}
```

hgame{Ea$Y-Vm-t0-PrOTeCT_cOde!!}