攻防世界reverse新手练习区通关教程

原创

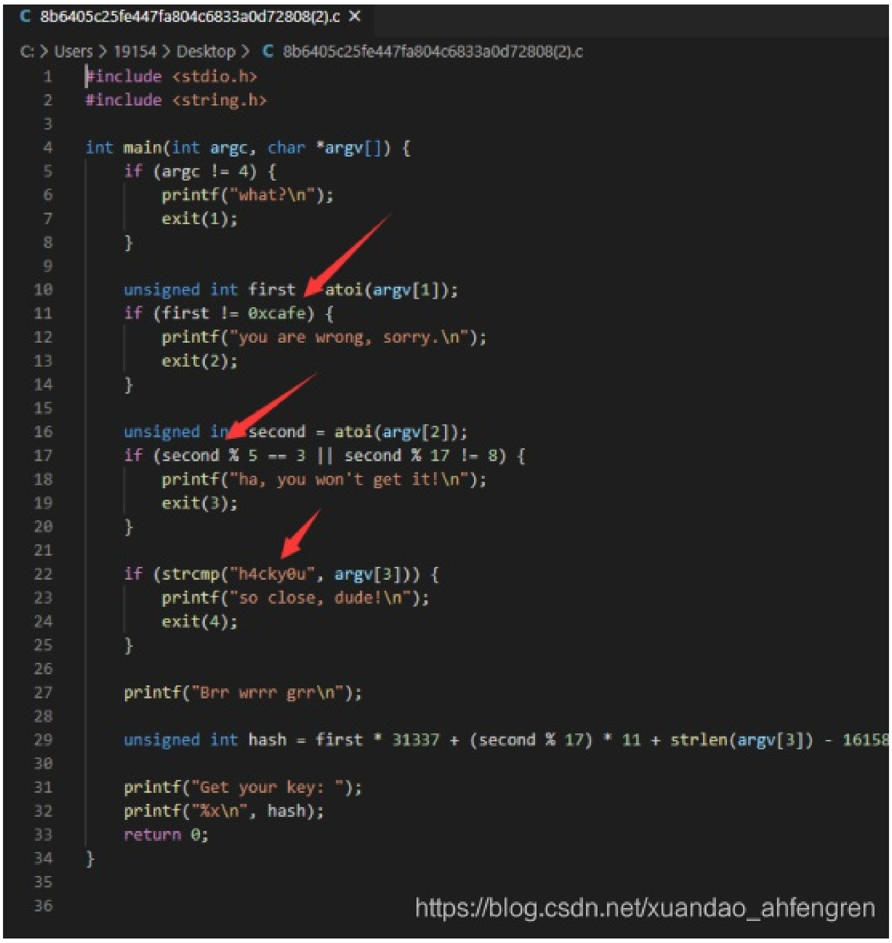
锋刃科技 于 2020-05-25 10:37:41 发布 999 收藏 8
文章标签： 攻防世界 ctf

open-source

下载附件打开来看看，三个条件达成即可

第一个是0xcafe,第二个是满足or的一个数字，第三个是h4cky0u

最后会输出key



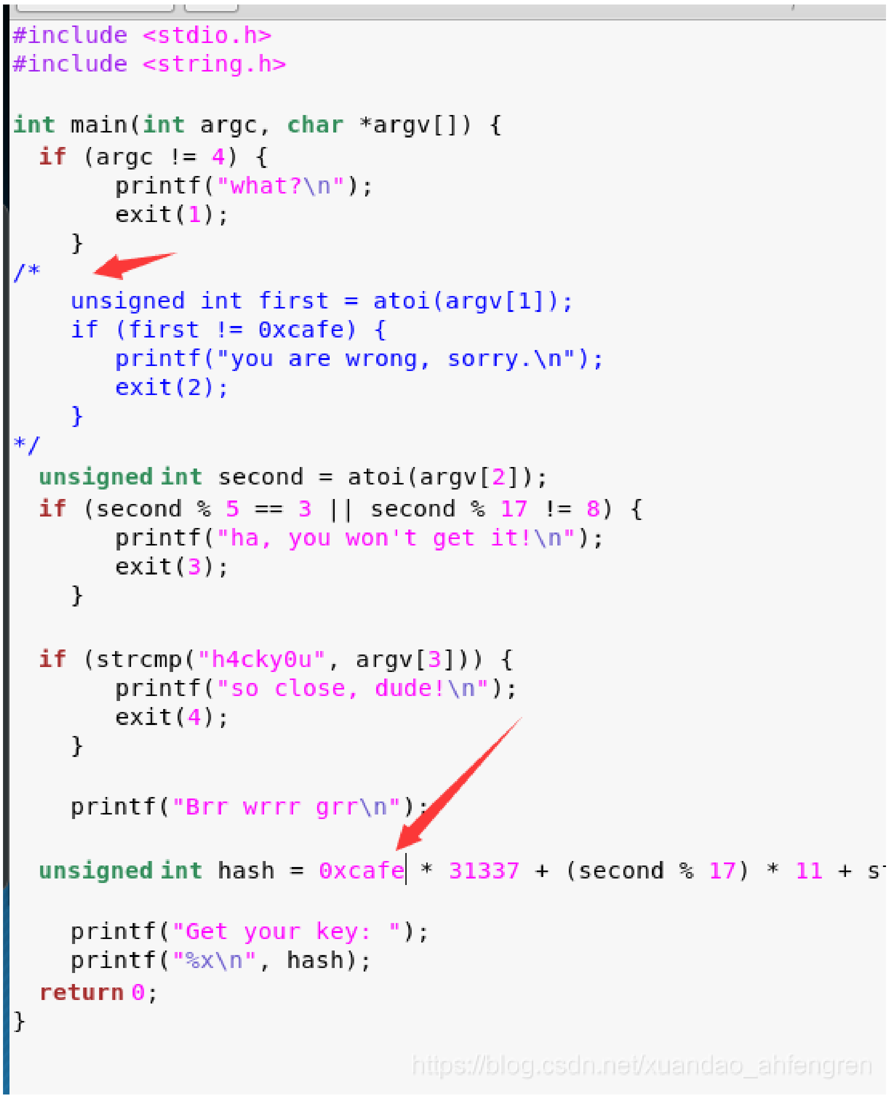
C 8b6405c25fe447fa804c6833a0d72808(2).c ×

C: > Users > 19154 > Desktop > C 8b6405c25fe447fa804c6833a0d72808(2).c

```c
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[]) {
    if (argc != 4) {
        printf("what?\n");
        exit(1);
    }

    unsigned int first  atoi(argv[1]);
    if (first != 0xcafe) {
        printf("you are wrong, sorry.\n");
        exit(2);
    }

    unsigned in second = atoi(argv[2]);
    if (second % 5 == 3 || second % 17 != 8) {
        printf("ha, you won't get it!\n");
        exit(3);
    }

    if (strcmp("h4cky0u", argv[3])) {
        printf("so close, dude!\n");
        exit(4);
    }

    printf("Brr wrrr grr\n");

    unsigned int hash = first * 31337 + (second % 17) * 11 + strlen(argv[3]) - 16158

    printf("Get your key: ");
    printf("%x\n", hash);
    return 0;
}
```

https://blog.csdn.net/xuandao_ahfengren

把first注释掉，加上参数



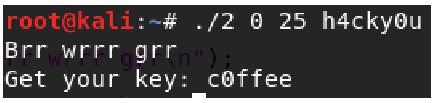
```c
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[]) {
  if (argc != 4) {
        printf("what?\n");
        exit(1);
    }
/*
    unsigned int first = atoi(argv[1]);
    if (first != 0xcafe) {
        printf("you are wrong, sorry.\n");
        exit(2);
    }
*/
  unsigned int second = atoi(argv[2]);
  if (second % 5 == 3 || second % 17 != 8) {
        printf("ha, you won't get it!\n");
        exit(3);
    }

  if (strcmp("h4cky0u", argv[3])) {
        printf("so close, dude!\n");
        exit(4);
    }

    printf("Brr wrrr grr\n");

  unsigned int hash = 0xcafe * 31337 + (second % 17) * 11 + s

    printf("Get your key: ");
    printf("%x\n", hash);
  return 0;
}
```

编译c文件

gcc 1.c -o 2

./2 0 25 h4cky0u

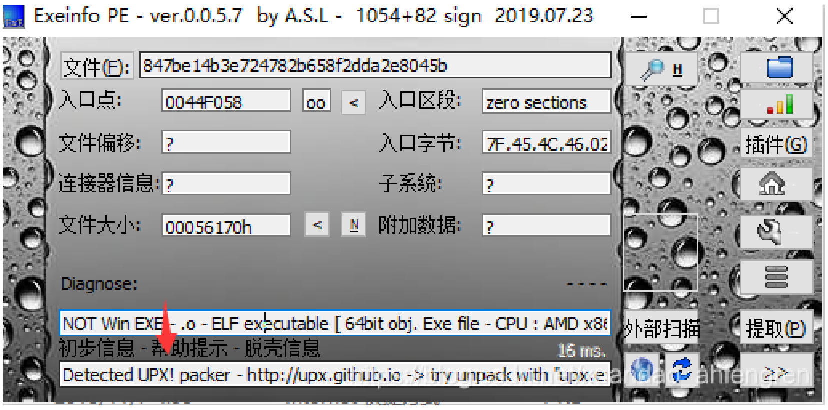
最后输出key即可



```
root@kali:~# ./2 0 25 h4cky0u
Brr wrrr grr
Get your key: c0ffee
```

simple-unpack

我们先用扫描壳工具，发现有upx壳



Exeinfo PE - ver.0.0.5.7 by A.S.L - 1054+82 sign 2019.07.23

文件(F): 847be14b3e724782b658f2dda2e8045b

入口点: 0044F058

入口区段: zero sections

文件偏移: ?

入口字节: 7F.45.4C.46.02

连接器信息: ?

子系统: ?

文件大小: 00056170h

附加数据: ?

Diagnose:

NOT Win EXE - .o - ELF executable [64bit obj. Exe file - CPU : AMD x86

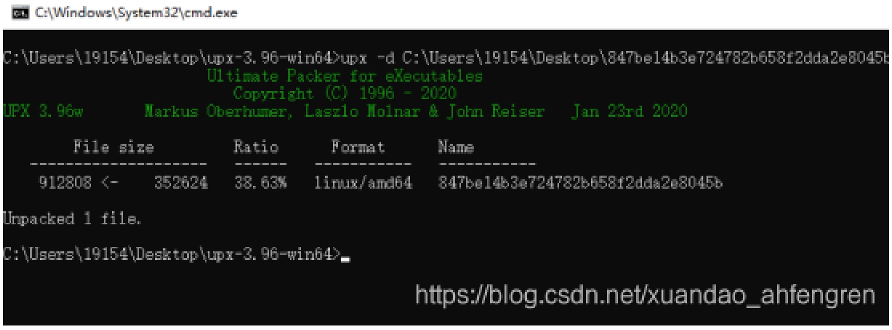
初步信息 - 帮助提示 - 脱壳信息

16 ms.

Detected UPX! packer - http://upx.github.io -> try unpack with "upx.e

进行脱壳

upx -d

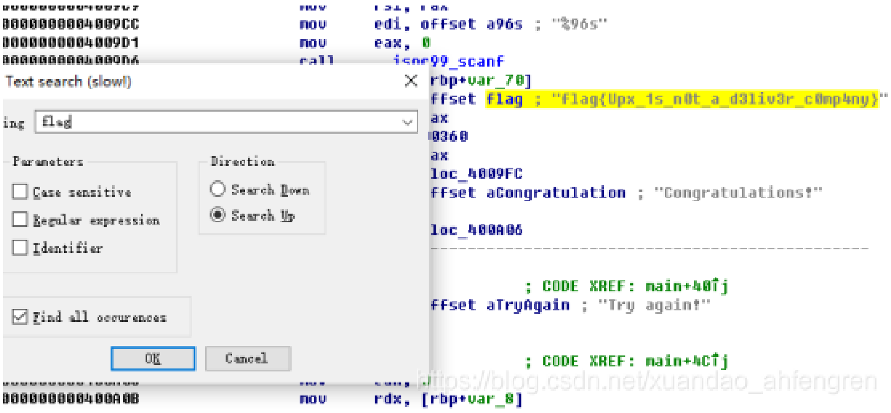


C:\Windows\System32\cmd.exe

```
C:\Users\19154\Desktop\upx-3.96-win64>upx -d C:\Users\19154\Desktop\847be14b3e724782b658f2dda2e8045b
                       Ultimate Packer for eXecutables
                          Copyright (C) 1996 - 2020
UPX 3.96w       Markus Oberhumer, Laszlo Molnar & John Reiser   Jan 23rd 2020

        File size         Ratio      Format      Name
   --------------------   ------   -----------   -----------
    912808 <-    352624   38.63%   linux/amd64   847be14b3e724782b658f2dda2e8045b

Unpacked 1 file.

C:\Users\19154\Desktop\upx-3.96-win64>_
```

https://blog.csdn.net/xuandao_ahfengren

直接拿去ida搜索flag即可



```
00000000004009CC                 mov     edi, offset a96s ; "%96s"
00000000004009D1                 mov     eax, 0
                                            rbp+var_70]
                                            ffset flag ; "flag{Upx_1s_n0t_a_d3liv3r_c0mp4ny}"
                                            ax
                                            0360
                                            ax
                                            loc_4009FC
                                            ffset aCongratulation ; "Congratulations!"

                                            loc_400A06
                                            ----------------------------------------------

                                                        ; CODE XREF: main+40↑j
                                            ffset aTryAgain ; "Try again!"

                                                        ; CODE XREF: main+4C↑j
0000000000400A0B                 mov     rdx, [rbp+var_8]
```

Text search (slow!)

ing flag

Parameters

□ Case sensitive

□ Regular expression

□ Identifier

Direction

○ Search Down

● Search Up

☑ Find all occurences

https://blog.csdn.net/xuandao_ahfengren

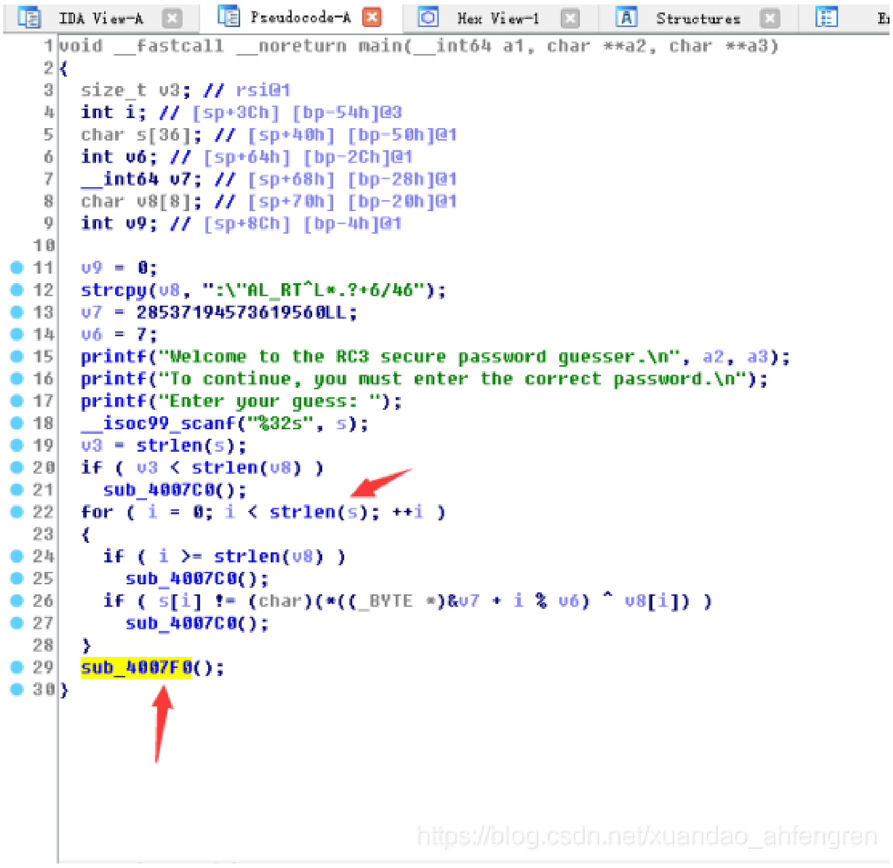
Logmein

直接加载进ida然后指定main按F5

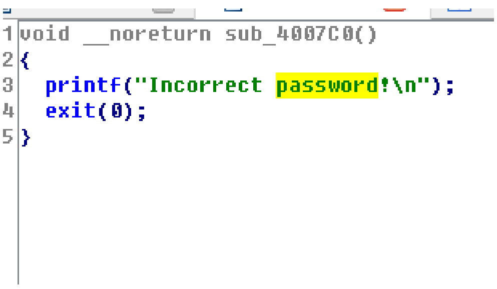
s是用户输入的字符串，先进行比较长度，如果长度比v8小

判断如果输入的字符串和经过运算后的后字符串不等则成功

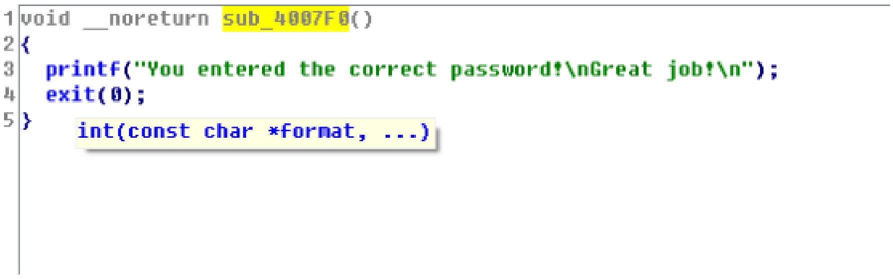
说明字符串就是flag



```
void __fastcall __noreturn main(__int64 a1, char **a2, char **a3)
{
  size_t v3; // rsi@1
  int i; // [sp+3Ch] [bp-54h]@3
  char s[36]; // [sp+40h] [bp-50h]@1
  int v6; // [sp+64h] [bp-2Ch]@1
  __int64 v7; // [sp+68h] [bp-28h]@1
  char v8[8]; // [sp+70h] [bp-20h]@1
  int v9; // [sp+8Ch] [bp-4h]@1

  v9 = 0;
  strcpy(v8, ":\"AL_RT^L*.?+6/46");
  v7 = 28537194573619560LL;
  v6 = 7;
  printf("Welcome to the RC3 secure password guesser.\n", a2, a3);
  printf("To continue, you must enter the correct password.\n");
  printf("Enter your guess: ");
  __isoc99_scanf("%32s", s);
  v3 = strlen(s);
  if ( v3 < strlen(v8) )
    sub_4007C0();
  for ( i = 0; i < strlen(s); ++i )
  {
    if ( i >= strlen(v8) )
      sub_4007C0();
    if ( s[i] != (char)(*((_BYTE *)&v7 + i % v6) ^ v8[i]) )
      sub_4007C0();
  }
  sub_4007F0();
}
```



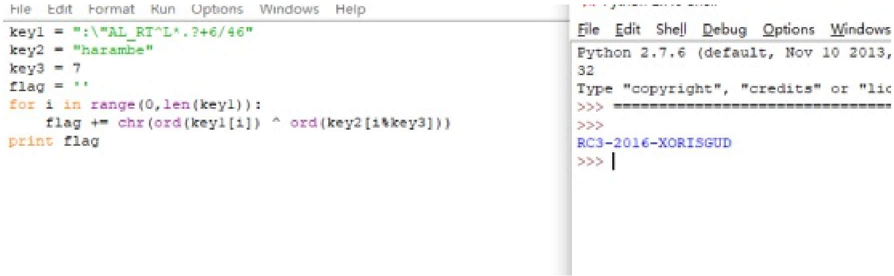
```
void __noreturn sub_4007C0()
{
  printf("Incorrect password!\n");
  exit(0);
}
```



```
void __noreturn sub_4007F0()
{
  printf("You entered the correct password!\nGreat job!\n");
  exit(0);
}
```

最后根据所知内容编写exp

```
key1 = ":\"AL_RT^L*.?+6/46"

key2 = "harambe"

key3 = 7

flag = ''

for i in range(0,len(key1)):

    flag += chr(ord(key1[i]) ^ ord(key2[i%key3]))

print flag
```



```
key1 = ":\"AL_RT^L*.?+6/46"
key2 = "harambe"
key3 = 7
flag = ''
for i in range(0,len(key1)):
    flag += chr(ord(key1[i]) ^ ord(key2[i%key3]))
print flag
```

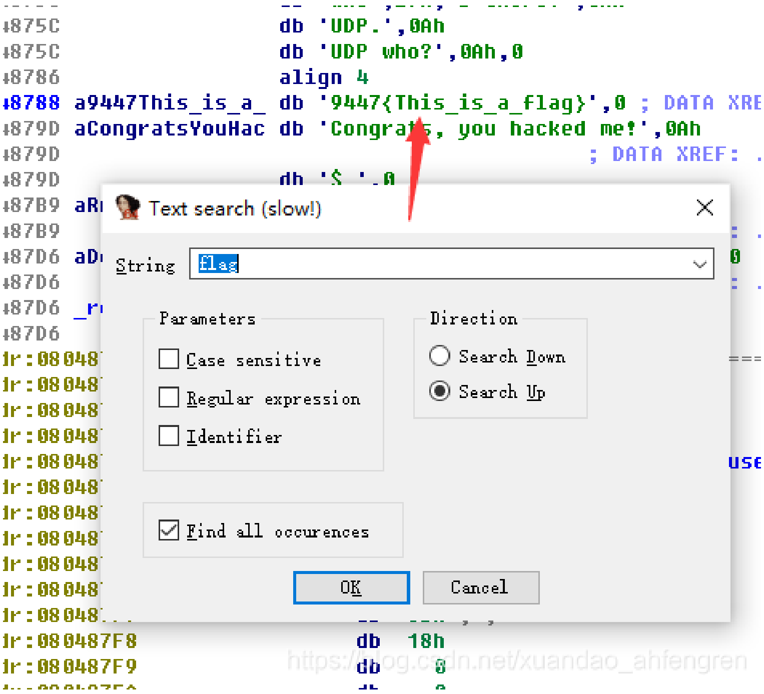
```
Python 2.7.6 (default, Nov 10 2013,
32
Type "copyright", "credits" or "lic
>>> ================================
>>>
RC3-2016-XORISGUD
>>> 
```

File Edit Format Run Options Windows Help
File Edit Shell Debug Options Windows

Insanity

直接放进ida然后搜索flag即可

9447{This_is_a_flag}



```
875C                 db 'UDP.',0Ah
875C                 db 'UDP who?',0Ah,0
8786                 align 4
8788 a9447This_is_a_ db '9447{This_is_a_flag}',0 ; DATA XRE
879D aCongratsYouHac db 'Congrats, you hacked me!',0Ah
879D                                         ; DATA XREF: .
879D                 db '$ ',0
87B9 aR
87B9
87D6 aD
87D6
87D6 _r
87D6
r:08048
r:080487F8              db  18h
r:080487F9              db   0
```

Text search (slow!)

String: flag

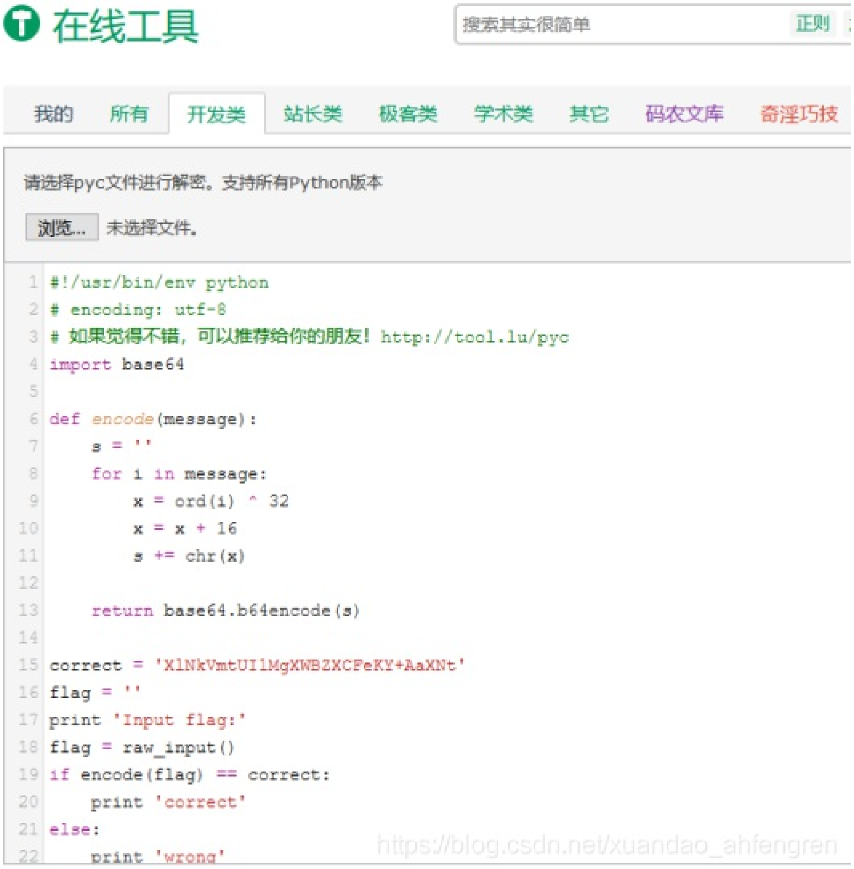
Parameters
- □ Case sensitive
- □ Regular expression
- □ Identifier

Direction
- ○ Search Down
- ● Search Up

☑ Find all occurences

python-trade

首先通过在线工具反编译出ptthon文件

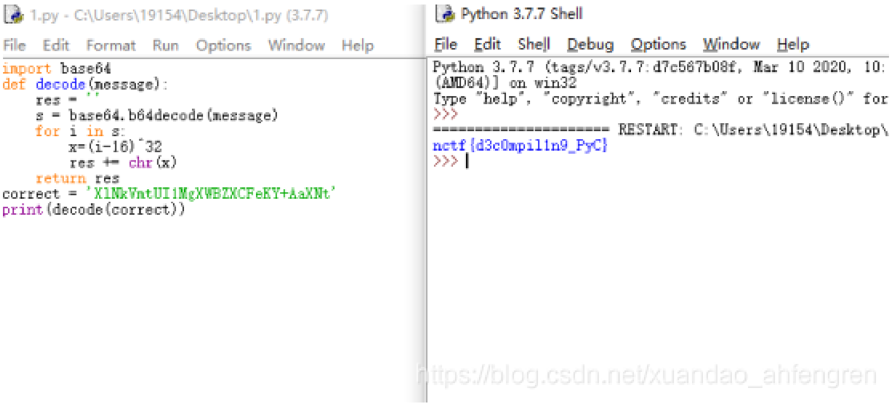


请选择pyc文件进行解密。支持所有Python版本

未选择文件。

```python
#!/usr/bin/env python
# encoding: utf-8
# 如果觉得不错，可以推荐给你的朋友！ http://tool.lu/pyc
import base64

def encode(message):
    s = ''
    for i in message:
        x = ord(i) ^ 32
        x = x + 16
        s += chr(x)

    return base64.b64encode(s)

correct = 'XlNkVmtUI1MgXWBZXCFeKY+AaXNt'
flag = ''
print 'Input flag:'
flag = raw_input()
if encode(flag) == correct:
    print 'correct'
else:
    print 'wrong'
```

然后通过他的编码方式写出解码方式

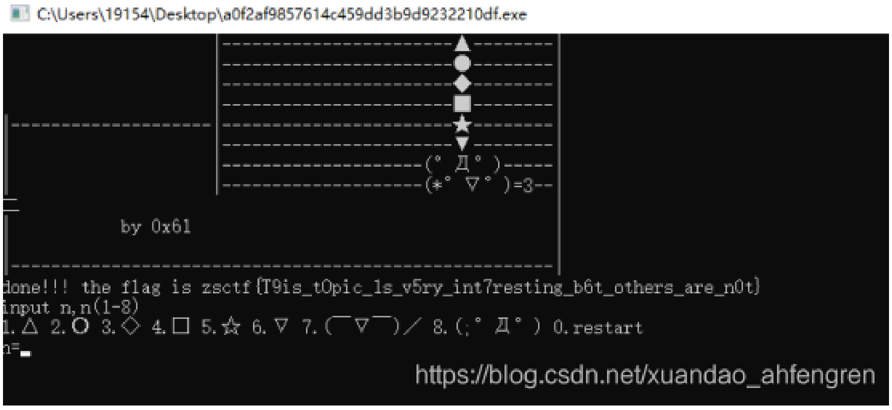

1.py - C:\Users\19154\Desktop\1.py (3.7.7)
File Edit Format Run Options Window Help
Python 3.7.7 Shell
File Edit Shell Debug Options Window Help

```python
import base64
def decode(message):
    res = ''
    s = base64.b64decode(message)
    for i in s:
        x=(i-16)^32
        res += chr(x)
    return res
correct = 'XlNkVmtUI1MgXWBZXCFeKY+AaXNt'
print(decode(correct))
```

```
Python 3.7.7 (tags/v3.7.7:d7c567b08f, Mar 10 2020, 10:
(AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for
>>>
====================== RESTART: C:\Users\19154\Desktop\
nctf{d3c0mpil1n9_PyC}
>>> |
```

Game

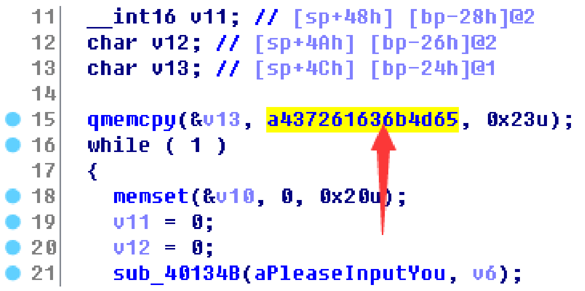
顺序输入8，7，6，5，4，3，2，1即可



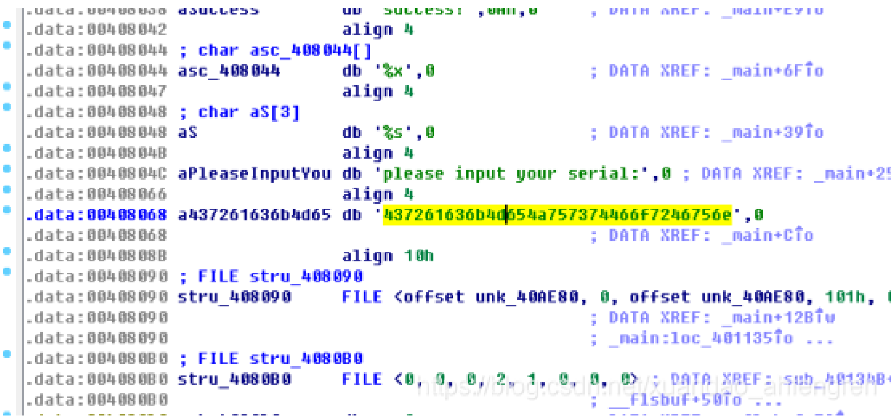
```
        by 0x61

done!!! the flag is zsctf{T9is_tOpic_1s_v5ry_int7resting_b6t_others_are_n0t}
input n,n(1-8)
1.△ 2.○ 3.◇ 4.□ 5.☆ 6.▽ 7.(￣▽￣)／ 8.(;°Д°) 0.restart
n=
```

Hello, CTF

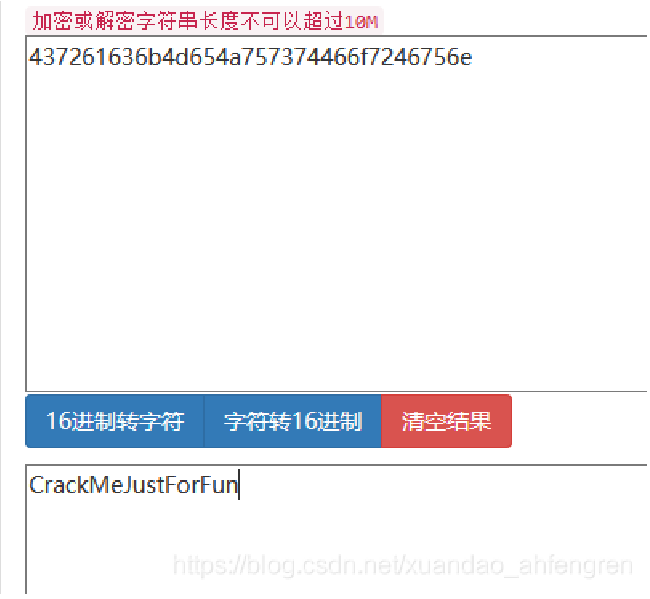
双击进去，就看到了十六进制的字符，转换一下即可



```
  __int16 v11; // [sp+48h] [bp-28h]@2
  char v12; // [sp+4Ah] [bp-26h]@2
  char v13; // [sp+4Ch] [bp-24h]@1

  qmemcpy(&v13, a437261636b4d65, 0x23u);
  while ( 1 )
  {
    memset(&v10, 0, 0x20u);
    v11 = 0;
    v12 = 0;
    sub_40134B(aPleaseInputYou, v6);
```



```
.data:00408042                 align 4
.data:00408044 ; char asc_408044[]
.data:00408044 asc_408044      db '%x',0              ; DATA XREF: _main+6F↑o
.data:00408047                 align 4
.data:00408048 ; char aS[3]
.data:00408048 aS              db '%s',0              ; DATA XREF: _main+39↑o
.data:0040804B                 align 4
.data:0040804C aPleaseInputYou db 'please input your serial:',0 ; DATA XREF: _main+2!
.data:00408066                 align 4
.data:00408068 a437261636b4d65 db '437261636b4d654a757374466f7246756e',0
.data:00408068                                         ; DATA XREF: _main+C↑o
.data:0040808B                 align 10h
.data:00408090 ; FILE stru_408090
.data:00408090 stru_408090     FILE <offset unk_40AE80, 0, offset unk_40AE80, 101h, (
.data:00408090                                         ; DATA XREF: _main+12B↑w
.data:00408090                                         ; _main:loc_401135↑o ...
.data:004080B0 ; FILE stru_4080B0
.data:004080B0 stru_4080B0     FILE <0, 0, 0, 2, 1, 0, 0, 0> ; DATA XREF: sub_40134B+
.data:004080B0                                         ; __flsbuf+50↑o ...
```



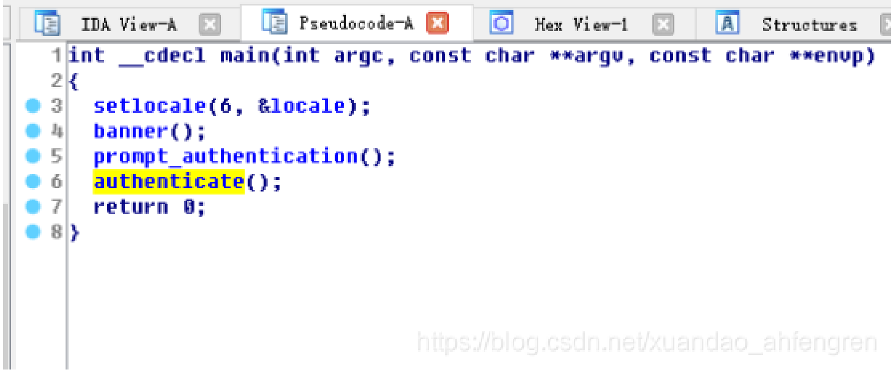
加密或解密字符串长度不可以超过10M

437261636b4d654a757374466f7246756e

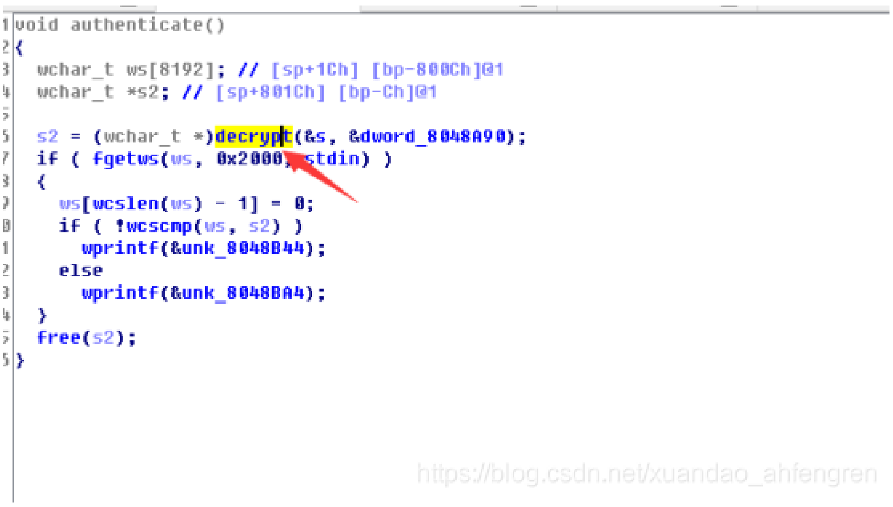
CrackMeJustForFun

no-strings-attached

F5直接进去，双击authenticate，有个加密函数



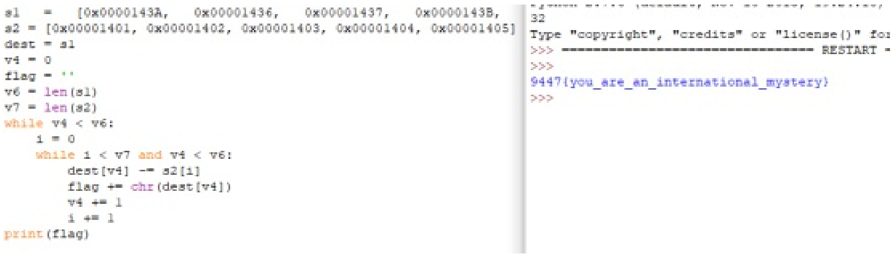
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  setlocale(6, &locale);
  banner();
  prompt_authentication();
  authenticate();
  return 0;
}
```



```
void authenticate()
{
  wchar_t ws[8192]; // [sp+1Ch] [bp-800Ch]@1
  wchar_t *s2; // [sp+801Ch] [bp-Ch]@1

  s2 = (wchar_t *)decrypt(&s, &dword_8048A90);
  if ( fgetws(ws, 0x2000, stdin) )
  {
    ws[wcslen(ws) - 1] = 0;
    if ( !wcscmp(ws, s2) )
      wprintf(&unk_8048B44);
    else
      wprintf(&unk_8048BA4);
  }
  free(s2);
}
```

然后就是编写exp

```
s1   =   [0x0000143A,   0x00001436,   0x00001437,   0x0000143B,   0x00001480,   0x0000147A, 0x00001471, 0x0
s2 = [0x00001401, 0x00001402, 0x00001403, 0x00001404, 0x00001405]
dest = s1
v4 = 0
flag = ''
v6 = len(s1)
v7 = len(s2)
while v4 < v6:
    i = 0
    while i < v7 and v4 < v6:
        dest[v4] -= s2[i]
        flag += chr(dest[v4])
        v4 += 1
        i += 1
print(flag)
```

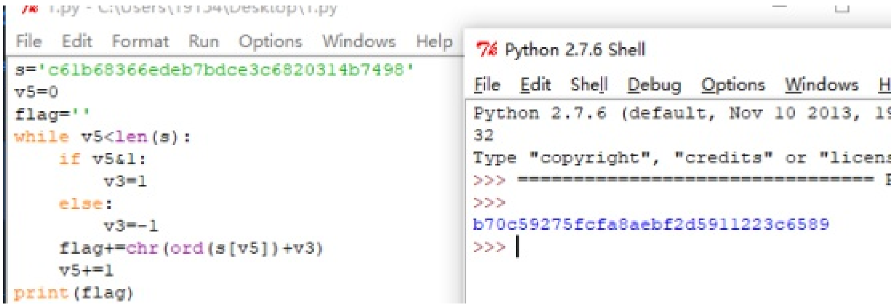


```
s1   =   [0x0000143A,   0x00001436,   0x00001437,   0x0000143B,
s2 = [0x00001401, 0x00001402, 0x00001403, 0x00001404, 0x00001405]
dest = s1
v4 = 0
flag = ''
v6 = len(s1)
v7 = len(s2)
while v4 < v6:
    i = 0
    while i < v7 and v4 < v6:
        dest[v4] -= s2[i]
        flag += chr(dest[v4])
        v4 += 1
        i += 1
print(flag)
```

```
32
Type "copyright", "credits" or "license()" fo:
>>> ================================ RESTART =
>>>
9447{you_are_an_international_mystery}
>>>
```

Getit

```
s='c61b68366edeb7bdce3c6820314b7498'
v5=0
flag=''
while v5<len(s):
    if v5&1:
        v3=1
    else:
        v3=-1
    flag+=chr(ord(s[v5])+v3)
    v5+=1
print(flag)
```



1.py - C:\Users\19154\Desktop\1.py
File Edit Format Run Options Windows Help
Python 2.7.6 Shell
File Edit Shell Debug Options Windows H

```
s='c61b68366edeb7bdce3c6820314b7498'
v5=0
flag=''
while v5<len(s):
    if v5&1:
        v3=1
    else:
        v3=-1
    flag+=chr(ord(s[v5])+v3)
    v5+=1
print(flag)
```

```
Python 2.7.6 (default, Nov 10 2013, 19
32
Type "copyright", "credits" or "licens
>>> ================================ R
>>>
b70c59275fcfa8aebf2d5911223c6589
>>> 
```

re1

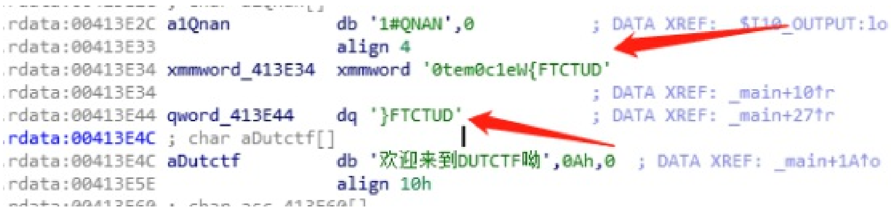
直接放进ida打开

假设一个十六进制数0x12345678

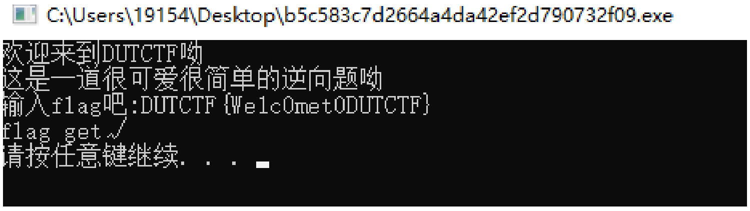
大端的存储方式是：12,34,56,78，然后读取的时候也是从前往后读

小端的存储方式是：78,56,34,12，然后读取的时候是从后往前读取

所以，最后的flag应该是：DUTCTF{We1c0met0DUTCTF}



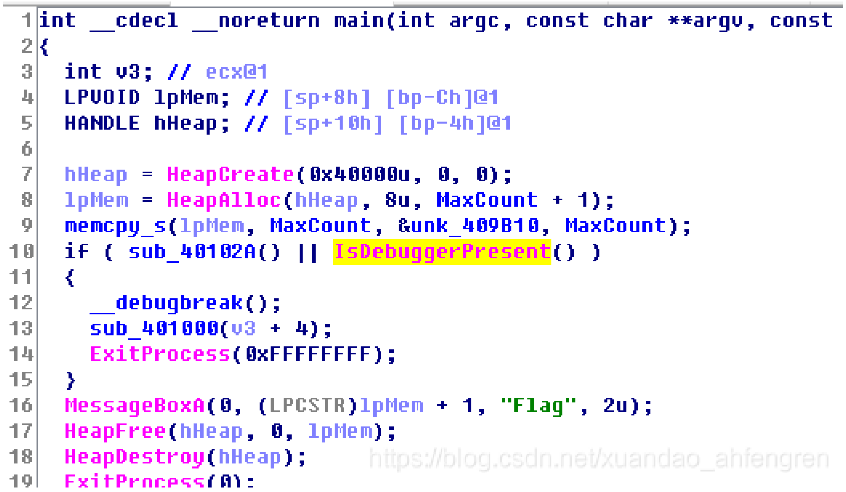
```
.rdata:00413E2C a1Qnan          db '1#QNAN',0           ; DATA XREF: $I10_OUTPUT:lo
.rdata:00413E33                 align 4
.rdata:00413E34 xmmword_413E34  xmmword '0tem0c1eW{FTCTUD'
.rdata:00413E34                                         ; DATA XREF: _main+10↑r
.rdata:00413E44 qword_413E44    dq '}FTCTUD'            ; DATA XREF: _main+27↑r
.rdata:00413E4C ; char aDutctf[]
.rdata:00413E4C aDutctf         db '欢迎来到DUTCTF呦',0Ah,0  ; DATA XREF: _main+1A↑o
.rdata:00413E5E                 align 10h
```



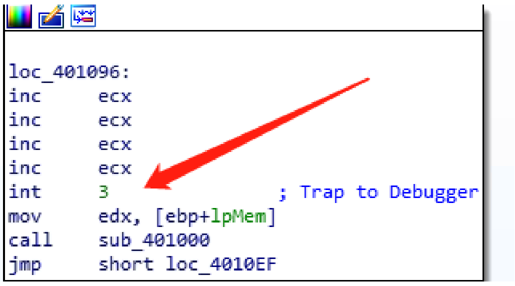
```
欢迎来到DUTCTF呦
这是一道很可爱很简单的逆向题呦
输入flag吧:DUTCTF{We1c0met0DUTCTF}
flag get √
请按任意键继续. . .
```

csaw2013reversing2

放进ida按F5，发现关键函数



```
int __cdecl __noreturn main(int argc, const char **argv, const
{
  int v3; // ecx@1
  LPVOID lpMem; // [sp+8h] [bp-Ch]@1
  HANDLE hHeap; // [sp+10h] [bp-4h]@1

  hHeap = HeapCreate(0x40000u, 0, 0);
  lpMem = HeapAlloc(hHeap, 8u, MaxCount + 1);
  memcpy_s(lpMem, MaxCount, &unk_409B10, MaxCount);
  if ( sub_40102A() || IsDebuggerPresent() )
  {
    __debugbreak();
    sub_401000(v3 + 4);
    ExitProcess(0xFFFFFFFF);
  }
  MessageBoxA(0, (LPCSTR)lpMem + 1, "Flag", 2u);
  HeapFree(hHeap, 0, lpMem);
  HeapDestroy(hHeap);
  ExitProcess(0);
```

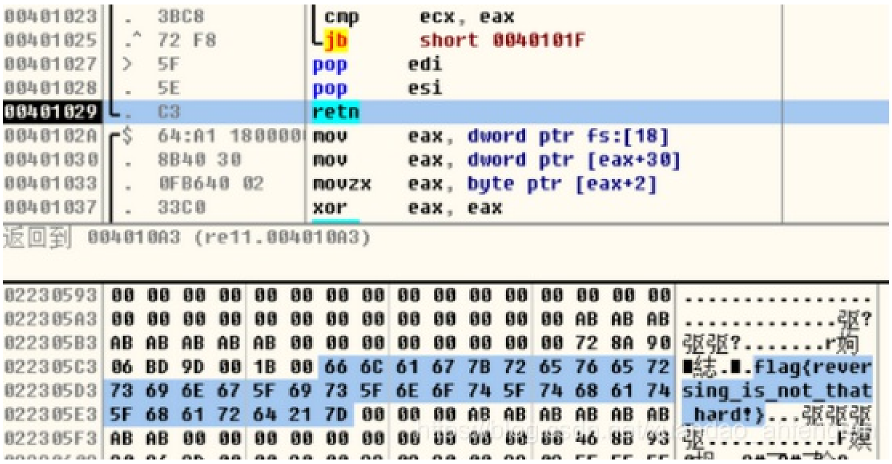


```
loc_401096:
inc     ecx
inc     ecx
inc     ecx
inc     ecx
int     3               ; Trap to Debugger
mov     edx, [ebp+lpMem]
call    sub_401000
jmp     short loc_4010EF
```

意思是如果在动态调试器中就进入判断运行，如果没有直接弹窗，显示乱码的值

Ollydbg

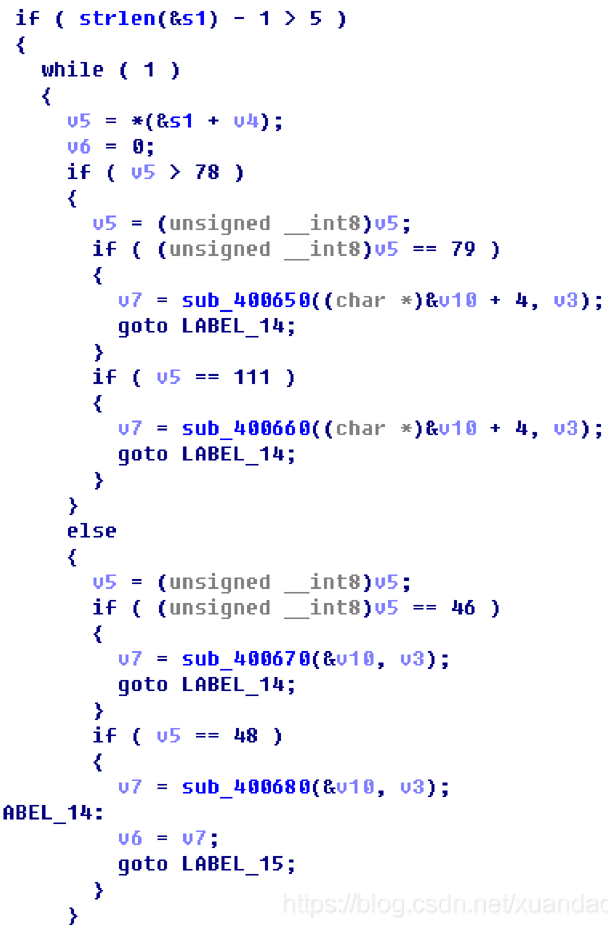
执行了mov指令，接下来调用call，F8继续执行，执行完，edx存的就是flag的地址



```
00401023  .   3BC8          cmp     ecx, eax
00401025  .^  72 F8         jb      short 0040101F
00401027  >   5F            pop     edi
00401028  .   5E            pop     esi
00401029  .   C3            retn
0040102A  $   64:A1 180000( mov     eax, dword ptr fs:[18]
00401030  .   8B40 30       mov     eax, dword ptr [eax+30]
00401033  .   0FB640 02     movzx   eax, byte ptr [eax+2]
00401037  .   33C0          xor     eax, eax
返回到 004010A3 (re11.004010A3)

02230593  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00  ................
022305A3  00 00 00 00 00 00 00 00 00 00 00 00 00 AB AB AB  .............张?
022305B3  AB AB AB AB AB 00 00 00 00 00 00 00 00 72 8A 90  张张?........r婀
022305C3  06 BD 9D 00 1B 00 66 6C 61 67 7B 72 65 76 65 72  ■结.■.flag{rever
022305D3  73 69 6E 67 5F 69 73 5F 6E 6F 74 5F 74 68 61 74  sing_is_not_that
022305E3  5F 68 61 72 64 21 7D 00 00 00 AB AB AB AB AB AB  _hard!}...张张张
022305F3  AB AB 00 00 00 00 00 00 00 00 00 00 00 46 8B 93  张...........F嬶
```

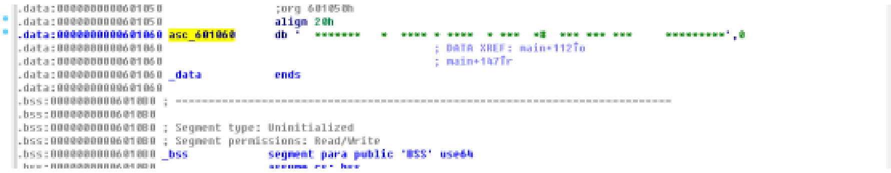
Maze

直接带进ida，发现main函数下有一些判断



```
 if ( strlen(&s1) - 1 > 5 )
 {
   while ( 1 )
   {
     v5 = *(&s1 + v4);
     v6 = 0;
     if ( v5 > 78 )
     {
       v5 = (unsigned __int8)v5;
       if ( (unsigned __int8)v5 == 79 )
       {
         v7 = sub_400650((char *)&v10 + 4, v3);
         goto LABEL_14;
       }
       if ( v5 == 111 )
       {
         v7 = sub_400660((char *)&v10 + 4, v3);
         goto LABEL_14;
       }
     }
     else
     {
       v5 = (unsigned __int8)v5;
       if ( (unsigned __int8)v5 == 46 )
       {
         v7 = sub_400670(&v10, v3);
         goto LABEL_14;
       }
       if ( v5 == 48 )
       {
         v7 = sub_400680(&v10, v3);
ABEL_14:
         v6 = v7;
         goto LABEL_15;
       }
     }
```

可以发现这些函数会跳到lable15的位置，然后，对lable15分析，发现特殊的字符串



```
.data:0000000000601050                 ;org 601050h
.data:0000000000601050                 align 20h
.data:0000000000601060 asc_601060      db '  *******   *  **** * ****  * ***  *#  *** *** ***     *********',0
.data:0000000000601060                                         ; DATA XREF: main+112↑o
.data:0000000000601060                                         ; main+147↑r
.data:0000000000601060 _data           ends
.data:0000000000601060
.bss:0000000000601080 ; ===========================================================================
.bss:0000000000601080
.bss:0000000000601080 ; Segment type: Uninitialized
.bss:0000000000601080 ; Segment permissions: Read/Write
.bss:0000000000601080 _bss            segment para public 'BSS' use64
.bss:0000000000601080                 assume cs:_bss
```

猜测可能是一个8*8的迷宫

根据迷宫最后得到的flag: nctf{o0oo00O000oooo..OO}