IDA Pro
Hex-Rays Decompiler
Simple Tricks

Hello!

aaaddress1(ADR)

1. Reverse Engineering Skills
   a. Windows
   b. Android
2. TDoHacker Core Member
3. Hack Bot
   a. CrackShield / MapleHack
   b. Tower Of Savior
   c. Facebook
4. CSharp,C/C++,Assembly x86 Python,Smali,Pascal,VB.NET

11:50

R0 國際會議廳 Conference Hall

What Google knows about you and your devices, and how to get it

Vladimir Katalov

11:50 ~ 12:40 (50 mins)

R1 第一會議室 Conference Room 1

Android AIDS:Automatic Intelligence De-advertisement Scheme In CSharproid

馬聖豪

11:50 ~ 12:40 (50 mins)

R2 第二會議室 Conference Room 2

木馬屠城 - 那些年你不知不覺間引入的漏洞

Flanker

11:50 ~ 12:40 (50 mins)

R4 交誼廳 Conference Room 4

惡意程式分析與逆向工程

HITCON GIRLS

11:50 ~ 12:40 (50 mins)


九阴真经
我这儿有本秘笈


熊掌大鹏
化骨绵掌
乾坤大挪移
凌波微步
降龙十八掌
移魂大法
吸星大法
玉女心经
九阴真经
不合适啊？还有


小孩打架才出这招!

Outline

- 動機
- IDA Pro Hex-Rays Decompiler是什麼
- 如何理解ASM Opcode並推導出C Code
- 意外的Fire Eye欺騙IDA Pro的梗
- IDA Pro Hex-Rays Decompiler解析弱點
- Q&A

動機

IDA Pro

by Ilfak Guilfanov

```
.text:004133F4                    mov     [ebp+var_10], esp
.text:004133F7 ; 11:    k = 0;
.text:004133F7                    mov     [ebp+k], 0
.text:004133FE ; 12:    for ( i = 0; i < 10; ++i )
.text:004133FE                    mov     [ebp+i], 0
.text:00413405                    jmp     short loc_413419
.text:00413407 ; ---------------------------------------
.text:00413407 ; 13:      k += i;
.text:00413407
.text:00413407 loc_413407:                                ;
.text:00413407                    mov     eax, [ebp+k]
.text:0041340A                    add     eax, [ebp+i]
.text:0041340D                    mov     [ebp+k], eax
.text:00413410                    mov     ecx, [ebp+i]
.text:00413413                    add     ecx, 1
.text:00413416                    mov     [ebp+i], ecx
.text:00413419
.text:00413419 loc_413419:                                ;
.text:00413419                    cmp     [ebp+i], 0Ah
.text:0041341D                    jge     short loc_413421
.text:0041341F                    jmp     short loc_413407
```

1.2

a4808b6

CodeXplorer v1.2 [minor release]

matrosov released this on 1 Aug 2014 · **15 commits** to master since this release

- Bug fixes
- Added support for IDA v6.6 and Hex-Rays Decompiler (x86) v2.0

Downloads

HexRaysCodeXplorer.plw	122 KB
Source code (zip)	
Source code (tar.gz)	

+

F5

```
1  void __cdecl __noreturn wmain()
2  {
3    int v0; // [sp+0h] [bp-F4h]@1
4    char v1; // [sp+Ch] [bp-E8h]@1
5    int i; // [sp+D0h] [bp-24h]@1
6    int k; // [sp+DCh] [bp-18h]@1
7    int *v4; // [sp+E4h] [bp-10h]@1
8    int v5; // [sp+F0h] [bp-4h]@4
9
10   memset(&v1, 0xCCu, 0xD8u);
11   v4 = &v0;
12   k = 0;
13   for ( i = 0; i < 10; ++i )
14     k += i;
15   _printf("%i\n", k);
16   v5 = 0;
17   _system("PAUSE");
18   _exit(0);
19 }
```

F5超好按

→ 分析記憶體

→ 數位鑑識大神

→ 我超會逆向、破解

→ 我超強

→ IDA Pro好棒棒！

如何理解ASM Opcode並推導出C Code

Asm to C : Push Stack

1. Push 0x12345678

2. Sub Esp,04
   mov [esp], 0x12345678

Asm to C : Pop Stack

1. Pop eax

2. mov eax, [esp]
   Add Esp,04

Asm to C : Jmp (GoTo)

1. Jmp Addr

2. Push Addr
   ret

3. Sub Esp, 04
   mov [Esp], Addr
   ret

Asm to C : 清空

mov eax, 0

→ sub eax,eax //自己減自己，NO GG

→ and eax,0　//eax與0做交集

→ xor eax,eax //自己對自己互斥或，結果為0

Asm to C : 條件跳躍

cmp eax,1

Je 0x12345678

→ sub eax,1

→ Je 0x12345678

Asm to C : 條件跳躍

cmp eax,1

Jg 0x12345678

→ sub eax,1

→ Jg 0x12345678

Asm to C : 條件跳躍

cmp eax,1

Jl 0x12345678

→ sub eax,1

→ Jl 0x12345678

Asm to C : 條件跳躍

cmp eax, 0

Je 0x12345678

→ test eax,eax　//編譯器常用老梗

→ Jz 0x12345678

→ Je 0x12345678

Asm to C : 條件跳躍

cmp eax, 0

Je 0x12345678

→ or eax,eax

→ Je 0x12345678

cmp eax,0

Je 0x12345678

→ and eax,eax

→ Je 0x1234578

Asm to C : Call

MessageBoxA(Arg1, Arg2, Arg3, Arg4);

```
push Arg4
push Arg3
push Arg2
push Arg1
call MessageBoxA
```

Asm to C : Call

MessageBoxA(Arg1, Arg2, Arg3, Arg4);

push Arg4
push Arg3
push Arg2
push Arg1
push 返回地址
Jmp MessageBoxA

Asm to C : Call

MessageBoxA(Arg1, Arg2, Arg3, Arg4);

push Arg4

push Arg3

push Arg2

push Arg1

push 返回地址

push MessageBoxA

ret

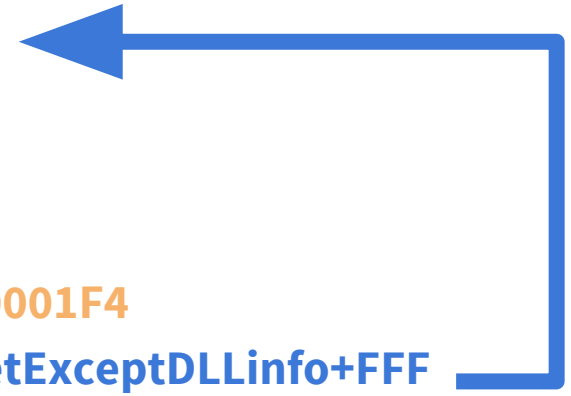
ASM → C : For

```
for (int i = 0; i < 500; i++)
{

/*...Do Anything You Want...*/
}
```

ASM → C : For

Project1._GetExceptDLLinfo+FFA - **xor eax,eax**
Project1._GetExceptDLLinfo+FFC - **mov [ebp-0C],eax**
Project1._GetExceptDLLinfo+FFF - inc [ebp-0C]
.
.
.
Project1._GetExceptDLLinfo+1002- **cmp [ebp-0C],000001F4**
Project1._GetExceptDLLinfo+1009- **jnge Project1._GetExceptDLLinfo+FFF**



ASM → C : For

```
for (int i = 2; i < 500; i++)
{

/*...Do Anything You Want...*/
}
```

ASM → C : For

Project1._GetExceptDLLinfo+FFA - **mov [ebp-0C],00000002**

Project1._GetExceptDLLinfo+1001- inc [ebp-0C]

•

•

•

•

Project1._GetExceptDLLinfo+1004- **cmp [ebp-0C],000001F4**

Project1._GetExceptDLLinfo+100B- **jnge Project1._GetExceptDLLinfo+1001**

IDA Pro Hex-Rays Decompiler解析弱點

```
int _tmain(int argc, _TCHAR* argv[])
{
    printf("Hello World\n");
    system("PAUSE");
    return 0;
}
```

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   _printf("Hello World\n");
4   _system("PAUSE");
5   return 0;
6 }
```

```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
        push Next
        ret
    }
Next:
    printf("Hello World\n");
    system("PAUSE");
    return 0;
}
```

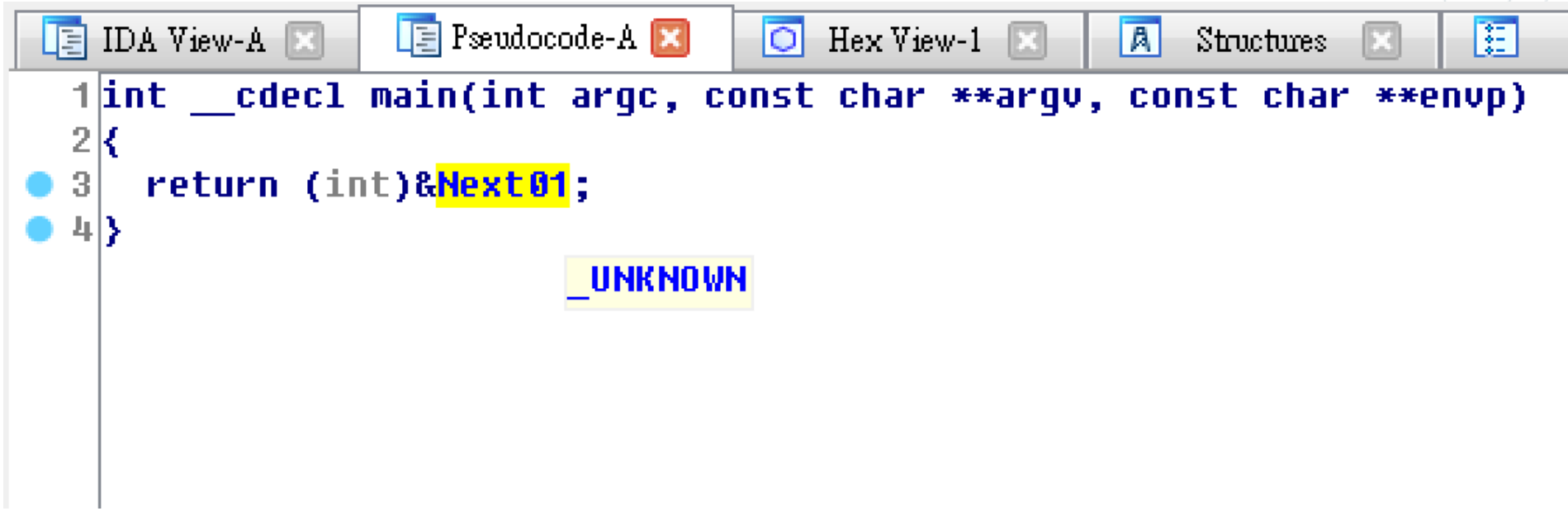
```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   _printf("Hello World\n");
4   _system("PAUSE");
5   return 0;
6 }
```

```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
            sub esp,04
            mov eax,Next
            mov[esp],eax
            ret
    }
Next:
    printf("Hello World\n");
    system("PAUSE");
    return 0;
}
```

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   _printf("Hello World\n");
4   _system("PAUSE");
5   return 0;
6 }
```

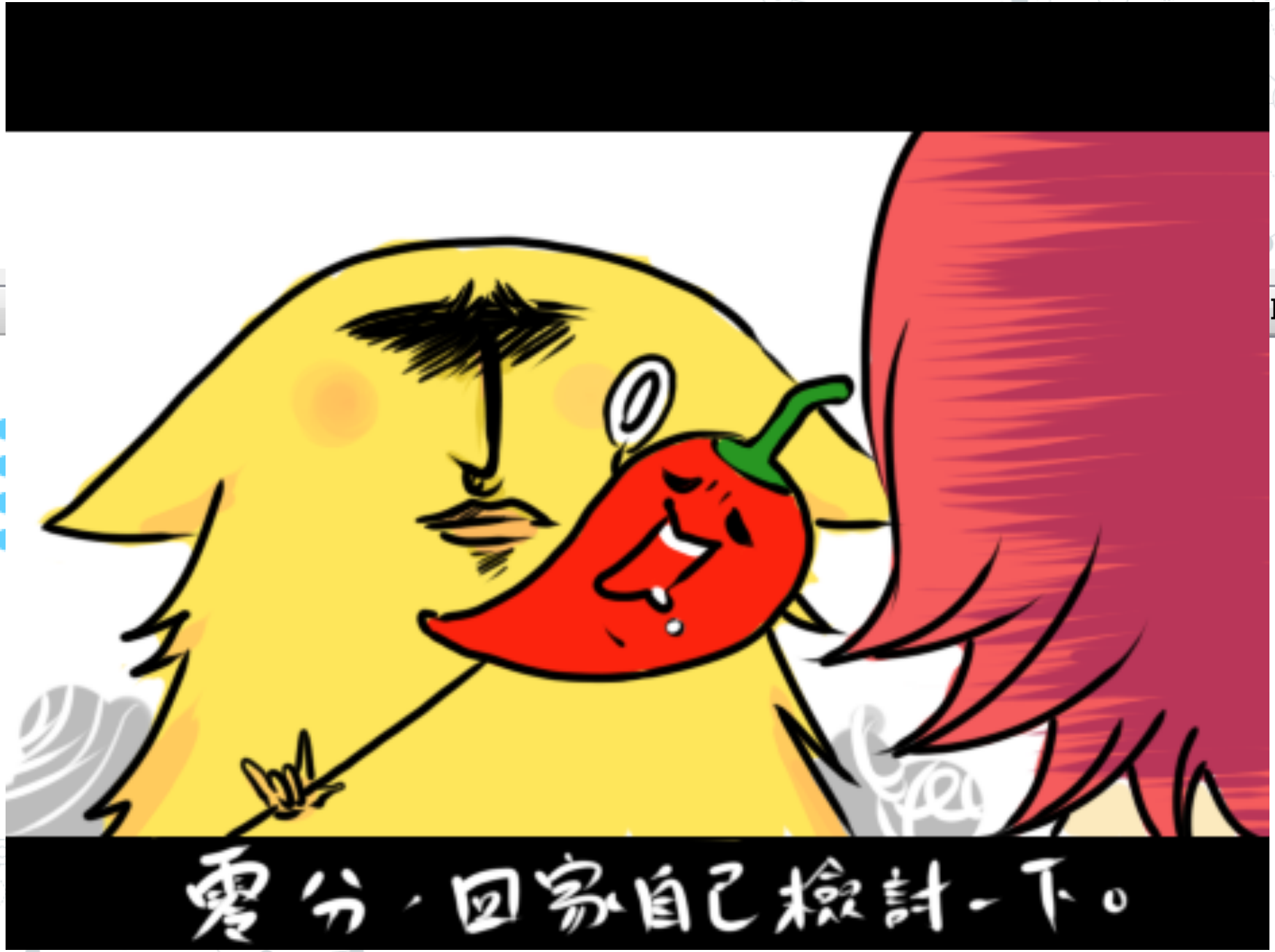
```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
            sub esp,04
            mov eax, Next01
            mov[esp],eax
            ret
    }
Next:
    printf("Hello World\n");
    system("PAUSE");
    return 0;

Next01:
    goto Next;
}
```


IDA View-A
Pseudocode-A
Hex View-1
Structures
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3 return (int)&Next01;
4 }
_UNKNOWN

```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
        push Next01
        ret
    }
Next:
    printf("Hello World\n");
    _tsystem(L"PAUSE");
    return 0;

Next01:
    _asm
    {
        push Next
        ret
    }
    return 0;
}
```


零分，回家自己檢討一下。

小總結

1. IDA Pro 可以識別Push Addr : ret形式跳轉

2. IDA Pro 不善於分析非純粹Jmp、Jl、Je...等條件/直接跳轉的邏輯順序置換手法(只要遇到第一個ret後的事情都不管了只記錄[esp]返回點到哪)

```
void __declspec(naked) H3llW0r1d()
{
    printf("Hello World\n");
    _tsystem(L"PAUSE");
    _asm
    {
        add esp,0x04
        ret
    }
}

unsigned int _next = (unsigned int)H3llW0r1d;
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
        jmp dword ptr[_next]
    }
    return 0;
}
```

```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   return _next();
4 }
```

```
int (*)(void)
```

```
.data:00403020 ; int (*_next)(void)
.data:00403020 ?_next@@3IA     dd offset ?H3llW0r1d@@YAXXZ ; DATA XREF: _main+3↑r
.data:00403020                                             ; H3llW0r1d(void)
.data:00403024                 align 8
```

以上這樣做邏輯順序置換，
IDA很快就暈了

喔，啊所以咧？

這很簡單啊

這可以幹嘛？

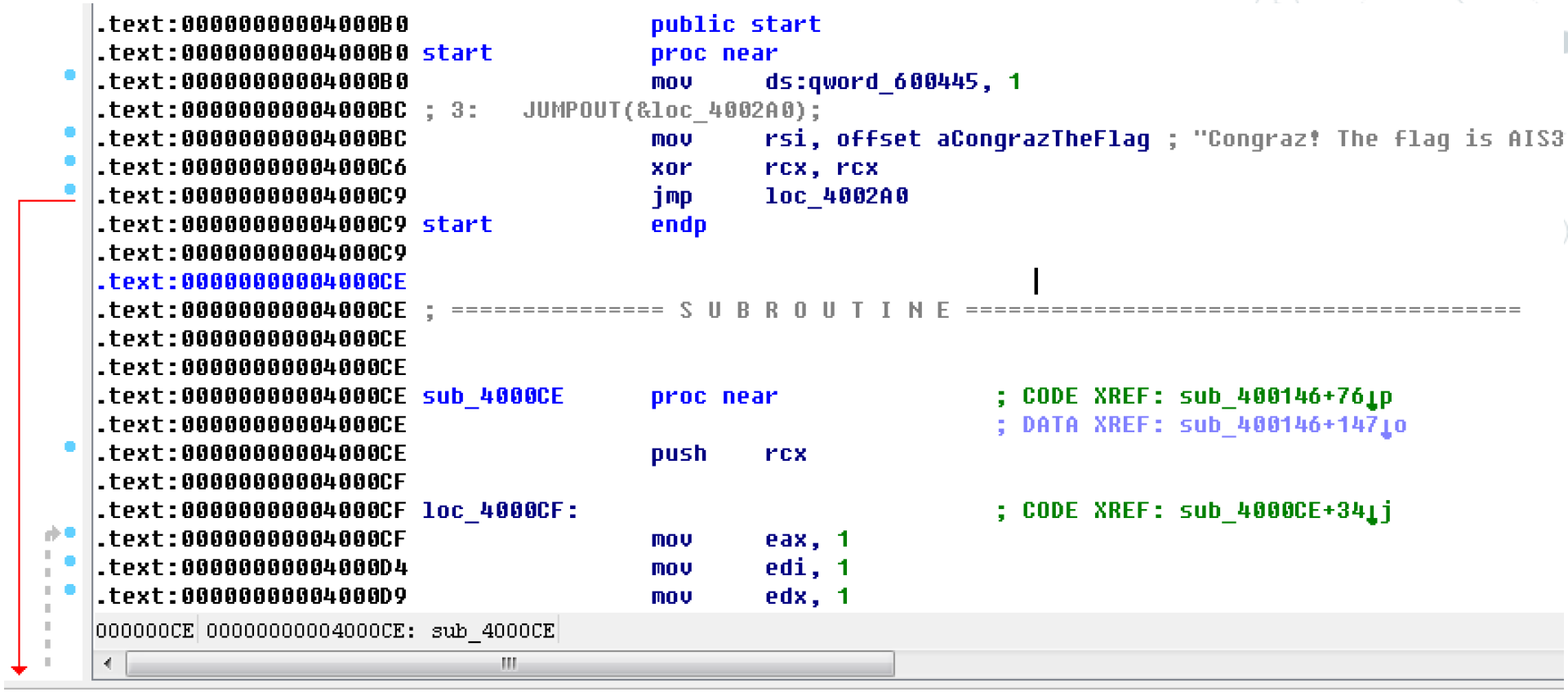
說到這個，不得不提一下

AIS3 Crypto3

新型態資安暑期課程

Advanced Information Security Summer School (AIS3)

```
mashenghaodeMacBook:downloads aaaddress1$ file stupid
stupid: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked,
stripped
mashenghaodeMacBook:downloads aaaddress1$
```


.text:00000000004000B0 public start
.text:00000000004000B0 start proc near
.text:00000000004000B0 mov ds:qword_600445, 1
.text:00000000004000BC ; 3: JUMPOUT(&loc_4002A0);
.text:00000000004000BC mov rsi, offset aCongrazTheFlag ; "Congraz! The flag is AIS3
.text:00000000004000C6 xor rcx, rcx
.text:00000000004000C9 jmp loc_4002A0
.text:00000000004000C9 start endp
.text:00000000004000C9
.text:00000000004000CE
.text:00000000004000CE ; =============== S U B R O U T I N E =======================================
.text:00000000004000CE
.text:00000000004000CE
.text:00000000004000CE sub_4000CE proc near ; CODE XREF: sub_400146+76↓p
.text:00000000004000CE ; DATA XREF: sub_400146+147↓o
.text:00000000004000CE push rcx
.text:00000000004000CF
.text:00000000004000CF loc_4000CF: ; CODE XREF: sub_4000CE+34↓j
.text:00000000004000CF mov eax, 1
.text:00000000004000D4 mov edi, 1
.text:00000000004000D9 mov edx, 1
000000CE 00000000004000CE: sub_4000CE

```
1 void start()
2 {
3   qword_600445 = 1LL;
4   JUMPOUT(&loc_4002A0);
5 }
```

_UNKNOWN

IDA Pro 解析 Jmp

1. 正常情況下，遇到ret視為函數結束
2. 若找查不到ret則以Jmp當作結尾
3. Jmp在函數內來回跳可做解析
4. Jmp往下跳則無法解析回C（但可點擊至該地址）
5. Jmp往回跳則無法明確解析地址

說到這個，又不得不提一下

Flare-On CTF 2015

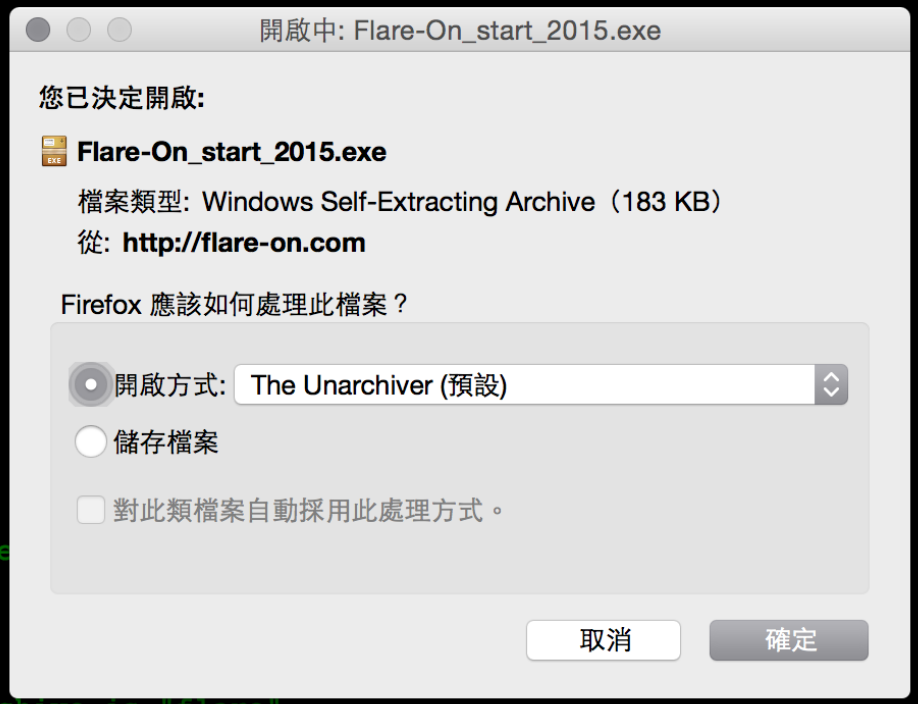
The F L A R E On Challenge

```
Enter a command or type "help" for help.
[user@server ~]$ help
FOO bash, version 1.0.0
These shell commands are defined internally.
Type help to see this list.

cd [tab] - Change to the target tab
clear - Clear the screen
help - Display this list
ls - List the contents of the current tab
[user@server ~]$ ls
 Overview          Instructions          Resources
[user@server ~]$ Instructions
-bash: instructions: command not found
[user@server ~]$ cd Instructions
[user@server Instructions]$ ls
It's simple: Analyze the sample, find the key.
Each key is an email address. Send an email to the
Complete all the puzzles and win a prize.

To start The FLARE On Challenge 2015 download the

The password for each challenge that is a .zip archive is "flare".
[user@server Instructions]$
```



FLARE-On Challenge #1 Completed!

收件匣 x

[redacted]@flare-on.com

7月30日

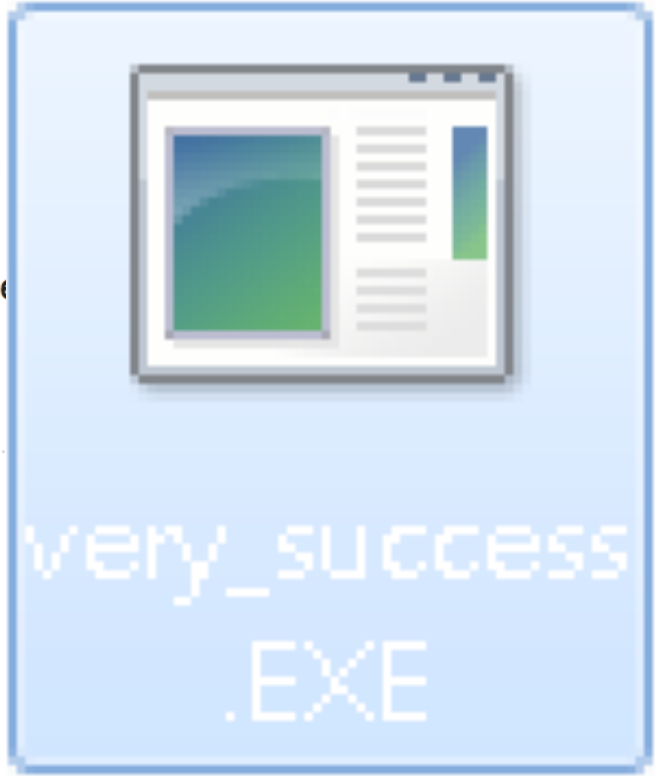
寄給 我

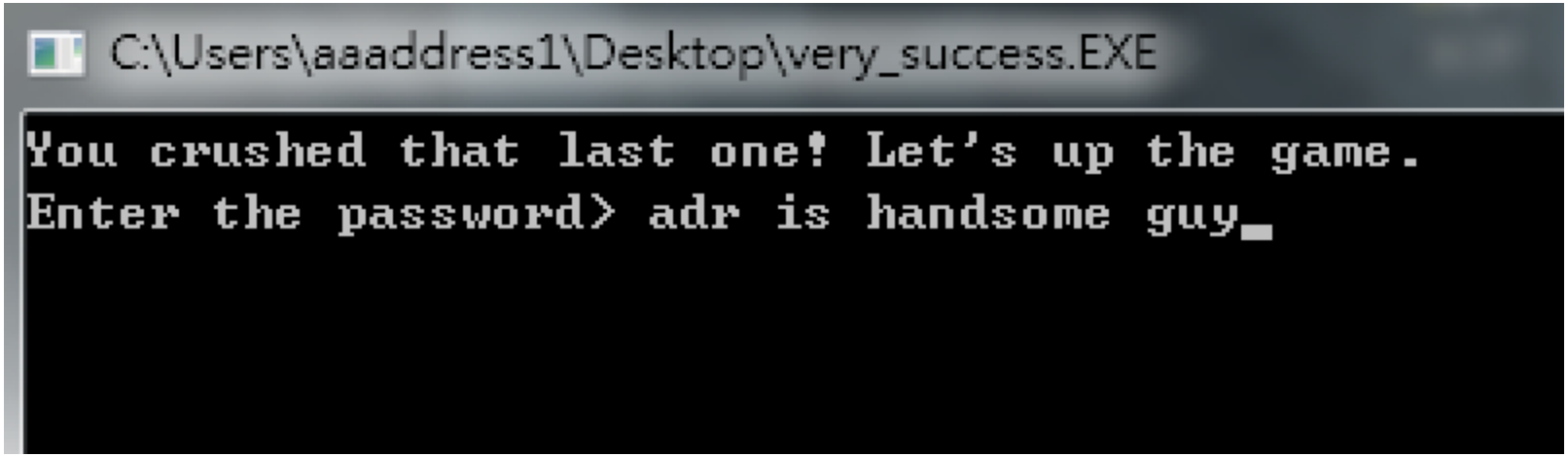
Congrats! I've attached the next [redacted]he password to this zip archive is "flare".

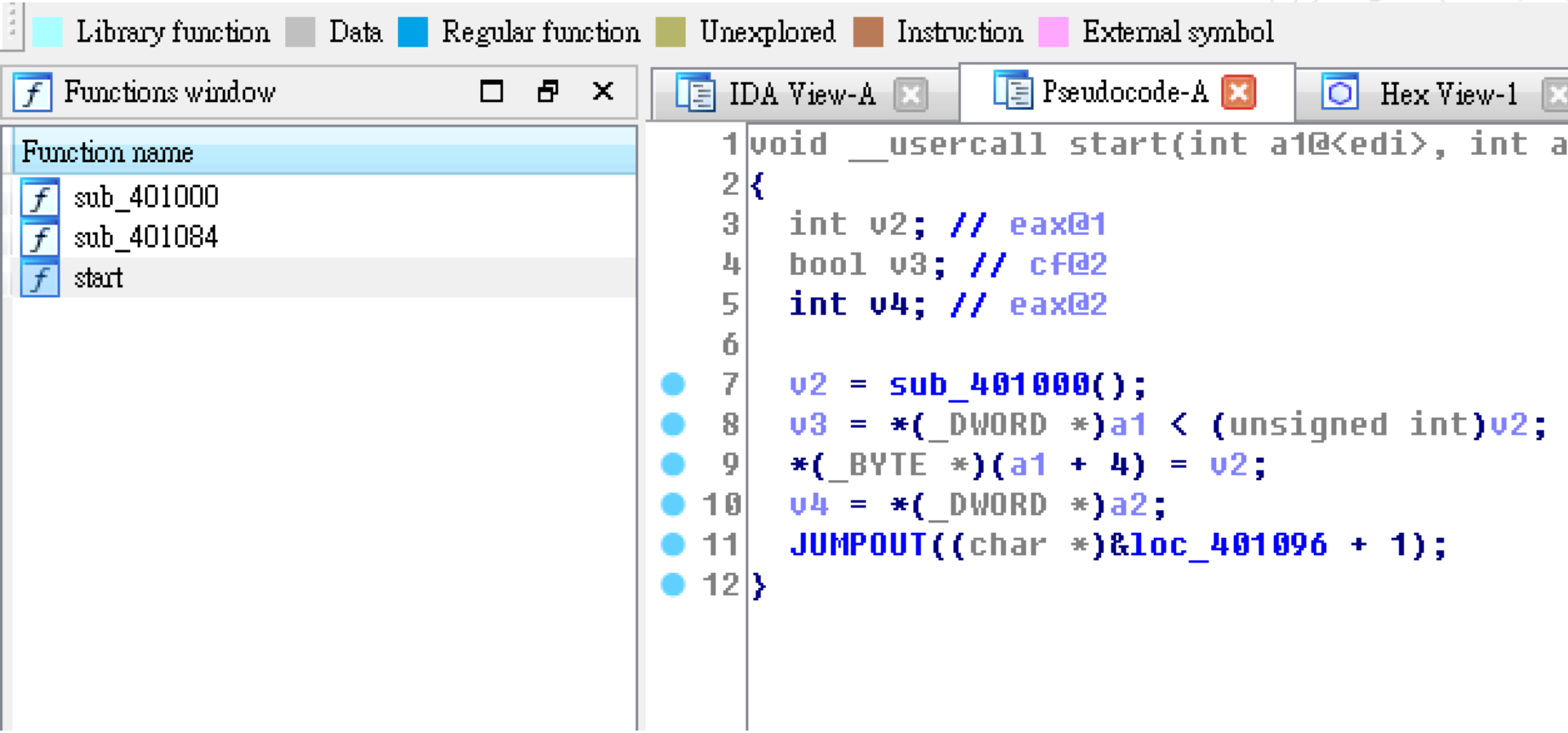
This challenge looks a lot like th[redacted]ne out too, Good luck!

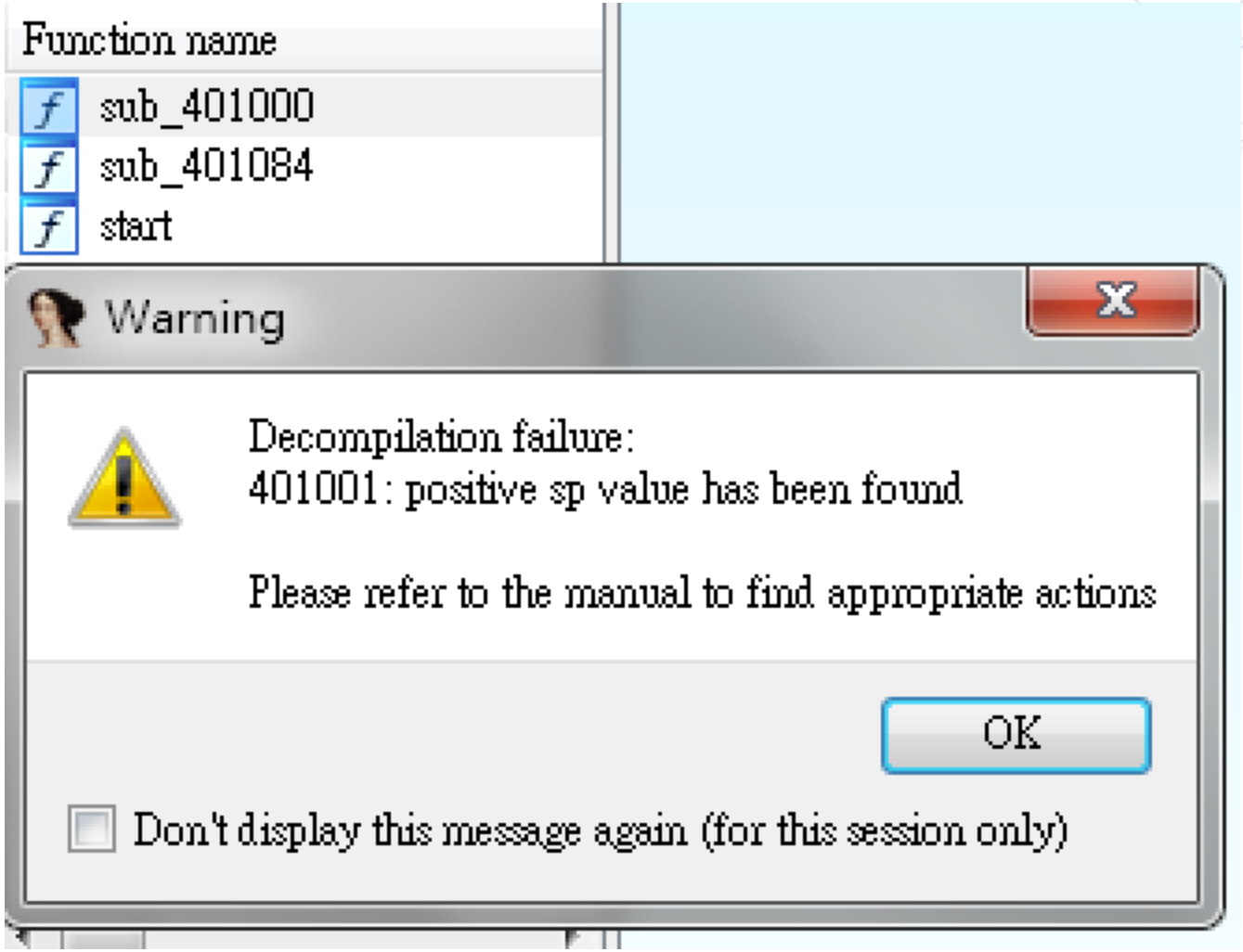
-FLARE

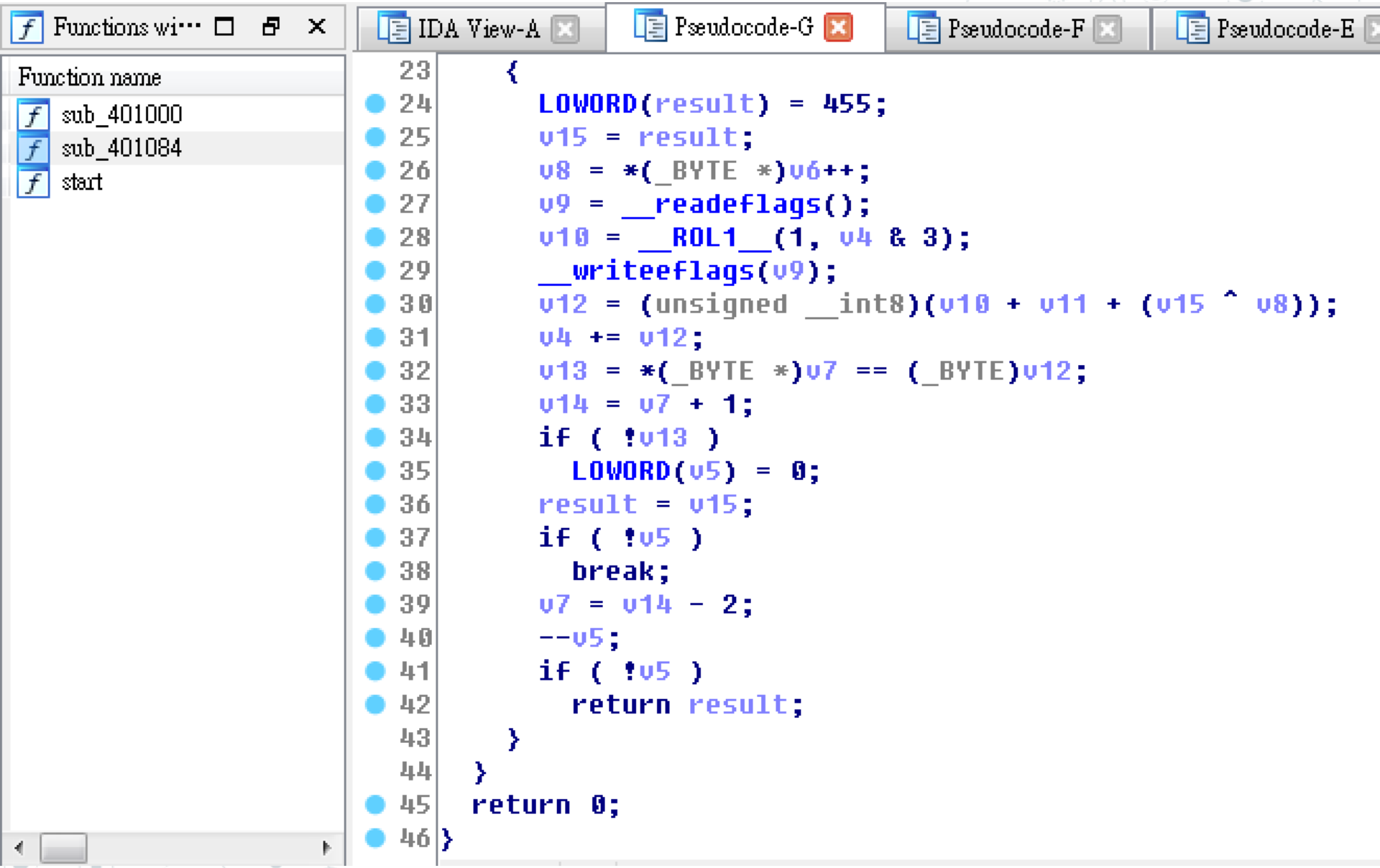
599EA8F84AD97...




C:\Users\aaaddress1\Desktop\very_success.EXE
You crushed that last one! Let's up the game.
Enter the password> adr is handsome guy


Library function
Data
Regular function
Unexplored
Instruction
External symbol
Functions window
Function name
sub_401000
sub_401084
start
IDA View-A
Pseudocode-A
Hex View-1
1 void __usercall start(int a1@<edi>, int a
2 {
3 int v2; // eax@1
4 bool v3; // cf@2
5 int v4; // eax@2
6
7 v2 = sub_401000();
8 v3 = *(_DWORD *)a1 < (unsigned int)v2;
9 *(_BYTE *)(a1 + 4) = v2;
10 v4 = *(_DWORD *)a2;
11 JUMPOUT((char *)&loc_401096 + 1);
12 }


Function name
sub_401000
sub_401084
start
Warning
Decompilation failure:
401001: positive sp value has been found
Please refer to the manual to find appropriate actions
OK
Don't display this message again (for this session only)


Functions wi···
Function name
sub_401000
sub_401084
start
IDA View-A
Pseudocode-G
Pseudocode-F
Pseudocode-E
23 {
24 LOWORD(result) = 455;
25 v15 = result;
26 v8 = *(_BYTE *)v6++;
27 v9 = __readeflags();
28 v10 = __ROL1__(1, v4 & 3);
29 __writeeflags(v9);
30 v12 = (unsigned __int8)(v10 + v11 + (v15 ^ v8));
31 v4 += v12;
32 v13 = *(_BYTE *)v7 == (_BYTE)v12;
33 v14 = v7 + 1;
34 if (!v13)
35 LOWORD(v5) = 0;
36 result = v15;
37 if (!v5)
38 break;
39 v7 = v14 - 2;
40 --v5;
41 if (!v5)
42 return result;
43 }
44 }
45 return 0;
46 }

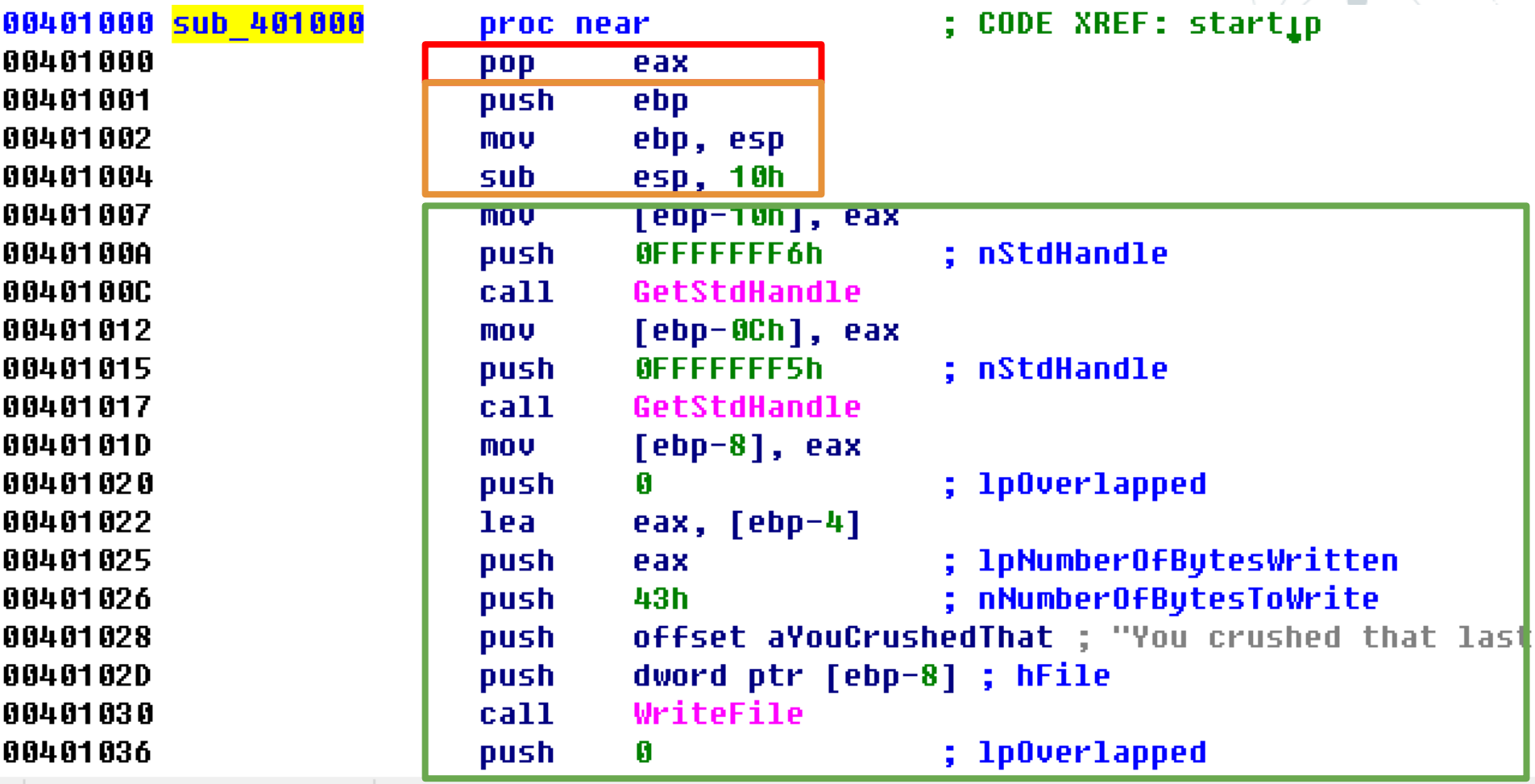
```
1  void __usercall start(int a1@<edi>, int a2@<esi>)
2  {
3    int v2; // eax@1
4    bool v3; // cf@2
5    int v4; // eax@2
6
7    v2 = sub_401000();
8    v3 = *(_DWORD *)a1 < (unsigned int)v2;
9    *(_BYTE *)(a1 + 4) = v2;
10   v4 = *(_DWORD *)a2;
11   JUMPOUT((char *)&loc_401096 + 1);
12 }
```

```
.text:004010DB                  mov     esp, ebp
.text:004010DD                  pop     ebp
.text:004010DE                  retn
.text:004010DE sub_401084       endp
.text:004010DE
.text:004010DF ; 6:   v2 = sub_401000();
.text:004010DF
.text:004010DF ; =============== S U B R O U T I N E =======================
.text:004010DF
.text:004010DF
.text:004010DF                  public start
.text:004010DF start            proc near
.text:004010DF                  call    sub_401000
.text:004010E4 ; 7:   v3 = *(_DWORD *)a1 < (unsigned int)v2;
.text:004010E4                  scasd
.text:004010E5 ; 8:   *(_BYTE *)(a1 + 4) = v2;
.text:004010E5                  stosb
.text:004010E6 ; 9:   v4 = *(_DWORD *)a2;
.text:004010E6                  lodsd
                                [illegible]
.text:004010E7                  jmp     short near ptr loc_401096+1
                                [illegible]
.text:004010E7
.text:004010E7 ; ---------------------------------------------------------------
```

```
1  void __usercall start(int a1@<edi>, int a2@<esi>)
2  {
3    int v2; // eax@1
4    bool v3; // cf@2
5    int v4; // eax@2
6
7    v2 = sub_401000();
8    v3 = *(_DWORD *)a1 < (unsigned int)v2;
9    *(_BYTE *)(a1 + 4) = v2;
10   v4 = *(_DWORD *)a2;
11   JUMPOUT((char *)&loc_401096 + 1);
12 }
```

IDA Pro在反組譯解析時,
解析到第一個ret或者非迴圈的Jmp即當作函數尾

```
.text:004010DB                  mov     esp, ebp
.text:004010DD                  pop     ebp
.text:004010DE                  retn
.text:004010DE sub_401084       endp
.text:004010DE
.text:004010DF ; 6:   v2 = sub_401000();
.text:004010DF
.text:004010DF ; =============== S U B R O U T I N E =======================
.text:004010DF
.text:004010DF
.text:004010DF                  public start
.text:004010DF start            proc near
.text:004010DF                  call    sub_401000
.text:004010E4 ; 7:   v3 = *(_DWORD *)a1 + (unsigned int)v2;
.text:004010E4                  scasd
.text:004010E5 ; 8:   *(_BYTE *)(a1 + 4) = v2;
.text:004010E5                  stosb
.text:004010E6 ; 9:   v4 = *(_DWORD *)a2;
.text:004010E6                  lodsd
.text:004010E7 ; 10:   JUMPOUT((char *)&loc_401096 + 1);
.text:004010E7                  jmp     short near ptr loc_401096+1
.text:004010E7 start            endp
.text:004010E7
.text:004010E7 ; ---------------------------------------------------------------
```


00401000 sub_401000 proc near ; CODE XREF: start↓p
00401000 pop eax
00401001 push ebp
00401002 mov ebp, esp
00401004 sub esp, 10h
00401007 mov [ebp-10h], eax
0040100A push 0FFFFFFF6h ; nStdHandle
0040100C call GetStdHandle
00401012 mov [ebp-0Ch], eax
00401015 push 0FFFFFFF5h ; nStdHandle
00401017 call GetStdHandle
0040101D mov [ebp-8], eax
00401020 push 0 ; lpOverlapped
00401022 lea eax, [ebp-4]
00401025 push eax ; lpNumberOfBytesWritten
00401026 push 43h ; nNumberOfBytesToWrite
00401028 push offset aYouCrushedThat ; "You crushed that last
0040102D push dword ptr [ebp-8] ; hFile
00401030 call WriteFile
00401036 push 0 ; lpOverlapped

```
00401054                push    dword ptr [ebp-4]
00401057                push    offset unk_402159
0040105C                push    dword ptr [ebp-10h]
0040105F                call    sub_401084
00401064                add     esp, 0Ch
00401067                test    eax, eax
00401069                jz      short loc_401072
0040106B                push    offset aYouAreSuccess ; "You are success\r\n"
00401070                jmp     short loc_401077
00401072 ; ---------------------------------------------------------------------------
00401072
00401072 loc_401072:                            ; CODE XREF: sub_401000+69↑j
00401072                push    offset aYouAreFailure ; "You are failure\r\n"
00401077
00401077 loc_401077:                            ; CODE XREF: sub_401000+70↑j
00401077                push    dword ptr [ebp-8] ; hFile
0040107A                call    WriteFile
00401080                mov     esp, ebp
00401082                pop     ebp
00401083                retn
00401083 sub_401000     endp
00401083
00401084
00401084 ; =============== S U B R O U T I N E =======================================
```

```
00401084 ; =============== S U B R O U T I N E =======================================
00401084
00401084 ; Attributes: bp-based frame
00401084
00401084 sub_401084      proc near               ; CODE XREF: sub_401000+5F↑p
00401084
00401084 var_C           = byte ptr -0Ch
00401084 arg_0           = dword ptr  8
00401084 arg_4           = dword ptr  0Ch
00401084 arg_8           = dword ptr  10h
00401084
00401084                 push    ebp
00401085                 mov     ebp, esp
00401087                 sub     esp, 0
0040108A                 push    edi
0040108B                 push    esi
0040108C                 xor     ebx, ebx
0040108E                 mov     ecx, 25h
00401093                 cmp     [ebp+arg_8], ecx
00401096
00401096 loc_401096:                             ; CODE XREF: start+8↓j
00401096                 jl      short loc_4010D7
00401098                 mov     esi, [ebp+arg_4]
0040109B                 mov     edi, [ebp+arg_0]
```

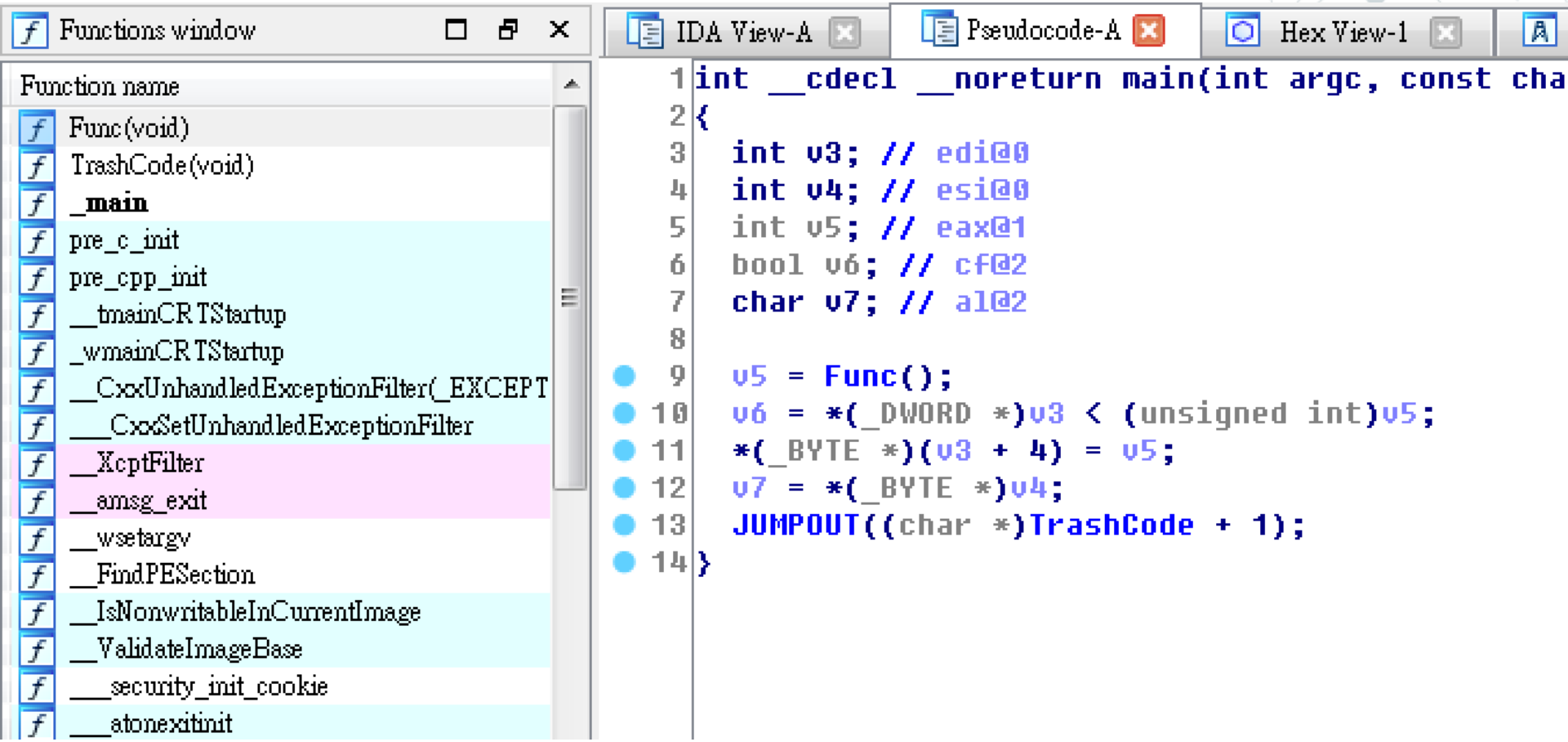
```
.text:004010DB                    mov     esp, ebp
.text:004010DD                    pop     ebp
.text:004010DE                    retn
.text:004010DE sub_401084         endp
.text:004010DE
.text:004010DF ; 6:   v2 = sub_401000();
.text:004010DF
.text:004010DF ; =============== S U B R O U T I N E =======================
.text:004010DF
.text:004010DF
.text:004010DF                    public start
.text:004010DF start              proc near
.text:004010DF                    call    sub_401000
```

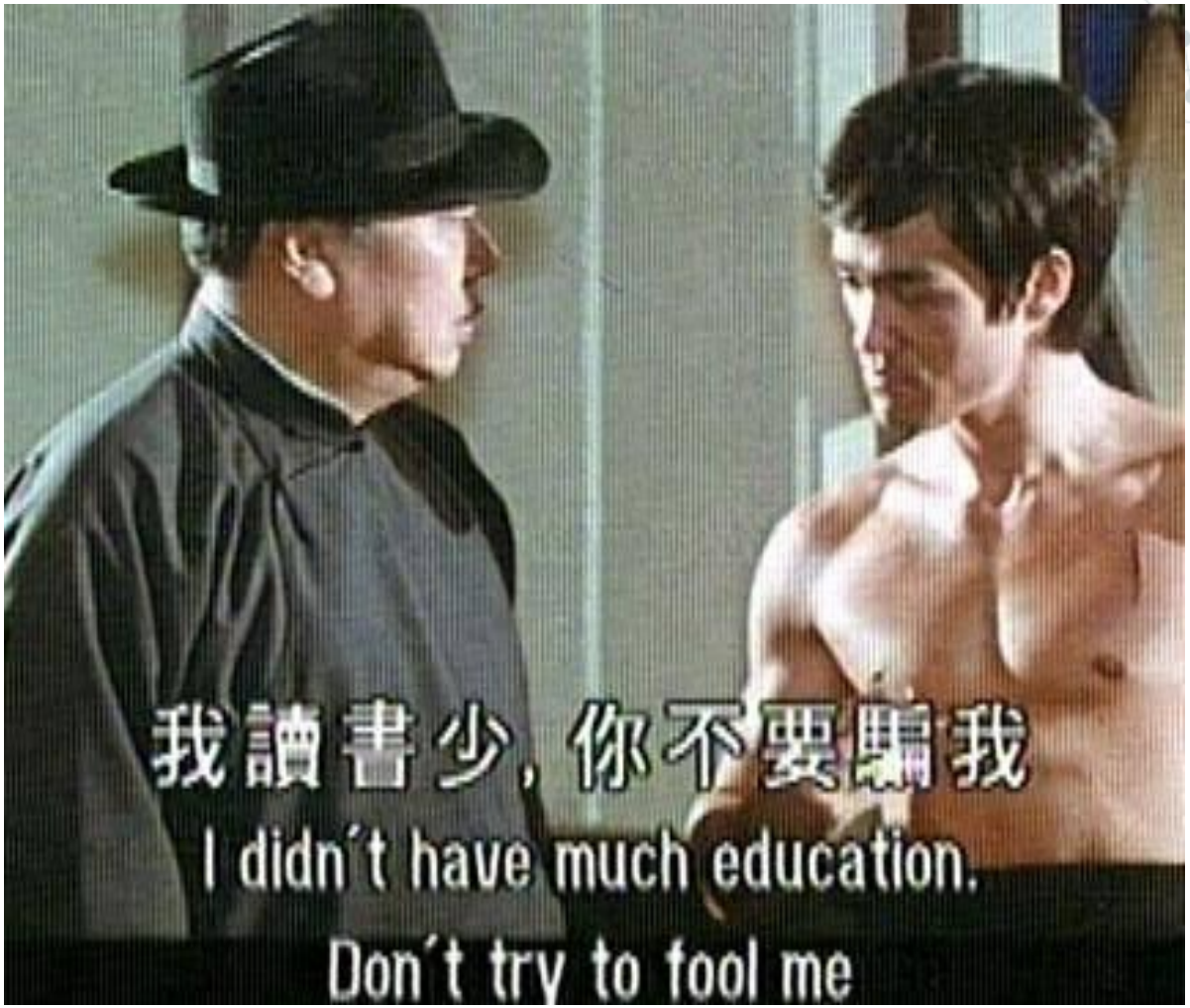
```
.text:004010E7
.text:004010E7 ; ---------------------------------------------------------
```

```
int __declspec(naked)Func()
{
    _asm
    {
        pop eax
    }
    printf("Hello World\n");
    /*ANYTHING YOU WANT*/
    _asm
    {
        mov esp,ebp
        pop ebp
        ret
    }
}

void TrashCode()
{
    for (int a, b, c, d, e, f, g, i; ; i = a*b*c*(d<<e*f*g) + 1) for(int j,k,l,m,n,o,p;;);
    /*USED FOR TRICKING CRACKER*/
}

int _tmain(int argc, _TCHAR* argv[])
{
    Func();
    _asm
    {
        scasd
        stosb
        lodsb
        jmp TrashCode+1
    }
}
```


Functions window
Function name
Func(void)
TrashCode(void)
_main
pre_c_init
pre_cpp_init
__tmainCRTStartup
_wmainCRTStartup
__CxxUnhandledExceptionFilter(_EXCEPT
___CxxSetUnhandledExceptionFilter
__XcptFilter
__amsg_exit
__wsetargv
__FindPESection
__IsNonwritableInCurrentImage
__ValidateImageBase
___security_init_cookie
___atonexitinit
IDA View-A
Pseudocode-A
Hex View-1
1 int __cdecl __noreturn main(int argc, const cha
2 {
3 int v3; // edi@0
4 int v4; // esi@0
5 int v5; // eax@1
6 bool v6; // cf@2
7 char v7; // al@2
8
9 v5 = Func();
10 v6 = *(_DWORD *)v3 < (unsigned int)v5;
11 *(_BYTE *)(v3 + 4) = v5;
12 v7 = *(_BYTE *)v4;
13 JUMPOUT((char *)TrashCode + 1);
14 }


我讀書少，你不要騙我
I didn´t have much education.
Don´t try to fool me

“

我們要怎麼做出詐欺的Code？


茶或
咖啡?
咖啡
答錯了,
這是茶.
譯：好色龍
VIA 9GAG.COM

唐鳳
@audreyt

Following

@clkao @ETBlue @evenwu 講個秘訣：講「講個秘訣前，先講講個秘訣，再講秘訣」前，先講「講個秘訣：講『講個秘訣前，先講講個秘訣，再講秘訣』前，先講『講個秘訣』，再講『講個秘訣前，先講講個秘訣，再講秘訣』」，再講「講個秘訣前，先講講個秘訣，再講秘訣」。

```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
        xor eax,eax
        xor ebx,ebx
        inc eax
        inc ebx
        inc ecx
        inc edx
        add[esp],eax
        sub[esp],ebx
        push 00
    }
    printf("Hello World\n");
    _asm
    {
        add esp,04
    }
    system("PAUSE");
    return 0;

}
```

```
.text:00401000
.text:00401000                 push    ebx
.text:00401001 ; 2:    _printf("Hello World\n");
.text:00401001                 xor     eax, eax
.text:00401003                 xor     ebx, ebx
.text:00401005                 inc     eax
.text:00401006                 inc     ebx
.text:00401007                 inc     ecx
.text:00401008                 inc     edx
.text:00401009                 add     [esp+4+var_4], eax
.text:0040100C                 sub     [esp+4+var_4], ebx
.text:0040100F                 push    0
.text:00401011                 push    offset Format   ; "Hello World\n"
.text:00401016                 call    ds:__imp__printf
.text:0040101C ; 3:    _system("PAUSE");
.text:0040101C                 add     esp, 4
```


```
1 int __cdecl main(int argc, const char **argv, const char **envp)
2 {
3   _printf("Hello World\n");
4   _system("PAUSE");
5   return 0;
6 }
```

IDA解析小特性:

IDA Pro Decomilper對純暫存器操作不敏感，只對函數內變數之操作敏感。(例如說:eax,ebx,ecx... etc)

IDA Pro 對單個函數反組譯時(回推C)從函數頭開始往下遇到第一個ret或者非迴圈的Jmp即當作函數尾。

但是IDA Pro相當重視Push跟Pop兩指令在函數內對堆棧的變化。(用於確認函數是否平衡堆疊，如果不平衡就直接不做解析了)

```
int _tmain(int argc, _TCHAR* argv[])
{
    _asm
    {
        push Next
        ret
    }
Next:
    printf("Hello World\n");
    system("PAUSE");
    return 0;
}
```

```
1 int
2 {
3     _p
4     _s
5     re
6 }
```

IDA解析小特性:

IDA Pro Decomilper對純暫存器操作不敏感，只對函數內變數之操作敏感。(例如說:eax,ebx,ecx,... etc)

咦...

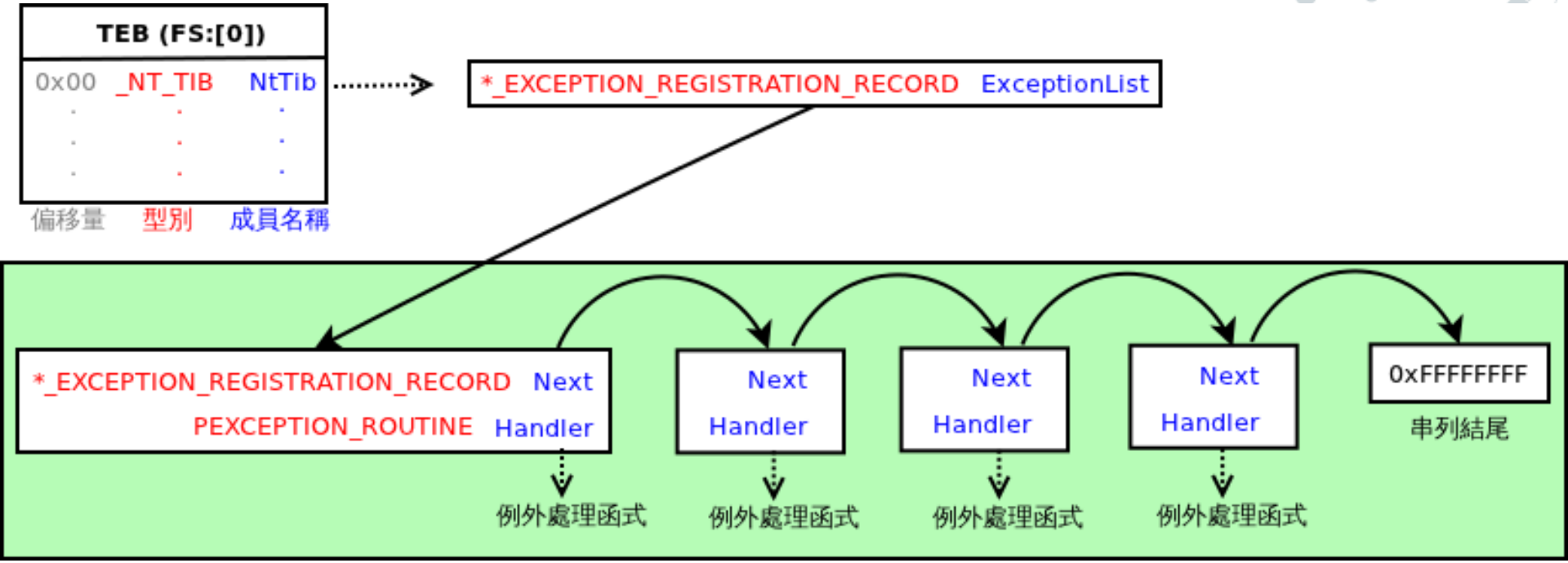
我們的C++內的Try不就是...

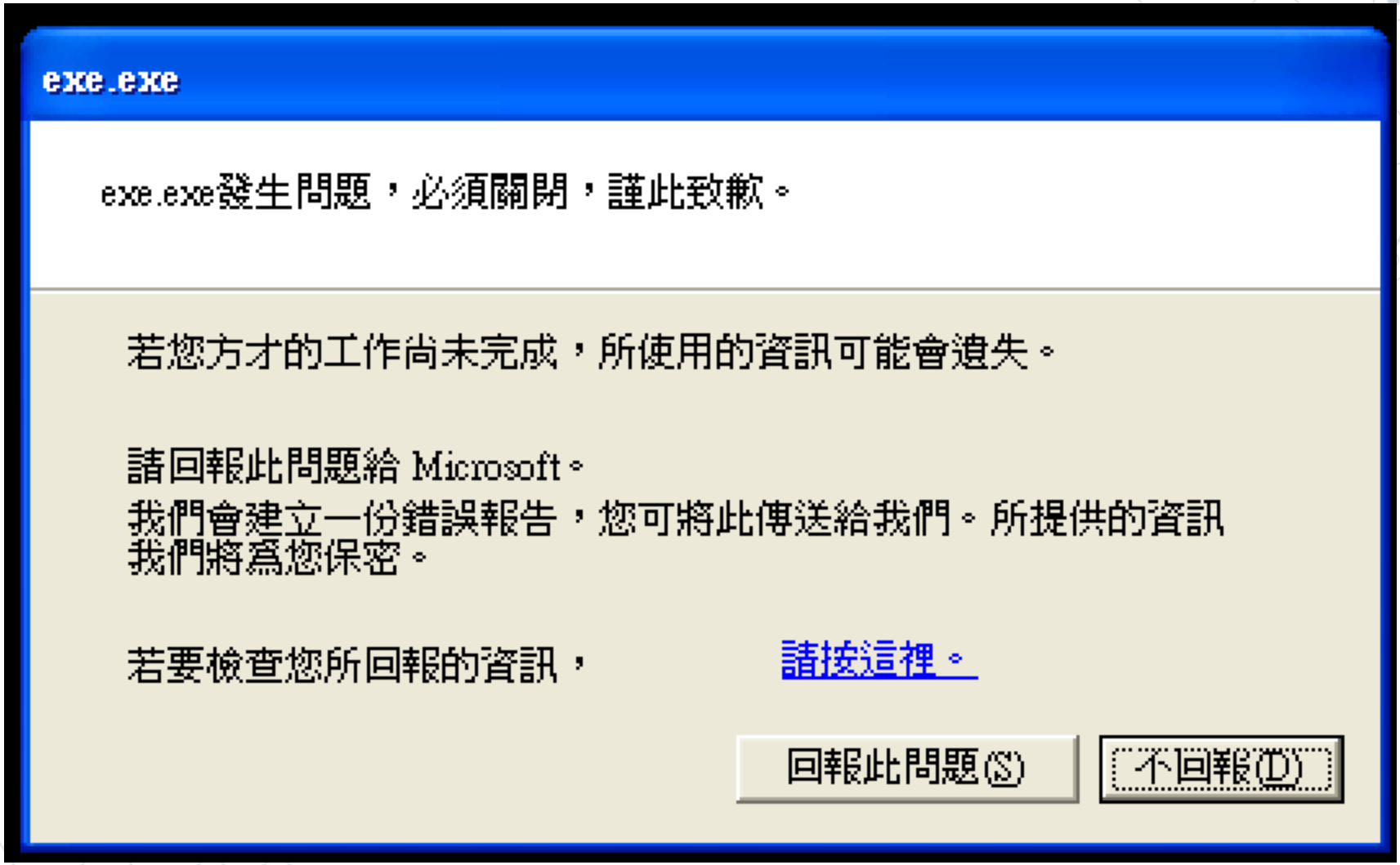

參考
組態屬性
一般
偵錯
VC++ 目錄
C/C++
一般
最佳化
前置處理器
啟用字串共用
啟用最少重建
否 (/Gm-)
啟用 C++ 例外狀況
是，但有 SEH 例外狀況 (/EHa)
較小類型檢查
基礎執行階段檢查
執行階段程式庫
結構成員對齊
安全性檢查
是，但有 SEH 例外狀況 (/EHa)
是 (/EHsc)
是，但有 Extern C 函式 (/EHs)
否
<從父代或專案預設值繼承>

```
push Handler
mov eax, fs:[0]
push eax
mov fs:[0],esp
```

```
EXCEPTION_DISPOSITION
__cdecl
_except_handler(
        struct _EXCEPTION_RECORD *ExceptionRecord,
        void * EstablisherFrame,
        struct _CONTEXT *ContextRecord,
        void * DispatcherContext )
    {
      printf( "Hello World”);
      return ExceptionContinueSearch;
    }
```

```
mov eax, [esp]
mov fs:[0], eax
add esp, 8
```


TEB (FS:[0])
0x00 _NT_TIB NtTib
偏移量 型別 成員名稱
*_EXCEPTION_REGISTRATION_RECORD ExceptionList
*_EXCEPTION_REGISTRATION_RECORD Next
PEXCEPTION_ROUTINE Handler
Next
Handler
Next
Handler
Next
Handler
0xFFFFFFFF
串列結尾
例外處理函式
例外處理函式
例外處理函式
例外處理函式


exe.exe
exe.exe發生問題，必須關閉，謹此致歉。
若您方才的工作尚未完成，所使用的資訊可能會遺失。
請回報此問題給 Microsoft。
我們會建立一份錯誤報告，您可將此傳送給我們。所提供的資訊我們將爲您保密。
若要檢查您所回報的資訊，
請按這裡。
回報此問題(S)
不回報(D)

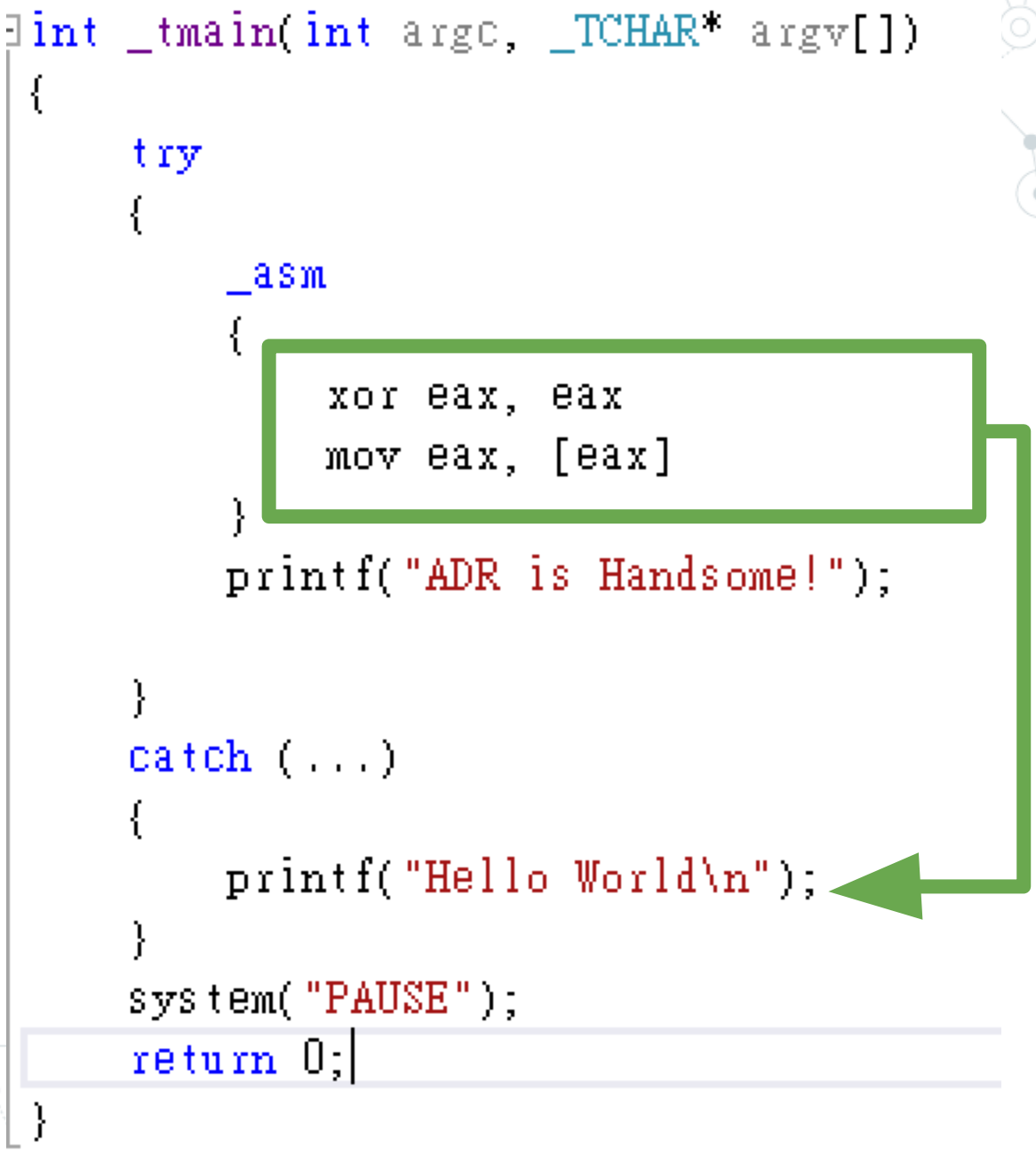
```
int _tmain(int argc, _TCHAR* argv[])
{
    try
    {
        printf("ADR is Handsome!");
    }
    catch (...)
    {
        printf("Hello World");
    }
    _tsystem(L"PAUSE");
    return 0;
}
```

```
004113D0                push    ebp
004113D1                mov     ebp, esp
004113D3                push    0FFFFFFFFh
004113D5                push    offset __ehhandler$_wma
004113DA                mov     eax, large fs:0
004113E0                push    eax
004113E1                push    ecx
004113E2                sub     esp, 0C0h
004113E8                push    ebx
004113E9                push    esi
004113EA                push    edi
004113EB ; 8:   memset(&v2, 0xCCu, 0xC0u);
004113EB                lea     edi, [ebp+var_D0]
004113F1                mov     ecx, 30h
004113F6                mov     eax, 0CCCCCCCCh
004113FB                rep stosd
004113FD ; 9:   v1 = (unsigned int)&savedregs ^ __securi
004113FD                mov     eax, ___security_cookie
00411402                xor     eax, ebp
00411404                push    eax
00411405                lea     eax, [ebp+var_C]
00411408                mov     large fs:0, eax
```

```
004113FD                mov     eax, ___security_cookie
00411402                xor     eax, ebp
00411404                push    eax
00411405                lea     eax, [ebp+var_C]
00411408                mov     large fs:0, eax
0041140E ; 10:   v3 = &v1;
0041140E                mov     [ebp+var_10], esp
00411411 ; 11:   v4 = 0;
00411411                mov     [ebp+var_4], 0
00411418 ; 12:   _printf("ADR is Handsome!");
00411418                mov     esi, esp
0041141A                push    offset Format   ; "ADR is Handsome!"
0041141F                call    ds:__imp__printf
00411425                add     esp, 4
00411428                cmp     esi, esp
0041142A                call    j___RTC_CheckEsp
0041142F                jmp     short loc_41144E
00411431 ; ---------------------------------------------------------------------
00411431
00411431 __catch$_wmain$0:                      ; DATA XREF: .rdata:0
00411431                mov     esi, esp
00411433                push    offset aHelloWorld ; "Hello World"
00411438                call    ds:__imp__printf
0041143E                add     esp, 4
00411441                cmp     esi, esp
00411443                call    j___RTC_CheckEsp
00411448                mov     eax, offset $LN7
0041144D                retn
```

```
 1 int __cdecl wmain()
 2 {
 3   int v1; // [sp+0h] [bp-DCh]@1
 4   char v2; // [sp+Ch] [bp-D0h]@1
 5   int *v3; // [sp+CCh] [bp-10h]@
 6   int v4; // [sp+D8h] [bp-4h]@1
 7
 8   memset(&v2, 0xCCu, 0xC0u);
 9   v3 = &v1;
10   v4 = 0;
11   _printf("ADR is Handsome!");
12   v4 = -1;
13   _system("PAUSE");
14   return 0;
15 }
```

```
int _tmain(int argc, _TCHAR* argv[])
{
    try
    {
        printf("ADR is Handsome!");
    }
    catch (...)
    {
        printf("Hello World");
    }
    _tsystem(L"PAUSE");
    return 0;
}
```


int _tmain(int argc, _TCHAR* argv[])
{
try
{
_asm
{
xor eax, eax
mov eax, [eax]
}
printf("ADR is Handsome!");
}
catch (...)
{
printf("Hello World\n");
}
system("PAUSE");
return 0;
}

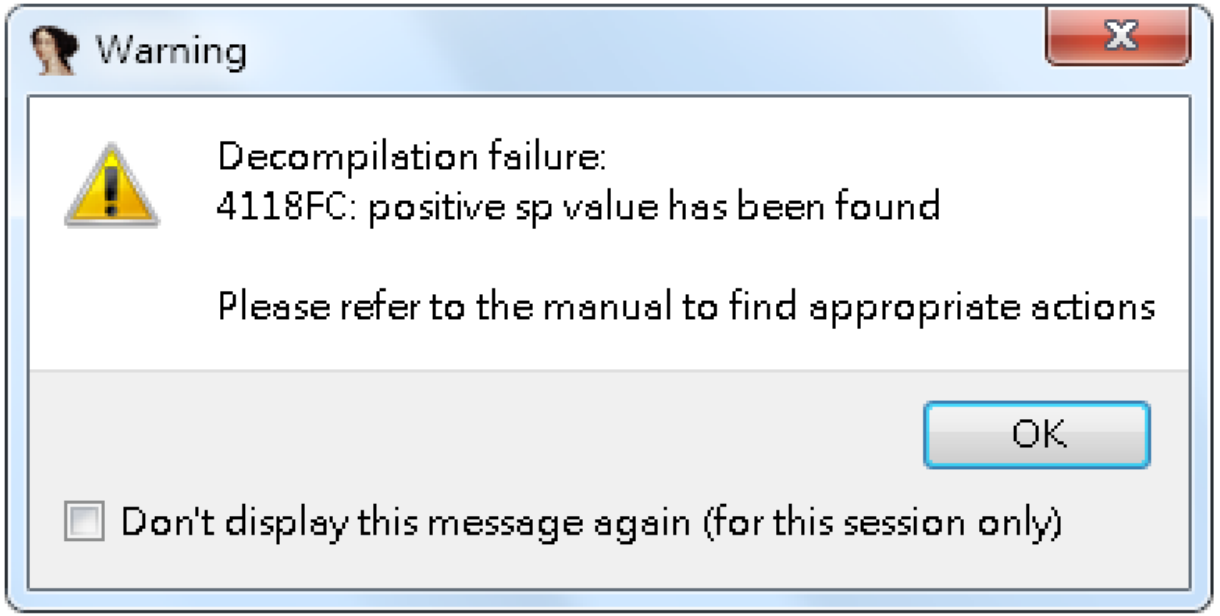
```
 1 int __cdecl wmain()
 2 {
 3   int v1; // [sp+0h] [bp-DCh]@1
 4   char v2; // [sp+Ch] [bp-D0h]@1
 5   int *v3; // [sp+CCh] [bp-10h]@
 6   int v4; // [sp+D8h] [bp-4h]@1
 7
 8   memset(&v2, 0xCCu, 0xC0u);
 9   v3 = &v1;
10   v4 = 0;
11   _printf("ADR is Handsome!");
12   v4 = -1;
13   _system("PAUSE");
14   return 0;
15 }
```


C:\Users\aaaddress1\Desktop\Hello\Console\Debug\Console.exe
Hello World
請按任意鍵繼續 . . .

```
int _tmain(int argc, _TCHAR* argv[])
{
    try
    {
        _asm
        {
            xor eax, eax
            mov eax, [eax]
        }
        printf("ADR is Handsome!");
        _asm
        {
            ret 0xFF
        }
    }
    catch (...)
    {
        printf("Hello World\n");
    }
    system("PAUSE");
    return 0;
}
```

```
1  int __userpurge wmain@<eax>(int a1@<eax>, int a2@<ebx>, int a3@<edi>, int a4@<esi>, int argc, wchar_t **argv, int a7, i
2  {
3    int v65; // [sp-Ch] [bp-DCh]@1
4    int v66; // [sp-8h] [bp-D8h]@1
5    int v67; // int v65; // [sp-Ch] [bp-DCh]@1
6    char v68; // [sp+0h] [bp-D0h]@1
7    int *v69; // [sp+C0h] [bp-10h]@1
8    int v70; // [sp+C4h] [bp-Ch]@1
9    int (*v71)(); // [sp+C8h] [bp-8h]@1
10   int v72; // [sp+CCh] [bp-4h]@1
11
12   v71 = _ehhandler__wmain;
13   v70 = a1;
14   v67 = a2;
15   v66 = a4;
16   v65 = a3;
17   memset(&v68, 0xCCu, 0xC0u);
18   v69 = &v65;
19   v72 = 0;
20   return _printf("ADR is Handsome!");
21 }
```

```
int __cdecl wmainCRTStartup()
{
  j____security_init_cookie();
  return _tmainCRTStartup();
}
```

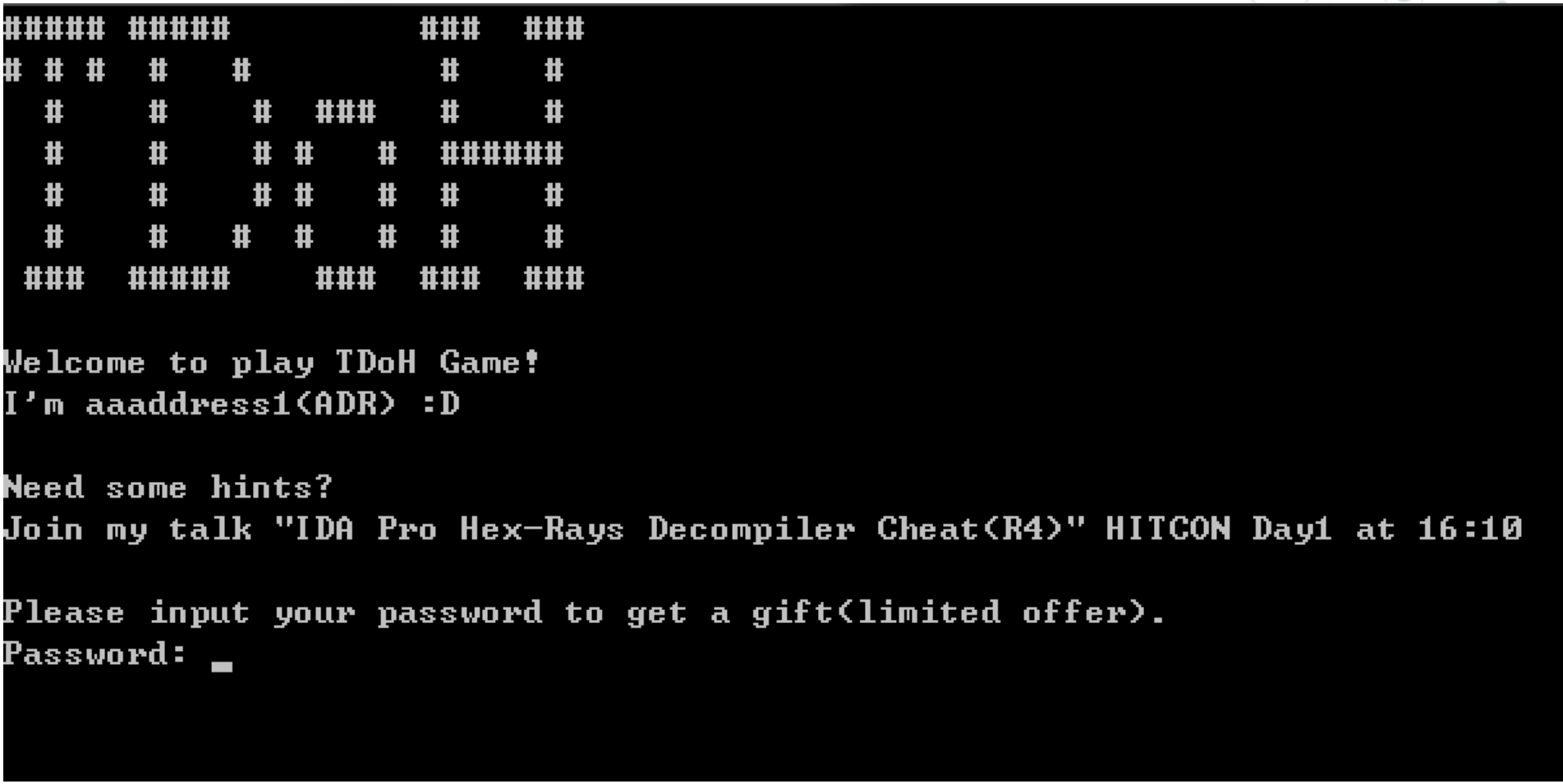


The Declaration of hacker - TDOH

8月26日 20:11 ·

哎呀。
在 Hacks In Taiwan 台灣駭客年會 (HITCON) Community社群場，TDoH的攤位將會有一些神祕的小物。
只要通過當天在攤位上的小活動就可以免費獲得喔！
如果您要贊助我們TDoH也可以直接購買。
敬請期待！！

讚　回應　分享


###
# # # # # # # #
  # # # ### # #
  # # # # # ######
  # # # # # # #
  # # # # # # #
 ### ##### ### ### ###
Welcome to play TDoH Game!
I'm aaaddress1(ADR) :D
Need some hints?
Join my talk "IDA Pro Hex-Rays Decompiler Cheat(R4)" HITCON Day1 at 16:10
Please input your password to get a gift(limited offer).
Password:

莫風徵伴侶！

如果各位單身飢渴已久的男、女性對莫風也有興趣，
歡迎到以下網址：

https://ioan.isalways.one


九阴真经
我这儿有本秘笈

Hello!

aaaddress1(ADR)

Q&A

aaaddress1@gmail.com