HITCON GIRLS 的第一個軟體破解課程

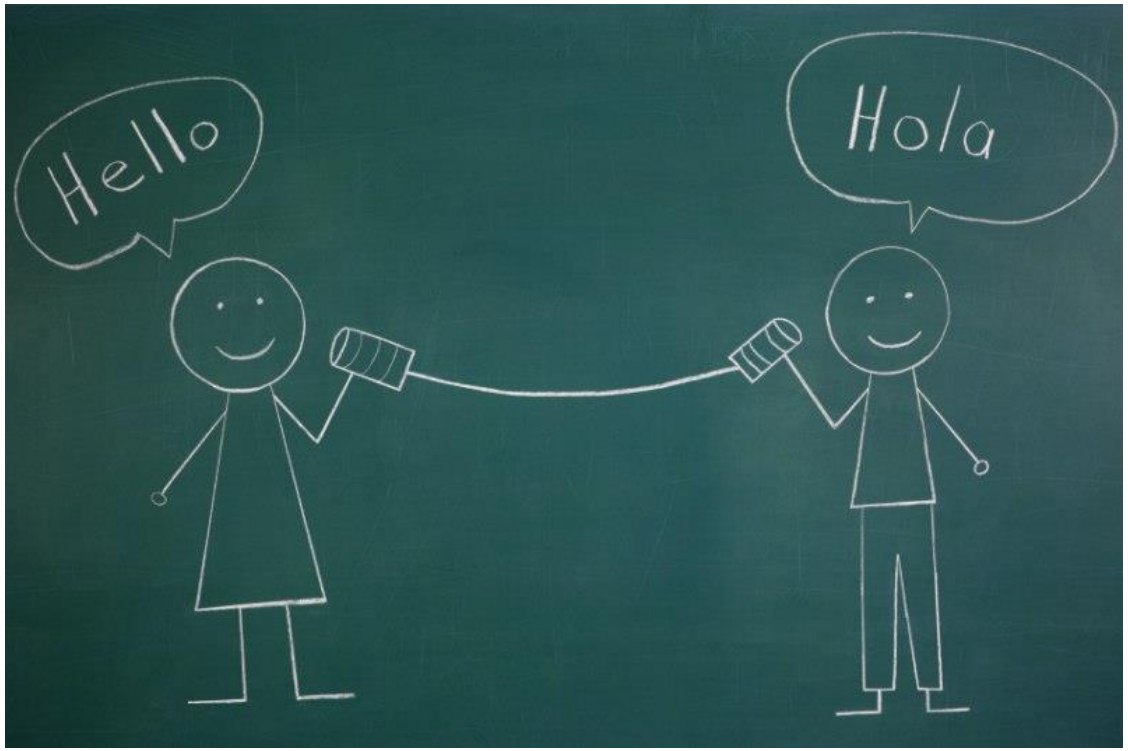
Charles

課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

逆向工程與軟體破解

- 何謂逆向工程 – 從宅仔和小P的愛情故事說起



逆向工程與軟體破解

- 何謂逆向工程 – 宅仔深愛著小 p，卻發現自己聽不懂她說的話

!@#$%^&*
()_+

```
int main(){
    printf("I love you\n!!");
}
```

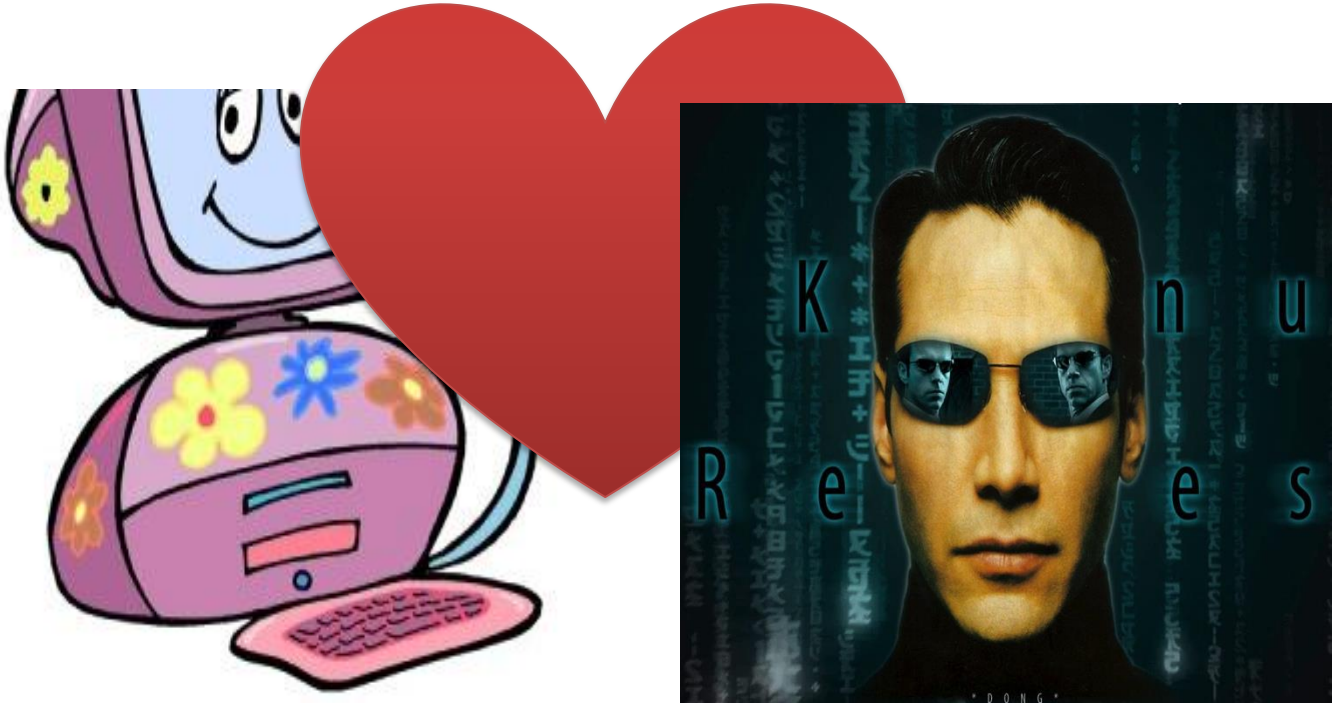
逆向工程與軟體破解

- 何謂逆向工程 – 痴心的宅仔想更了解小p的內心世界，於是努力學習小p的語言 – 機器語言。

```
817C24 08 1101>CMP DWORD PTR SS:[ESP+8],111
75 74          JNZ SHORT ncrackme.004010CE
8B4424 0C      MOV EAX,DWORD PTR SS:[ESP+C]
66:3D EA03     CMP AX,3EA
75 42          JNZ SHORT ncrackme.004010A6
E8 C7010
85C0
```

逆向工程與軟體破解

- 何謂逆向工程 –宅仔成功了，他學會了逆向工程，成為了**駭客**
- 成功擄獲小p的心。

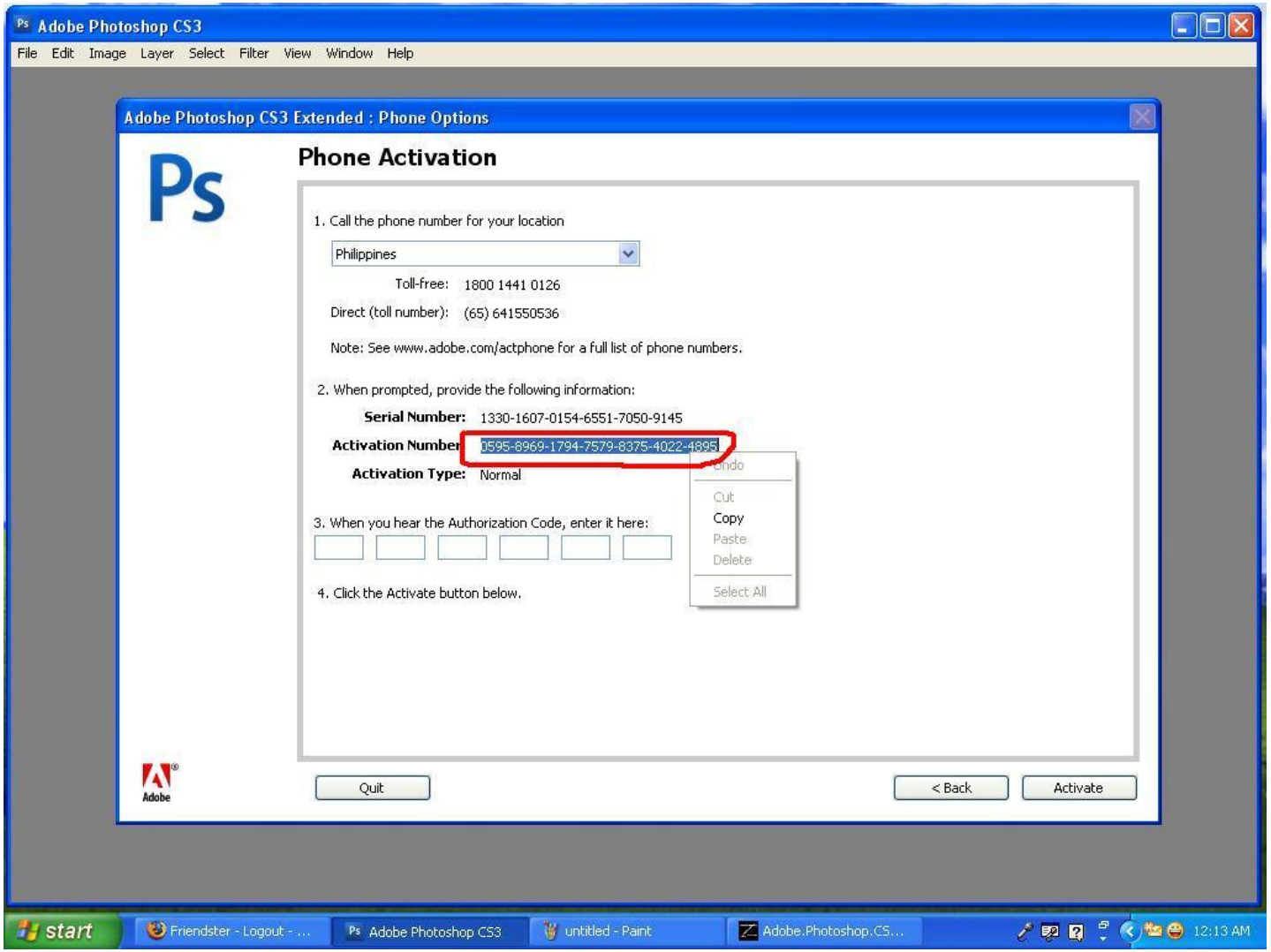


逆向工程與軟體破解

- 逆向工程：工程師在寫程式時，
  - 通常是以高階語言(high level language)作撰寫，
  - 再藉由Compiler的幫助將其轉化成電腦所能接受的機器語言(Machine language)。
  - 逆向工程是將一個編譯過的二進位程式檔，試圖重製，或了解該程式運作方式的過程。
- 軟體破解：藉由逆向工程的手法，了解程式的架構或流程後，嘗試對其進行修改，以改變其原始運作方式

  →(讓它乖乖聽話)。

軟體破解實例 – 序號破解


Adobe Photoshop CS3
File Edit Image Layer Select Filter View Window Help
Adobe Photoshop CS3 Extended : Phone Options
Phone Activation
1. Call the phone number for your location
Philippines
Toll-free: 1800 1441 0126
Direct (toll number): (65) 641550536
Note: See www.adobe.com/actphone for a full list of phone numbers.
2. When prompted, provide the following information:
Serial Number: 1330-1607-0154-6551-7050-9145
Activation Number: 0595-8969-1794-7579-8375-4022-4895
Activation Type: Normal
Undo
Cut
Copy
Paste
Delete
Select All
3. When you hear the Authorization Code, enter it here:
4. Click the Activate button below.
Quit
< Back
Activate
start
Friendster - Logout - ...
Adobe Photoshop CS3
untitled - Paint
Adobe.Photoshop.CS...
12:13 AM

逆向工程與軟體破解

- 為什麼我們要學逆向工程：
  - 破解軟體保護 (跟朋友炫耀，~~用免錢的程式~~)
  - 讓沒有原始碼的軟體程式可以二次利用
  - 評估軟體程式品質和健全性 (黑箱測試)
  - 研究病毒和惡意程式 (like Charles ☺)
  - 幫現有程式增加功能 (寫遊戲外掛)

逆向工程與軟體破解

- 作逆向工程前，你必須：
  - 寫過簡單程式，了解何謂程式架構、邏輯判斷及流程跳轉
  - 碰過一點組合語言
  - 具備耐心與熱忱

逆向工程與軟體破解

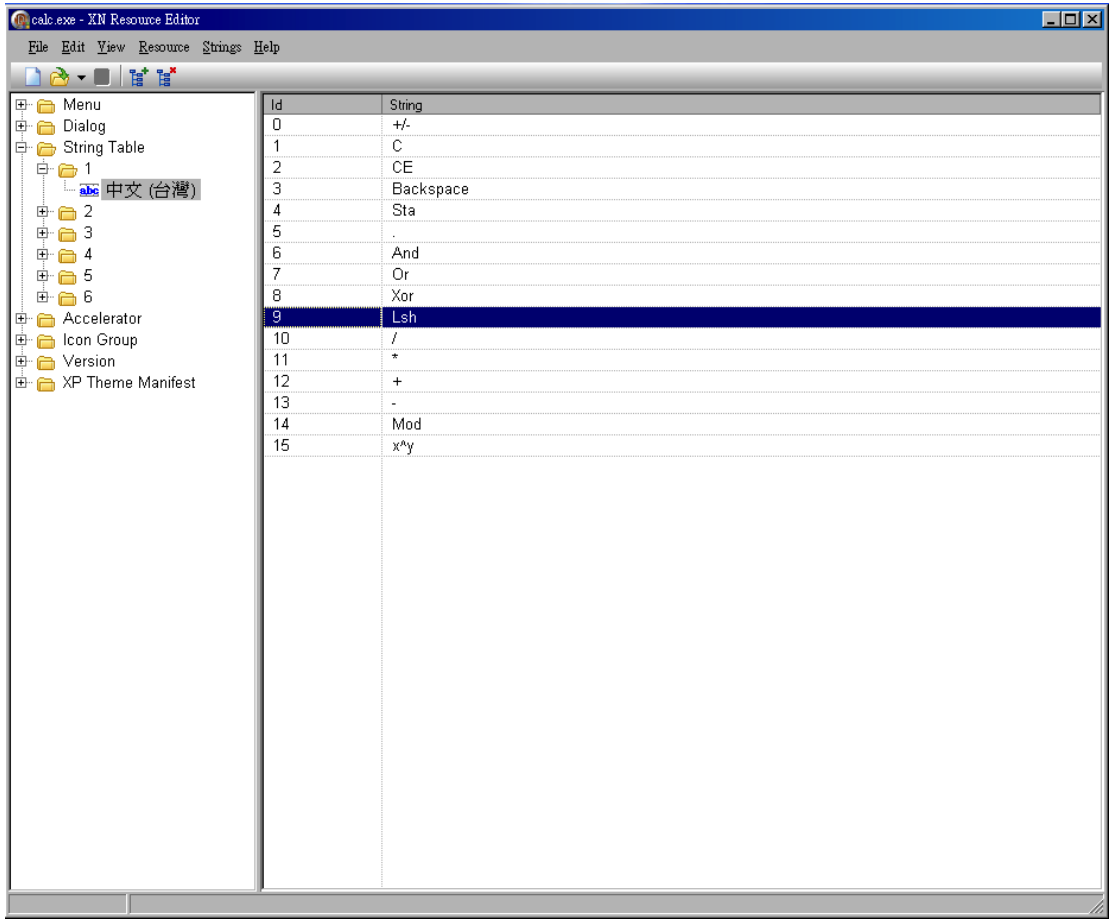
- 逆向工程常用到的工具：
  - 除錯器(Debuggers)
  - 16進位編輯器(Hex editors)
  - PE and resource viewers/editors
  - System Monitoring tools

課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

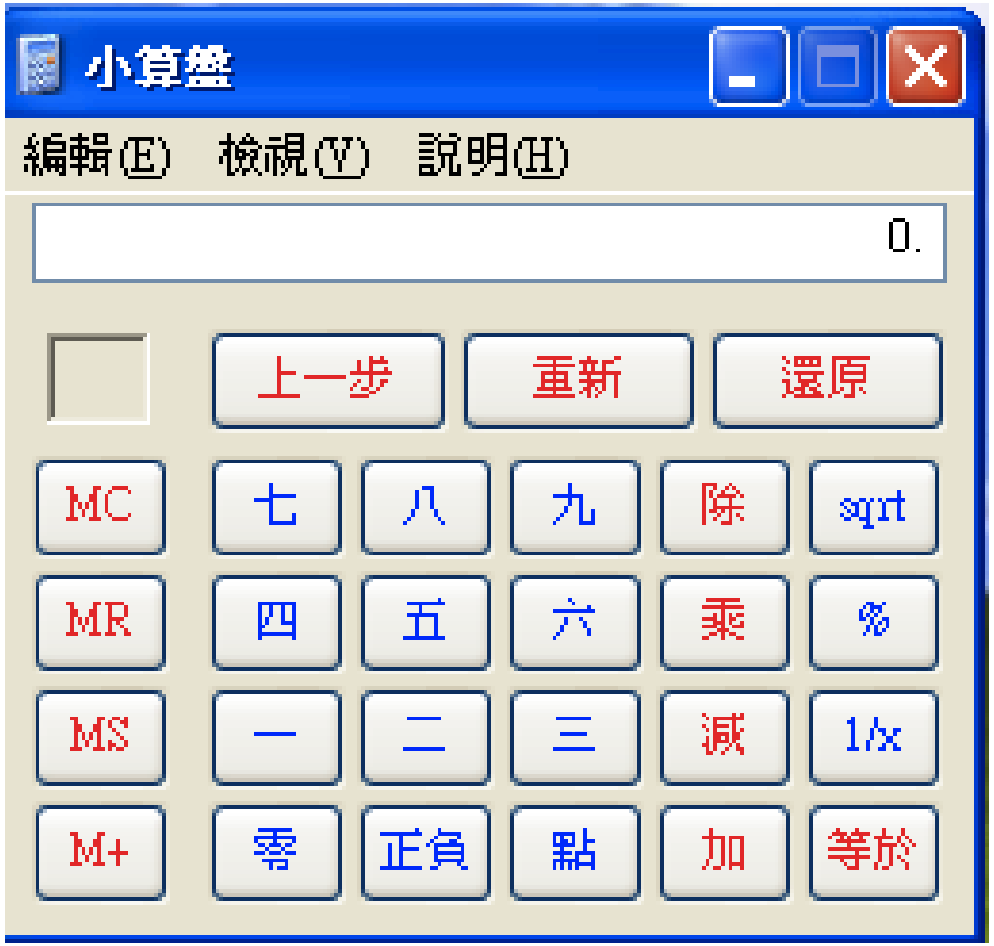
Lab 0

- 用資源編輯器來作我們的第一個程式修改：



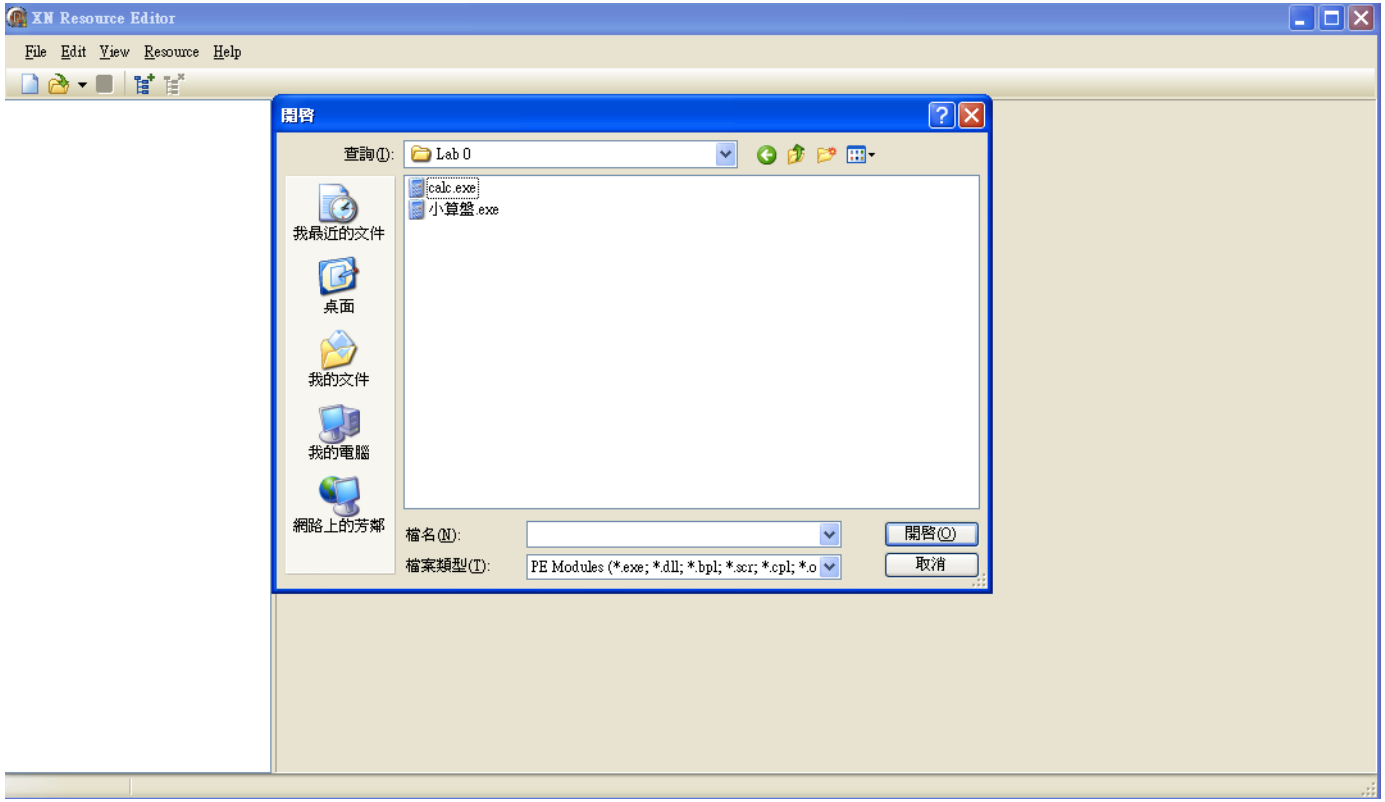
Lab 0

- 目的：修改出中文的小算盤：



Lab 0

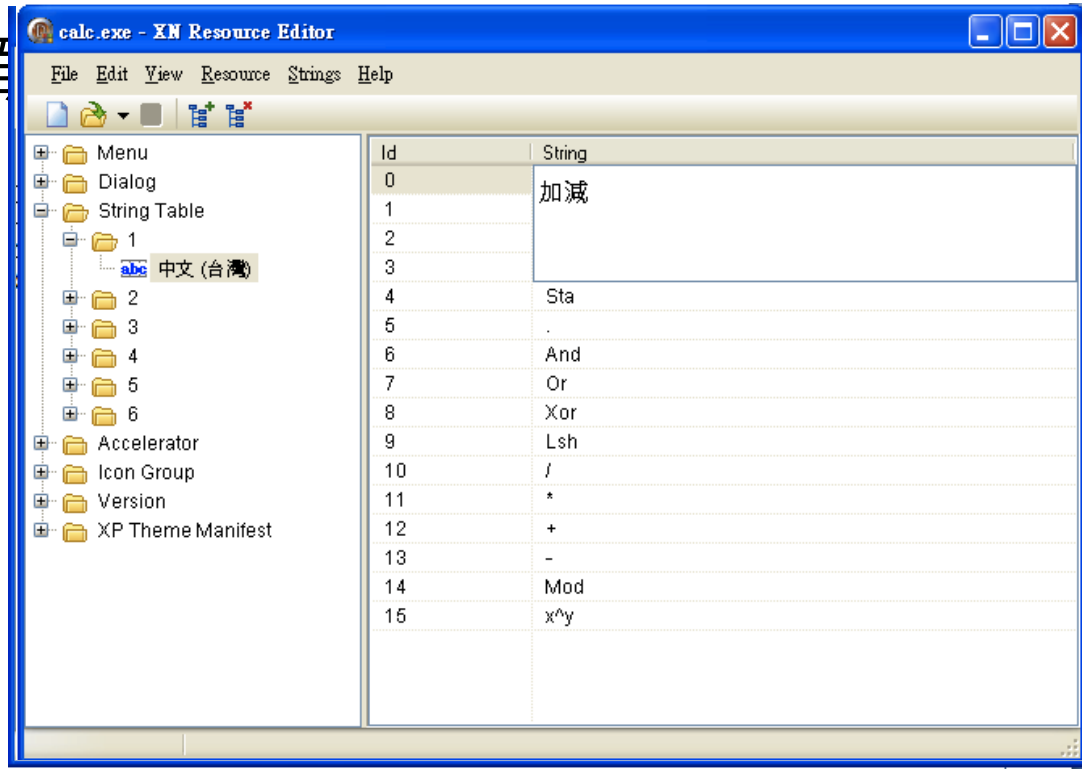
- 用XN Resource Editor打開”Lab 0”裡面附的calc.exe [File] → [Open]



Lab 0

- 選擇左方欄String Table, 依次展開，尋找想要修改的項目

- 找到想要修改的項
鍵修改



Lab 0

- 動手時間：10 mins

課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

程式破解前你必須知道的二三事

- 程式流程跳轉: if-else

```
int main(){

    unsigned int age;
    printf("Please enter your age: ");
    scanf("%d", &age);
    printf("\n");

    if(age < 18){
        printf
    }
    else{
        printf
    }
}
```

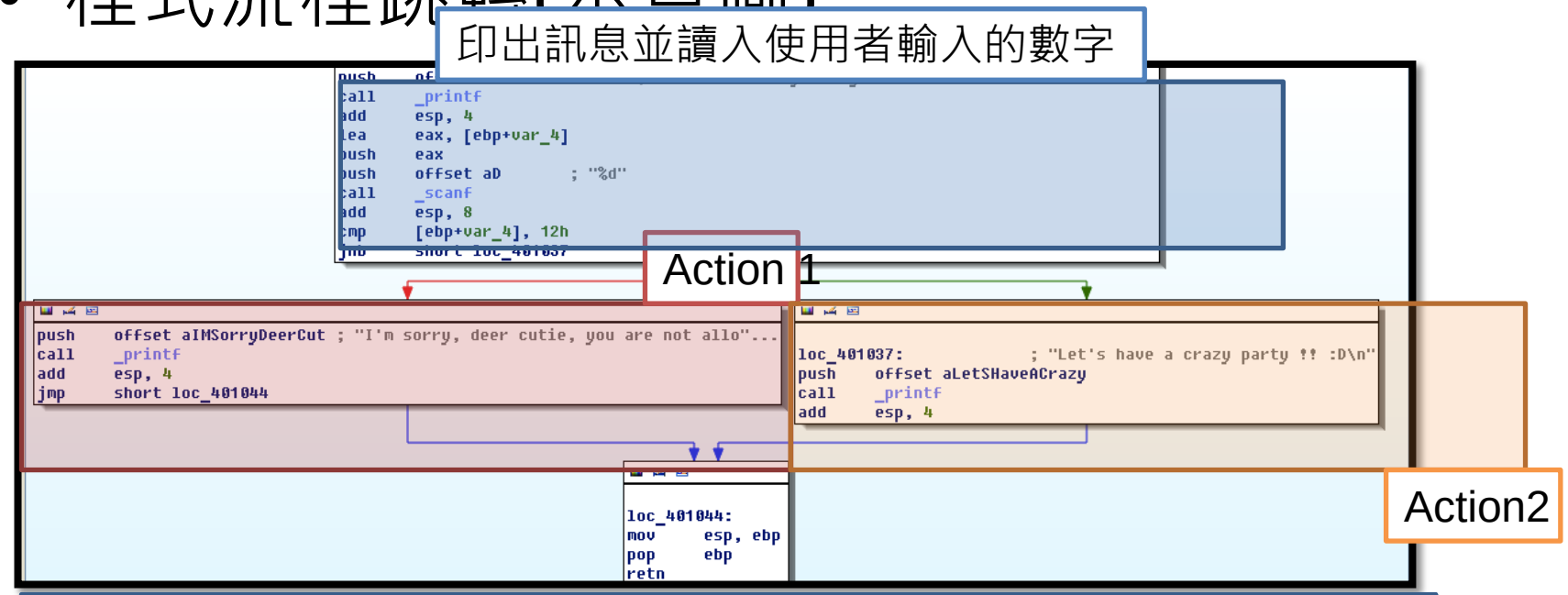
Action1 **程式操作**

```
C:\Documents and Settings\Administrator\桌面
\Hitcon_girls\Lab>age.exe
Please enter your age: 18
Let's have a crazy party !! :D

C:\Documents and Settings\Administrator\桌面
\Hitcon_girls\Lab>age.exe
Please enter your age: 12
I'm sorry, deer cutie, you are not allowed to drink beer :(
```

程式破解前你必須知道的二三事

- 程式流程跳轉(示意圖)



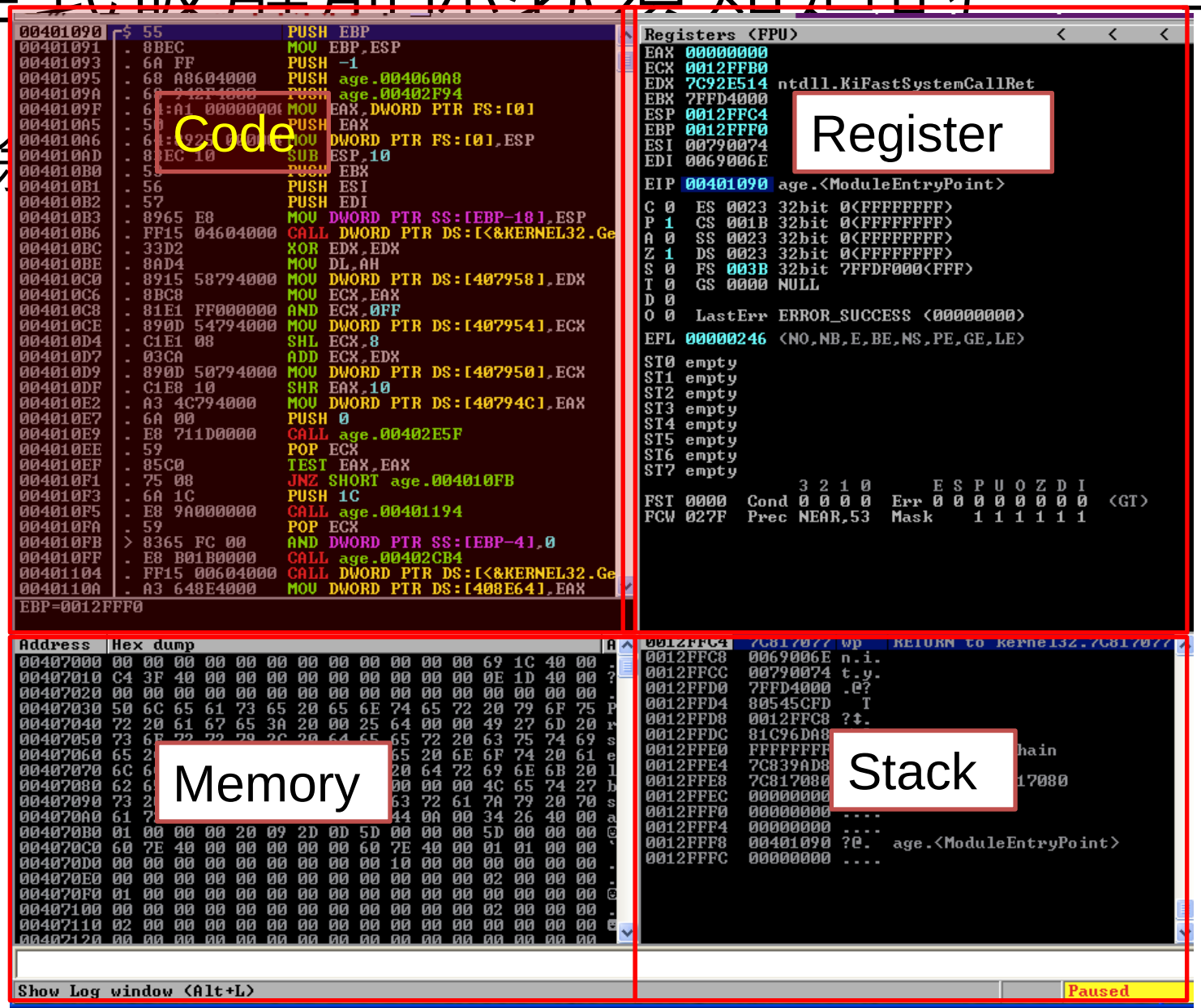

```
printf("Please enter your age: ");
scanf("%d", &age);
if(age < 18){
    printf("I'm sorry, deer cutie, you are not allowed to drink beer :(\n");}
else{
    printf("Let's have a crazy party !! :D\n");}
```

程式破解前你必須知道的二三事

- 除錯器：藉由除錯器(debugger)的幫助，可以一步步觀察目標程式在各時間點的狀態
  - 記憶體(memory)狀態、
  - 程式碼跳轉(code branch)，
- 有效幫助我們了解目標程式的運作邏輯。

程式破解前你必須知道的二三事

- 除



程式破解前你必須知道的二三事

- 除錯器 – Debugger 指令：
- 中斷點：cpu的特殊instruction，會使程式流程在該處中斷
  - bp xxxx: 在位址xxxx設置軟體斷點
  - F9: 執行程式，直到遇到斷點為止
  - F8: 執行單一指令
- **<u>Debugger操作</u>**

程式破解前你必須知道的二三事

- 我們的程式經編譯過後長什麼樣子呢？

記憶體位址

組合語言

```
00401000  -$ 55          PUSH EBP
00401001  . 8BEC         MOV EBP,ESP
00401003  . 51           PUSH ECX
00401004  . 68 30704     PUSH age.00407030          ASCII "Please enter your age: "
00401009  . E8 51000     CALL age.0040105F
0040100E  . 83C4 04      ADD ESP,4
00401011  . 8D45 FC      LEA EAX,DWORD PTR SS:[EBP-4
00401014  . 50           PUSH EAX
00401015  . 68 48704     PUSH age.00407048          ASCII "%d"
0040101A  . E8 29000     CALL age.00401048
0040101F  . 83C4 08      ADD ESP,8
00401022  . 837D FC      CMP DWORD PTR SS:[EBP-4],12
00401026  . 73 0F        JNB SHORT age.00401037
00401028  . 68 4C704     PUSH age.0040704C          ASCII "I'm sorry, deer cutie, you are not
0040102D  . E8 2D000     CALL age.0040105F
00401032  . 83C4 04      ADD ESP,4
00401035  . EB 0D        JMP SHORT age.00401044
00401037  > 68 8C704     PUSH age.0040708C          ASCII "Let's have a crazy party !! :D◙"
0040103C  . E8 1E000     CALL age.0040105F
00401041  . 83C4 04      ADD ESP,4
00401044  > 8BE5         MOV ESP,EBP
00401046  . 5D           POP EBP
00401047  -. C3          RETN
```

Action 1

Action 2

機器碼

程式破解前你必須知道的二三事

- 機器語言如何實作「流程分支」的概念：

```
00401000  $ 55          PUSH EBP
00401001  . 8BEC        MOV EBP,ESP
00401003  . 51          PUSH ECX
00401004  . 68 30704    PUSH age.00407030          ASCII "Please enter your age: "
00401
00401
00401                   PUSH EAX
                48704   PUSH age.00407048          ASCII "%d"
                        CALL age.00401048
0040101F  . 83C4 08     ADD ESP,8
00401022  . 837D FC     CMP DWORD PTR SS:[EBP-4],12
00401026  . 73 0F       JNB SHORT age.00401037
00401028  . 68 4C704    PUSH age.0040704C          ASCII "I'm sorry, deer cutie, you are not
0040102D  . E8 2D000    CALL age.0040105F
00401032  . 83C4 04     ADD ESP,4
00401035  . EB 0D       JMP SHORT age.00401044
00401037  > 68 8C704    PUSH age.0040708C          ASCII "Let's have a crazy party !! :D"
0040103C  . E8 1E000    CALL age.0040105F
00401041  . 83C4 04     ADD ESP,4
00401044  > 8BE5        MOV ESP,EBP
00401046  . 5D          POP EBP
00401047  . C3          RETN
```

比較變數跟0x12(18)的大小值

JNB: Jmp if Not Below

cmp: compare

Action 1

Action 2

如果條件(不小於)成立，程式跳到0x401037(Action 2)去執行

課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

Lab 1

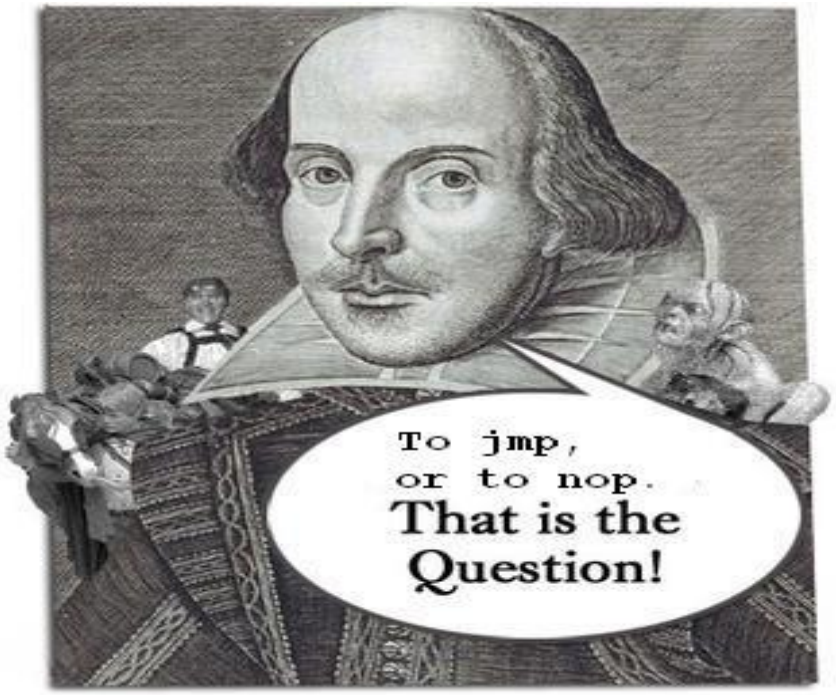
- 目標：修改我們的程式age.exe，讓它無論輸入年紀多少都顯示: “Let's have a crazy party !! :D”

```
C:\Documents and Settings\Administrator\桌面\Hitcon_girls\Lab 1>noage.exe
Please enter your age: 10
Let's have a crazy party !! :D

C:\Documents and Settings\Administrator\桌面\Hitcon_girls\Lab 1>noage.exe
Please enter your age: 99
Let's have a crazy party !! :D
```

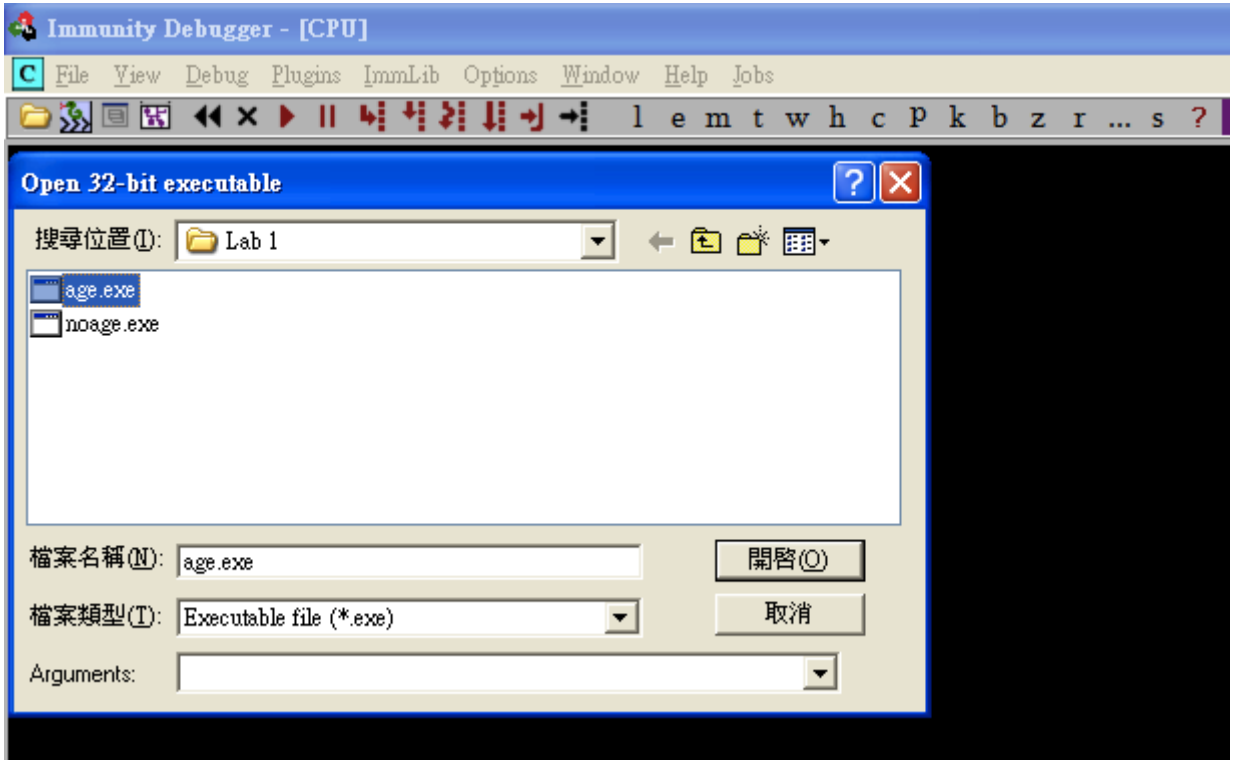
Lab 1

- 兩個必須熟知的組合語言指令:
  - nop: No Operation, 不作任何事
  - jmp xxxx: 將程式流程強制跳至位址xxxx去執行



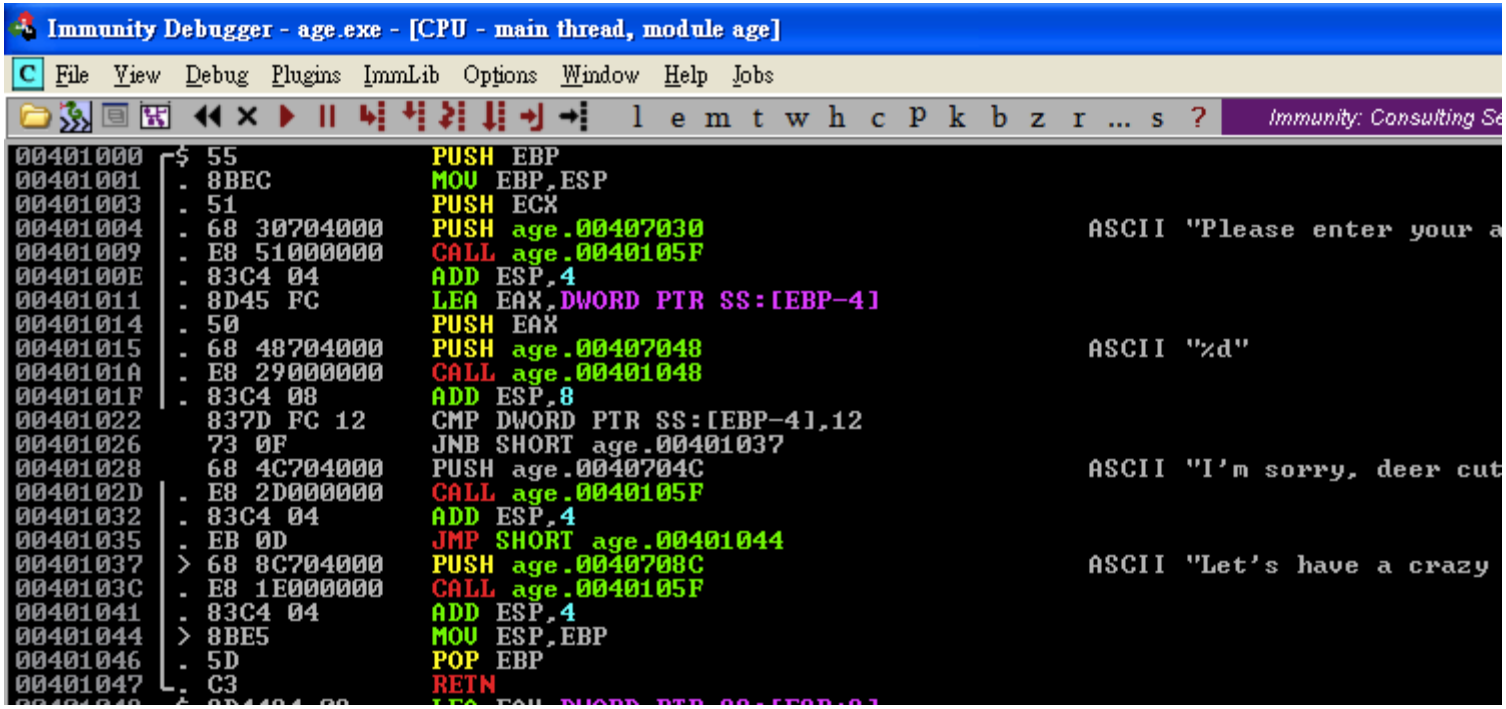
Lab 1

- 用debugger開啟我們的 age.exe:
- [File] → [Open], 選擇age.exe



Lab 1

- 在code視窗(左上角)，作上下捲動，直到找到0x401000 – 我們的main function 所在為止(或按Ctrl+G，輸入0x401000直接到該位址)



Lab 1

- To jmp or to nop, that’s the question:
  - 在紅色箭頭所在處，程式有一個分歧點：
    - 繼續往下一個指令執行(不jmp)
    - 亦或跳到0x401037執行

```
00401000 $ 55          PUSH EBP
00401001 . 8BEC        MOV EBP,ESP
00401003 . 51          PUSH ECX
00401004 . 68 30704    PUSH age.00407030          ASCII "Please enter your age: "
00401009 . E8 51000    CALL age.0040105F
0040100E . 83C4 04     ADD ESP,4
00401011 . 8D45 FC     LEA EAX,DWORD PTR SS:[EBP-4
00401014 . 50          PUSH EAX
00401015 . 68 48704    PUSH age.00407048
0040101A . E8 29000    CALL age.00401048
0040101F . 83C4 08     ADD ESP,8
00401022 . 837D FC     CMP DWORD PTR SS:[EBP-4],12
00401026 . 73 0F       JNB SHORT age.00401037
00401028 . 68 4C704    PUSH age.0040704C          ASCII "I’m sorry, deer cutie, you are not
0040102D . E8 2D000    CALL age.0040105F
00401032 . 83C4 04     ADD ESP,4
00401035 . EB 0D       JMP SHORT age.00401044
00401037 > 68 8C70     PUSH age.0040708C          ASCII "Let’s have a crazy party !! :D"
0040103C . E8 1E000    CALL age.0040105F
00401041 . 83C4 04     ADD ESP,4
00401044 > 8BE5        MOV ESP,EBP
00401046 . 5D          POP EBP
00401047 . C3          RETN
```

比較式，0x12=18

Lab 1

- 假設你希望把分歧點消除，讓程式永遠只作某件事：
  - 把jnb指令改成nop→永遠作action 1
  - 把jnb指令改成jmp→永遠作action 2

```
00401000 $ 55          PUSH EBP
00401001 . 8BEC        MOV EBP,ESP
00401003 . 51          PUSH ECX
00401004 . 68 30704    PUSH age.00407030          ASCII "Please enter your age: "
00401009 . E8 51000    CALL age.0040105F
0040100E . 83C4 04     ADD ESP,4
00401011 . 8D45 FC     LEA EAX,DWORD PTR SS:[EBP-4
00401014 . 50          PUSH EAX
00401015 . 68 48704    PUSH age.00407048          ASCII "%d"
0040101A . E8 29000    CALL age.00401048
0040101F . 83C4 08     ADD ESP,8
00401022 . 837D FC     CMP DWORD PTR SS:[EBP-4],12
00401026 . 73 0F       JMP SHORT age.00401037
00401028 . 68 4C704    PUSH age.0040704C          ASCII "I'm sorry, deer cutie, you are not
0040102D . E8 2D000    CALL age.0040105F
00401032 . 83C4 04     ADD ESP,4
00401035 . EB 0D       JMP SHORT age.00401044
00401037 > 68 8C704    PUSH age.0040708C          ASCII "Let's have a crazy party !! :D"
0040103C . E8 1E000    CALL age.0040105F
00401041 . 83C4 04     ADD ESP,4
00401044 > 8BE5        MOV ESP,EBP
00401046 . 5D          POP EBP
00401047 . C3          RETN
```

Action 1

Action 2

Lab 1

- 動手時間 – 10 mins – **怎麼修改指令？**
- 我們的目標是希望程式永遠執行action 2，因此要將jnb所在的指令改為jmp
  - 將滑鼠移至該指令上方選取它
  - 按右鍵
  - 選”Assemble”
  - 將原本的
    - ”JNB SHORT 00401037”，改成→
    - “Jmp SHORT 00401037”
  - 按F9執行程式，看結果

課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

通往破解點的道路

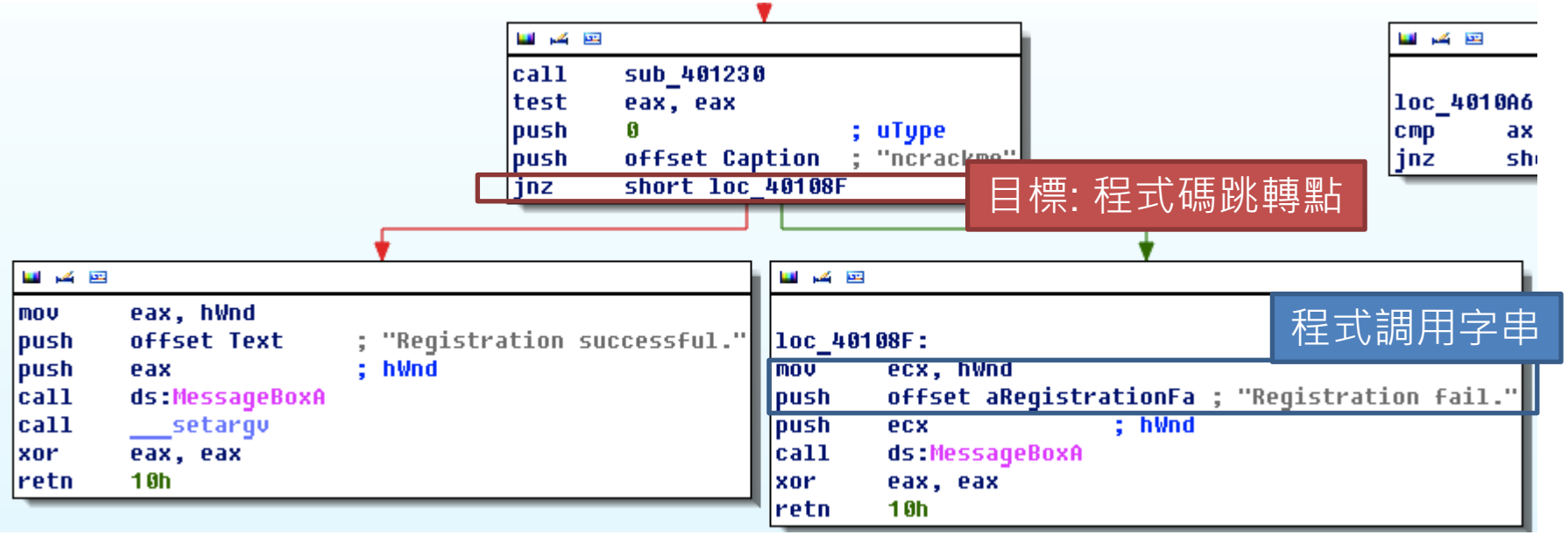
通往破解點的道路 – 字串引用

- 範例程式：簡單的序號驗證程式。
- **程式操作示範**
- 假想程式流程：
  - 讀進reg name 和 reg key
  - 將reg name, reg key當作參數，傳入自己的演算法，驗證是否正確
  - 驗證失敗，跳出視窗顯示錯誤

猜測：程式引用到”Registration fail”處的程式碼，可能很接近作驗證的程式碼(我們要修改的點)

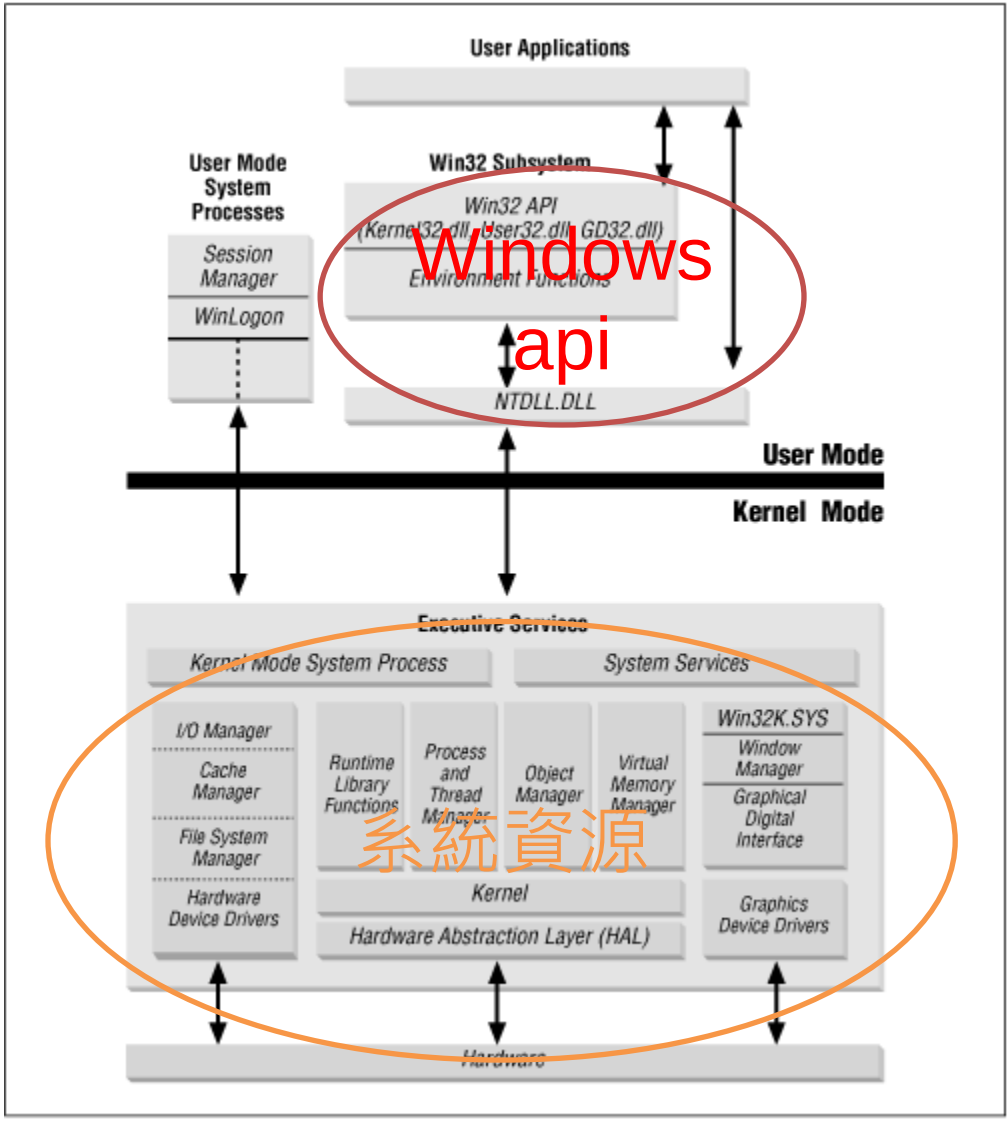
通往破解點的道路 – 字串引用

- 程式實際流程圖：



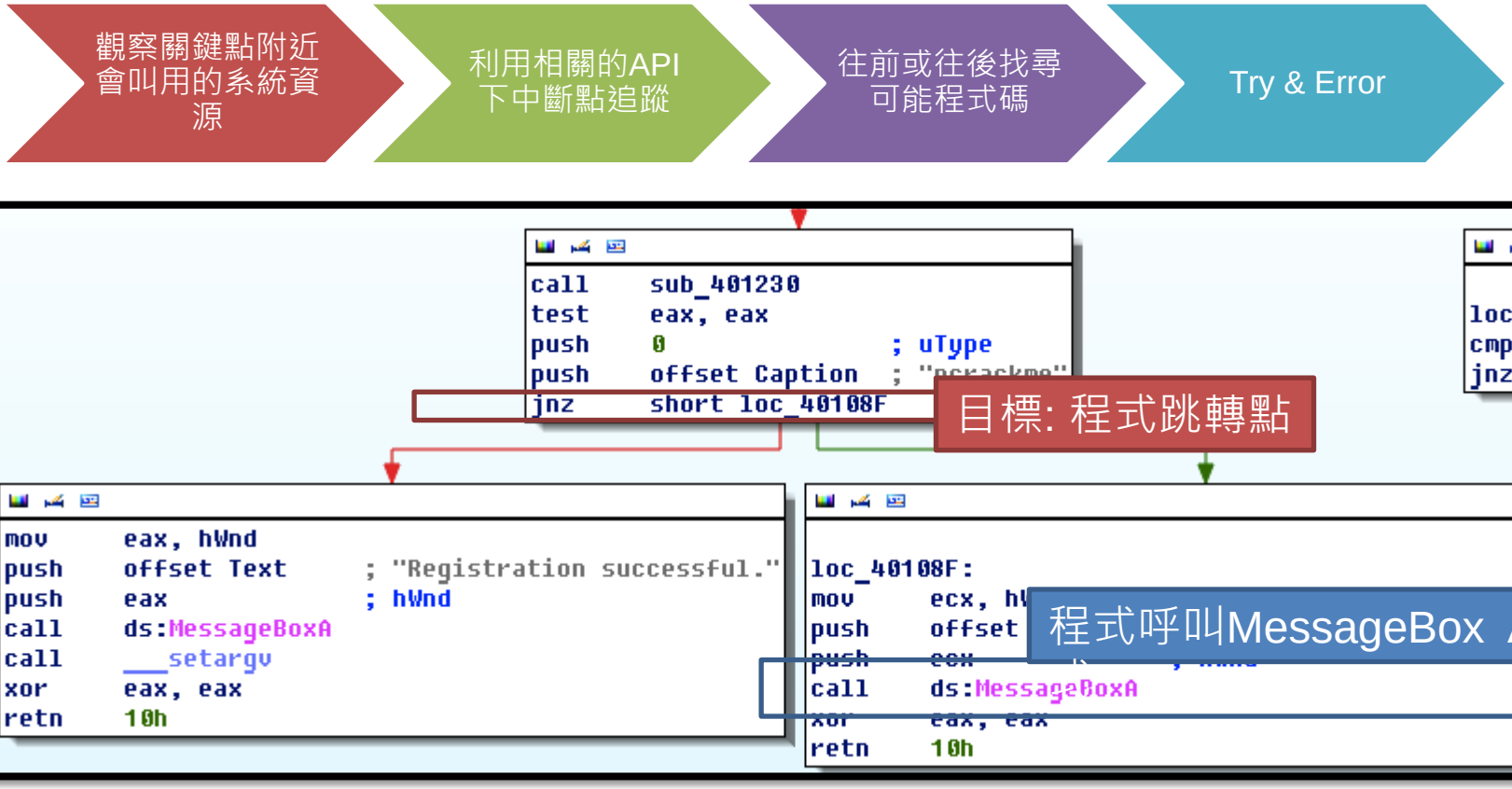
通往破解點的道路 - Windows API

- 視窗作業系統應用程式介面（Windows API）
  - Windows提供的函式
  - 用來讓使用者作底層的系統資源存取
  - 分門別類存放於不同的函式庫(DLL)中



通往破解點的道路 - Windows API

- 掌握適當Windows API, 可以幫助我們快速找到關鍵程式碼(破解點)。



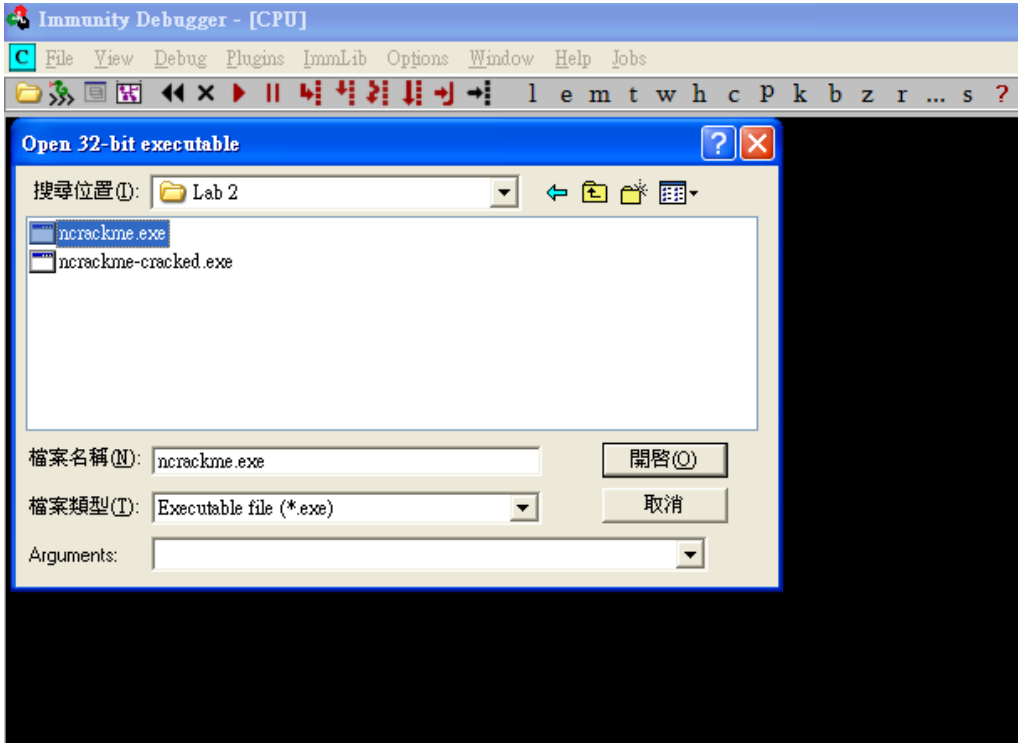
課程內容

- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

Lab 2

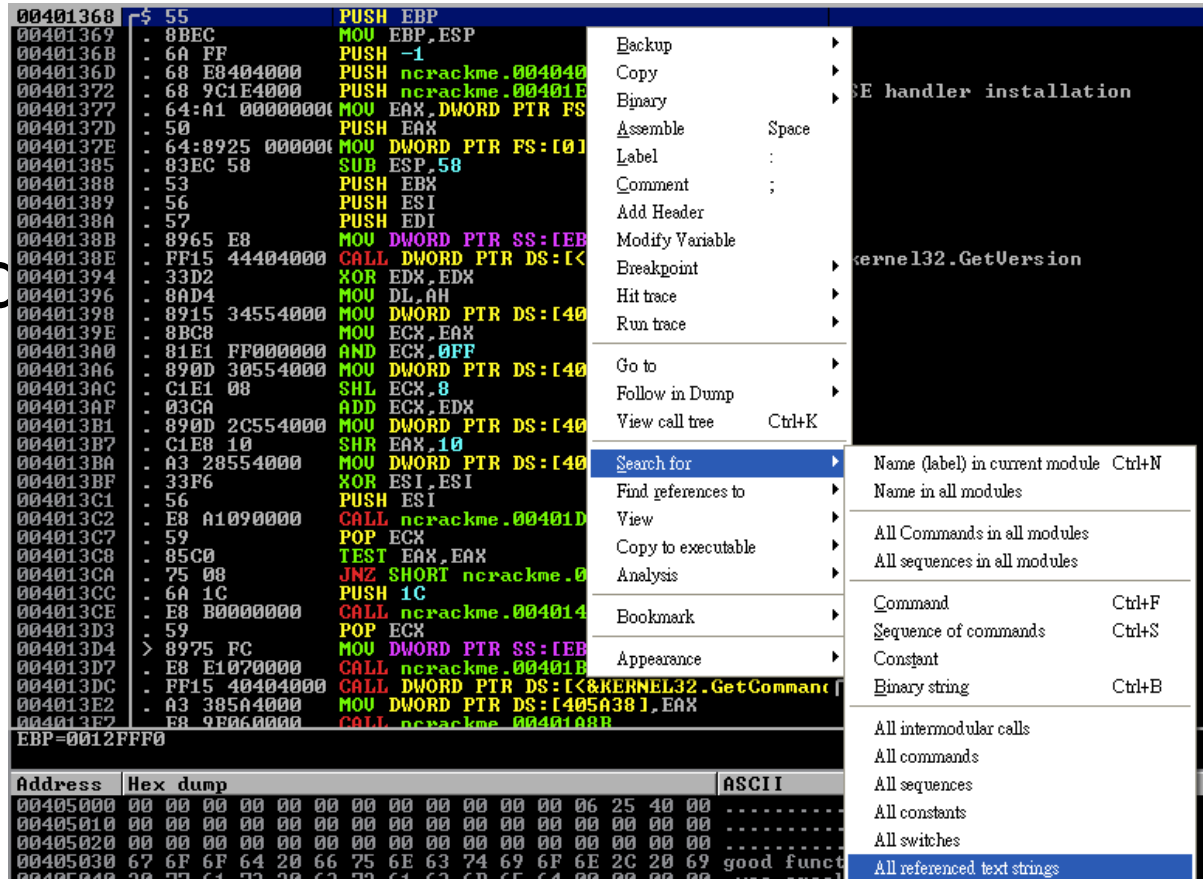
- 目標：用Debugger來追蹤ncrackme.exe，找出程式作序號檢查的程式點
- 用Debugger開啟ncrackme.exe:

  [File] → [Open]



Lab 2

- 方法一 – 利用字串引用來尋找：
- 按右鍵→
- “Search for”
- “All referenced text strings”



Lab 2

- 程式中會引用到的字串列表：

```
0040106D PUSH ncrackme.00405080                    ASCII "ncrackme"
00401079 PUSH ncrackme.00405064                    ASCII "Registration successful."
00401095 PUSH ncrackme.00405050                    ASCII "Registration fail."
004010BD PUSH ncrackme.00405080                    ASCII "ncrackme"
004010C2 PUSH ncrackme.00405030                    ASCII "good function, i was cracked"
00401140 MOV DWORD PTR SS:[ESP+58],ncrackme.0040   ASCII "myWindowClass"
00401176 PUSH ncrackme.0040508C                    ASCII "Newbie smallsize crackme - v1"
0040117B PUSH ncrackme.004040D8                    ASCII "myWindowClass"
00401368 PUSH EBP                                  (Initial CPU selection)
0040202A PUSH ncrackme.004043CC                    ASCII "<program name unknown>"
0040206C PUSH ncrackme.004043C8                    ASCII "..."
00402080 PUSH ncrackme.004043AC                    ASCII "Runtime Error!□□Program: "
0040209E PUSH ncrackme.004043A8                    ASCII "□□"
004020C6 PUSH ncrackme.00404380                    ASCII "Microsoft Visual C++ Runtime Lib
004032DB PUSH ncrackme.00404414                    ASCII "user32.dll"
004032F2 PUSH ncrackme.00404408                    ASCII "MessageBoxA"
00403303 PUSH ncrackme.004043F8                    ASCII "GetActiveWindow"
0040330B PUSH ncrackme.004043E4                    ASCII "GetLastActivePopup"
```

- 在想追蹤的string上按右鍵→”Follow in dissembler”

```
0040106D PUSH ncrackme.00405080                    ASCII "ncrackme"
00401079 PUSH ncrackme.00405064                    ASCII "Registration successful."
00401095 PUSH ncrackme.00405050                    ASCII "Registrati
004010BD PUSH ncrackme.00405080                    ASCII "ncrackme"
004010C2 PUSH ncrackme.00405030                    ASCII "good funct
00401140 MOV DWORD PTR SS:[ESP+58],ncrackme.0040   ASCII "myWindowCl
00401176 PUSH ncrackme.0040508C                    ASCII "Newbie sma
0040117B PUSH ncrackme.004040D8                    ASCII "myWindowCl
00401368 PUSH EBP                                  (Initial CPU sele
0040202A PUSH ncrackme.004043CC                    ASCII "<program r
0040206C PUSH ncrackme.004043C8                    ASCII "..."
00402080 PUSH ncrackme.004043AC                    ASCII "Runtime Er
0040209E PUSH ncrackme.004043A8                    ASCII "□□"
```

Follow in Disassembler	Enter
Search for text	
Toggle breakpoint	F2
Conditional breakpoint	Shift+F2
Conditional log breakpoint	Shift+F4
Set breakpoint on every command	

Lab 2

- 即可找到關鍵程式碼區塊：

```
00401050  . 817C24 08 1101 CMP DWORD PTR SS:[ESP+8],111
00401058  . 75 74          JNZ SHORT ncrackme.004010CE
0040105A  . 8B4424 0C      MOV EAX,DWORD PTR SS:[ESP+C]
0040105E  . 66:3D EA03     CMP AX,3EA
00401062  . 75 42          JNZ SHORT ncrackme.004010A6
00401064  . E8 C7010000    CALL ncrackme.00401230
00401069  . 85C0           TEST EAX,EAX
0040106B  . 6A 00          PUSH 0                          ┌Style = MB_OK|MB_APPLMODAL
0040106D  . 68 80504000    PUSH ncrackme.00405080          │Title = "ncrackme"
00401072  . 75 1B          JNZ SHORT ncrackme.0040108F     │
00401074  . A1 B8564000    MOV EAX,DWORD PTR DS:[4056B8]   │
00401079  . 68 64504000    PUSH ncrackme.00405064          │Text = "Registration successful."
0040107E  . 50             PUSH EAX                        │hOwner => NULL
0040107F  . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA└MessageBoxA
00401085  . E8 A6020000    CALL ncrackme.00401330
0040108A  . 33C0           XOR EAX,EAX
0040108C  . C2 1000        RETN 10
0040108F  > 8B0D B8564000  MOV ECX,DWORD PTR DS:[4056B8]   │
00401095  . 68 50504000    PUSH ncrackme.00405050          │Text = "Registration fail."
0040109A  . 51             PUSH ECX                        │hOwner => NULL
0040109B  . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA└MessageBoxA
004010A1  . 33C0           XOR EAX,EAX
004010A3  . C2 1000        RETN 10
004010A6  > 66:3D EB03     CMP AX,3EB
004010AA  . 75 22          JNZ SHORT ncrackme.004010CE
004010AC  . A1 C0564000    MOV EAX,DWORD PTR DS:[4056C0]
004010B1  . 85C0           TEST EAX,EAX
004010B3  . 74 19          JE SHORT ncrackme.004010CE
004010B5  . 8B15 B8564000  MOV EDX,DWORD PTR DS:[4056B8]
004010BB  . 6A 00          PUSH 0                          ┌Style = MB_OK|MB_APPLMODAL
004010BD  . 68 80504000    PUSH ncrackme.00405080          │Title = "ncrackme"
004010C2  . 68 30504000    PUSH ncrackme.00405030          │Text = "good function, i was cracked"
004010C7  . 52             PUSH EDX                        │hOwner => NULL
004010C8  . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA└MessageBoxA
004010CE  > 33C0           XOR EAX,EAX
004010D0  . C2 1000        RETN 10
004010D3    90             NOP
004010D4    90             NOP
004010D5    90             NOP
```

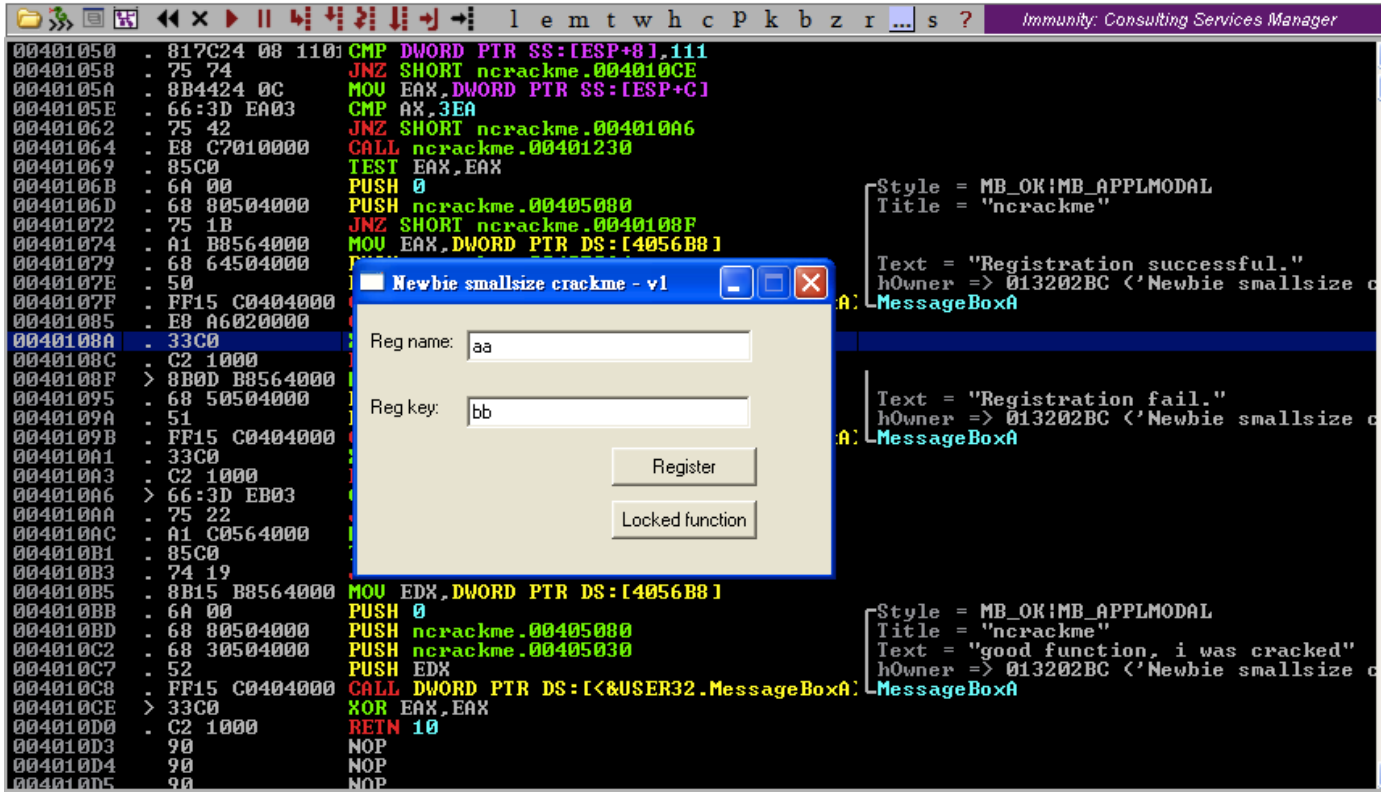
Lab 2

- 方法二— 利用API MessageBoxA來設置中斷點找尋：
- 按「Ctrl + N」，debugger會列出所有引用到的函式

0040402C	.rdata	Import	KERNEL32.HeapAlloc
00404080	.rdata	Import	KERNEL32.HeapCreate
0040407C	.rdata	Import	KERNEL32.HeapDestroy
00404088	.rdata	Import	KERNEL32.HeapFree
00404024	.rdata	Import	KERNEL32.HeapReAlloc
00404014	.rdata	Import	KERNEL32.LCMapStringA
00404010	.rdata	Import	KERNEL32.LCMapStringW
0040409C	.rdata	Import	USER32.LoadCursorA
00404098	.rdata	Import	USER32.LoadIconA
0040401C	.rdata	Import	KERNEL32.LoadLibraryA
004040C0	.rdata	Import	USER32.MessageBoxA
00401368	.text	Export	<ModuleEntryPoint>
00404018	.rdata	Import	KERNEL32.MultiByteToWideChar
004040CC	.rdata	Import	USER32.PostQuitMessage
004040A0	.rdata	Import	USER32.RegisterClassExA
0040408C	.rdata	Import	KERNEL32.RtlUnwind
00404070	.rdata	Import	KERNEL32.SetHandleCount
004040AC	.rdata	Import	USER32.ShowWindow
0040404C	.rdata	Import	KERNEL32.TerminateProcess
004040B8	.rdata	Import	USER32.TranslateMessage
00404054	.rdata	Import	KERNEL32.UnhandledExceptionFilter
004040B0	.rdata	Import	USER32.UpdateWindow
00404028	.rdata	Import	KERNEL32.VirtualAlloc
00404084	.rdata	Import	KERNEL32.VirtualFree
00404000	.rdata	Import	KERNEL32.WideCharToMultiByte
00404060	.rdata	Import	KERNEL32.WriteFile

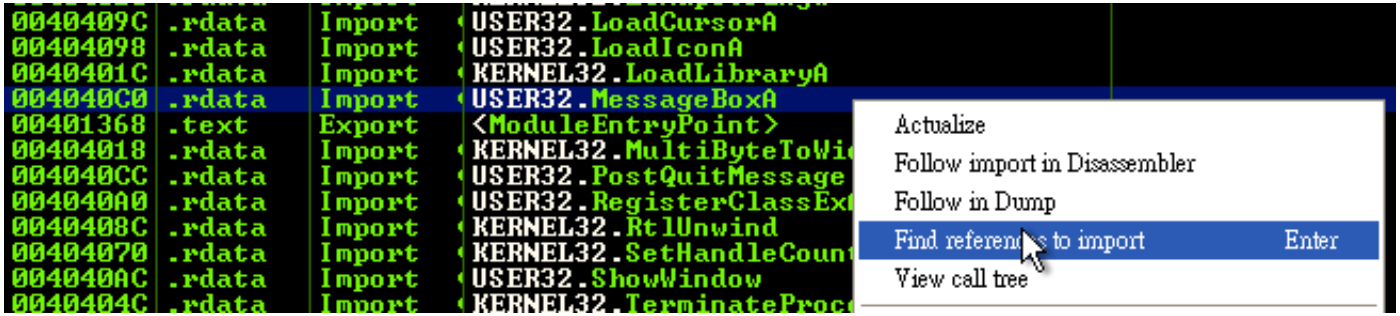
Lab 2

- 按”F9”讓程式執行，跳出視窗後在”Reg name”, “Reg key”欄位任意輸入東西，按”Register”

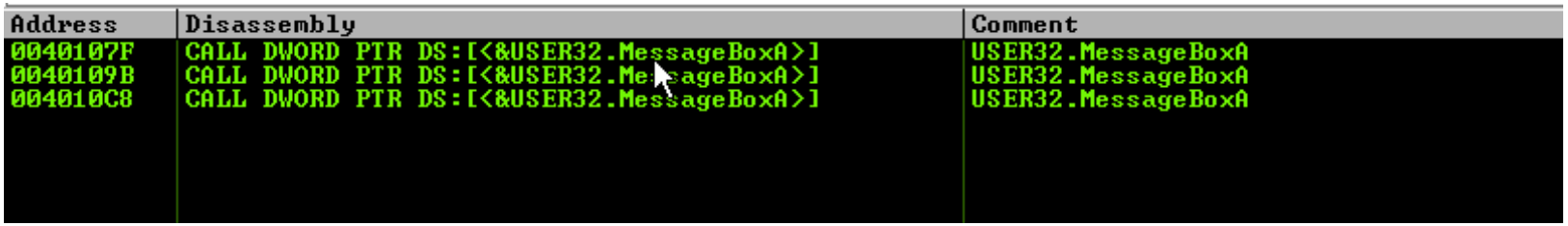


Lab 2

- 在API函式上按右鍵，選擇”Find reference to import”



- 除錯器會列出程式中所有引用此API的位址，在想追蹤的位址上按”enter”



Lab 2

- 即可找到呼叫”MessageBoxA”的程式碼：

程式跳轉點

註冊成功的情況

註冊失敗的情況

```
00401058   . 75 74          JNZ SHORT ncrackme.004010CE
0040105A   . 8B4424 0C      MOV EAX,DWORD PTR SS:[ESP+C]
0040105E   . 66:3D EA03     CMP AX,3EA
00401062   . 75 42          JNZ SHORT ncrackme.004010A6
00401064   . E8 C7010000    CALL ncrackme.00401230
00401069   . 85C0           TEST EAX,EAX
           . 6A 00          PUSH 0                          ┌Style = MB_OK|MB_APPLMODAL
           . 68 80504000    PUSH ncrackme.00405080          │Title = "ncrackme"
           . 75 1B          JNZ SHORT ncrackme.0040108F
           . A1 B8564000    MOV EAX,DWORD PTR DS:[4056B8]
           . 68 64504000    PUSH ncrackme.00405064          │Text = "Registration successful."
           .                PUSH EAX                        │hOwner => 013202BC ('Newbie smallsize crackme - v1',
           . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA] └MessageBoxA
           . E8 A6020000    CALL ncrackme.00401330
0040108A   . 33C0           XOR EAX,EAX
0040108C   . C2 1000        RETN 10
0040108F   > 8B0D B8564000  MOV ECX,DWORD PTR DS:[4056B8]
           . 68 50504000    PUSH ncrackme.00405050          │Text = "Registration fail."
           .                PUSH ECX                        │hOwner => 013202BC ('Newbie smallsize crackme - v1',
           . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA] └MessageBoxA
           . 33C0           XOR EAX,EAX
004010A3   . C2 1000        RETN 10
004010A6   > 66:3D EB03     CMP AX,3EB
004010AA   . 75 22          JNZ SHORT ncrackme.004010CE
004010AC   . A1 C0564000    MOV EAX,DWORD PTR DS:[4056C0]
004010B1   . 85C0           TEST EAX,EAX
004010B3   . 74 19          JE SHORT ncrackme.004010CE
004010B5   . 8B15 B8564000  MOV EDX,DWORD PTR DS:[4056B8]
004010BB   . 6A 00          PUSH 0                          ┌Style = MB_OK|MB_APPLMODAL
004010BD   . 68 80504000    PUSH ncrackme.00405080          │Title = "ncrackme"
004010C2   . 68 30504000    PUSH ncrackme.00405030          │Text = "good function, i was cracked"
004010C7   . 52             PUSH EDX                        │hOwner => 013202BC ('Newbie smallsize crackme - v1',
004010C8   . FF15 C0404000  CALL DWORD PTR DS:[<&USER32.MessageBoxA] └MessageBoxA
004010CE   > 33C0           XOR EAX,EAX
004010D0   . C2 1000        RETN 10
004010D3     90             NOP
004010D4     90             NOP
004010D5     90             NOP
004010D6     90             NOP
```

現在，試著修改程式，讓我們不管輸入什麼序號，永遠都會顯示註冊成功 ☺

Lab 2

- 動手時間 – 10 mins
- 將位址0x401072的指令由原本的
  - JNZ SHORT 0040108F，改為
  - NOP
- 按F9執行，觀察結果

課程內容

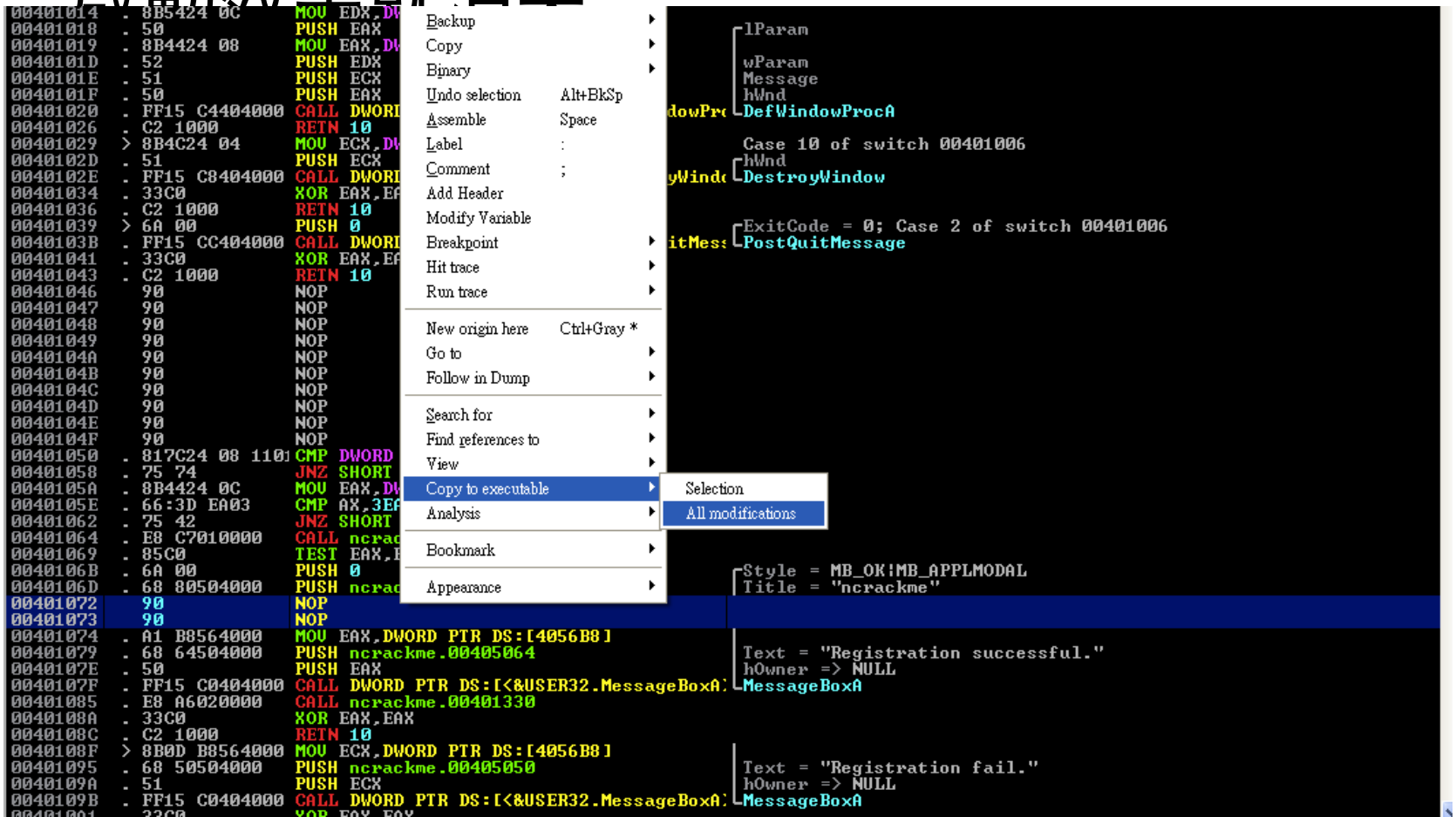
- 逆向工程與軟體破解概念簡介
- Lab 0
- 軟體破解前你必須知道的二三事
- Lab 1
- 通往破解點的道路 – Strings 與 API
- Lab 2
- 軟體破解實戰Demo – 010 Editor

demo

- 010序號驗證機制破解 (僅供教學目的)

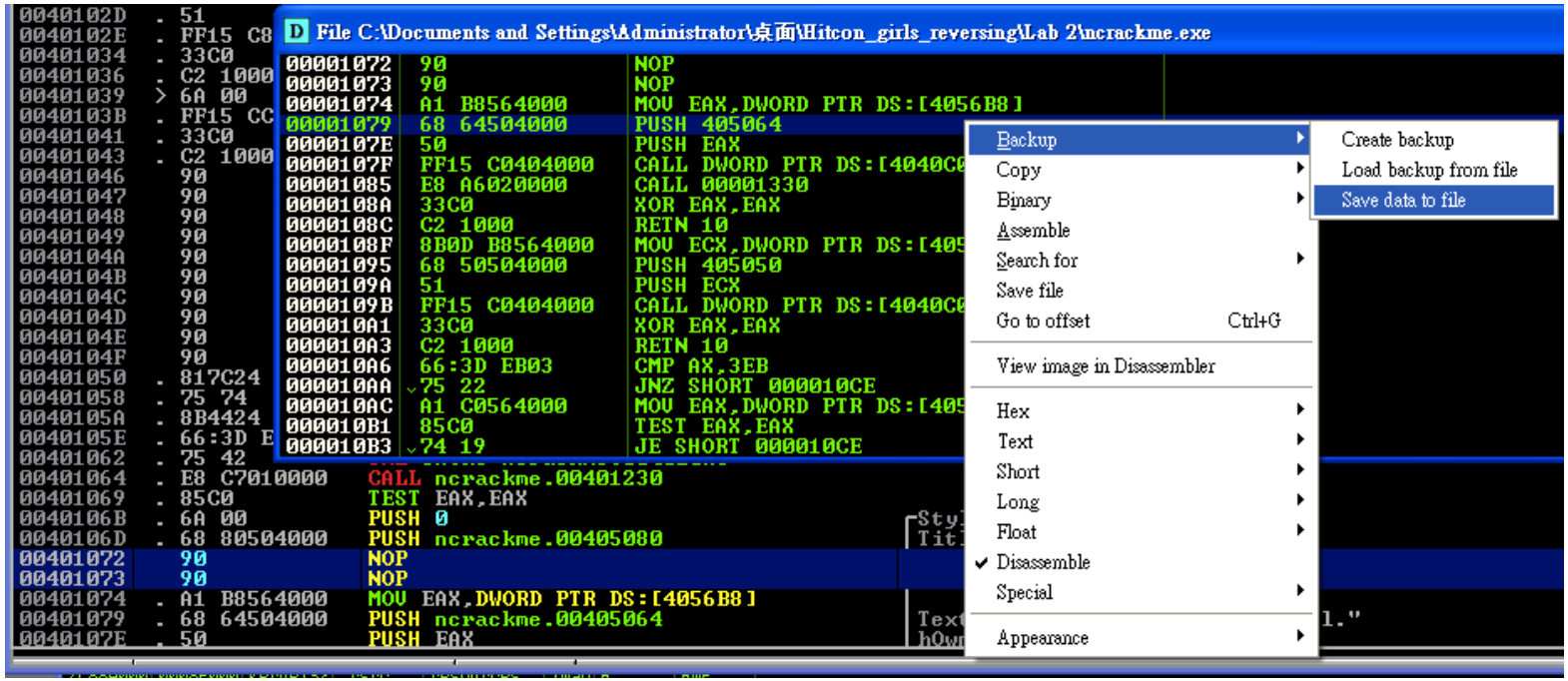
如何儲存修改過的檔案

- 之前的Lab都在記憶體中作修改，下次重新啟動效果就消失



如何儲存修改過的檔案

- 按右鍵→ “Backup” → “Save data to file”
- 另存新檔即可
- 現在執行看新存的檔案吧



Charles@teamt5.net

QUESTION?