INTRODUCTION TO THE CRACKING WITH OLLYDBG

FROM CRACKLATINOS

(_kienmanowar_)

Một cái đầu lạnh để vững vàng, một trái tim đỏ lửa để yêu và làm việc hết mình!

I. Giới thiệu chung

Khà khà, Giáng sinh rồi … có được chút thời gian rảnh rỗi tôi lại bắt tay vào viết tiếp bộ tutor này. Hi vọng các bạn vẫn còn hứng thú để đọc những gì tôi viết ☺. Ở phần 13 của loạt tuts về Ollydbg, tôi đã hướng dẫn các bạn cách kiểm tra xem file có bị pack hay không, cách tìm các điểm quan trọng để tiếp cận mục tiêu, phân tích chi tiết hoạt động của Crackme CrueHead thông qua việc trace và analyze code để từ đó tìm ra một real serial cho chuỗi Name nhập vào. Như vậy, qua bài viết đó tôi đã truyền tải tới các bạn những kinh nghiệm thực tế khi làm việc với một crackme đơn giản nhưng cũng sẽ là tiền đề cho các bạn khi gặp các crackme hoặc các chương trình khác sau này….Ở phần 14 này chúng ta sẽ quay lại làm việc với các target do lão Rincardo đưa ra, cụ thể hơn là tập trung vào chủ đề Fishing Serial ☺. N0w….L3t's G0 !!!!

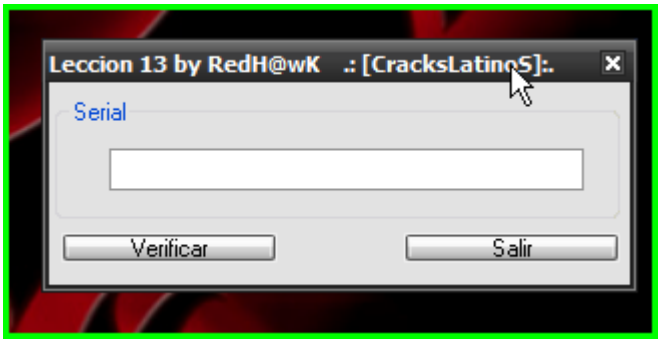
II. Fishing Serial ☺

Chà Fishing Serial tức là gì nhỉ? Nghe như kiểu chúng ta đang đi câu cá, giữa một hồ cá rộng và sâu, làm sao ta câu được một con cá ưng ý ☺… trong ngữ cảnh của Cracking thì ý nghĩa cũng gần như vậy. Fishing Serial ở đây có nghĩa là chúng ta đi câu Serial, mà phải là valid Serial nhé chứ câu lung tung là mệt và dễ stress. Đối với những bạn mới vào nghề thì việc tìm được một valid Serial luôn mang lại một cảm giác lâng lâng khó tả như tìm được một "kho báu" giữa lòng đại dương ☺. Hồi tôi chập chững lọ mọ đọc tutor và cặm cụi mò theo, cho đến khi tìm được serial hợp lệ tự nhiên cảm thấy sướng khó tả, lúc đó nếu có ai ở bên cạnh chắc tôi sẽ kéo vào và chỉ trỏ để khoe những gì mình đã làm, dù chắc gì người đó đã hiểu mình đang làm gì, có khi lại cho mình đang bị chập lolz.

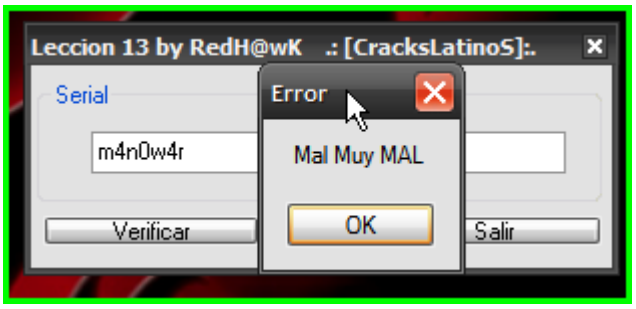
Trong phần 14 này tôi sẽ hướng dẫn các bạn làm việc với dạng Hardcoded Serial (đây là một dạng cơ bản và đơn giản), có nghĩa là dạng Serial cố định không được tính toán dựa trên Name nhập vào, cũng không thay đổi khi bạn chạy trên bất kì máy nào (tức là Serial đó valid trên mọi máy). Có bạn sẽ cho rằng vậy thì dễ quá, viết làm gì? Nhưng xin thưa, phải có dễ thì mới có khó, phải đi từ basic rồi mới tới advanced. Quan điểm của tôi là cứ từ từ mà tiến, không đi đâu mà phải vội vàng. Ok giờ chúng ta vào thực hành luôn nhé, trong bài này ta sẽ làm việc với hai Crackme của lão **RedH@wk** (lão này cũng là một thành viên trụ cột trong CrackLatinos).

1. LESSON 13 HARDCODED 1

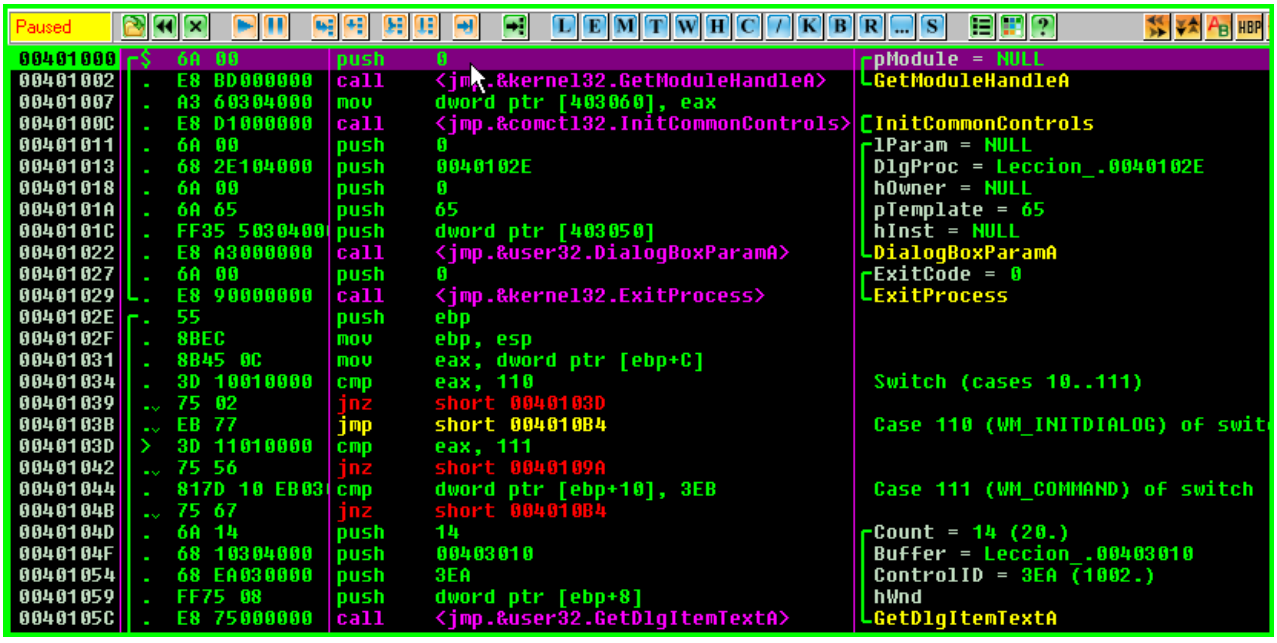
Trước khi load vào Olly, ta chạy thử xem mặt mũi cái crackme này thế nào :



Chà thấy có mỗi cái Textbox cho phép nhập Serial vào, còn mấy cái nút ở dưới toàn tiếng TBN ☹. Nhưng chắc 1 cái dùng để kiểm tra tính hợp lệ của Serial nhập vào, cái còn lại chắc dùng để thoát crackme.Ok, ta nhập đại một serial vào và nhấn thử nút Verificar :



Ặc thông báo bằng tiếng TBN, nhưng may cái Caption là Error, đủ để cho chúng ta biết rằng cái Serial mà ta nhập vào là không hợp lệ. Vậy thì ta phải **"đi câu"** để túm được valid Serial nào ☺. Load cái crackme vào trong Olly :



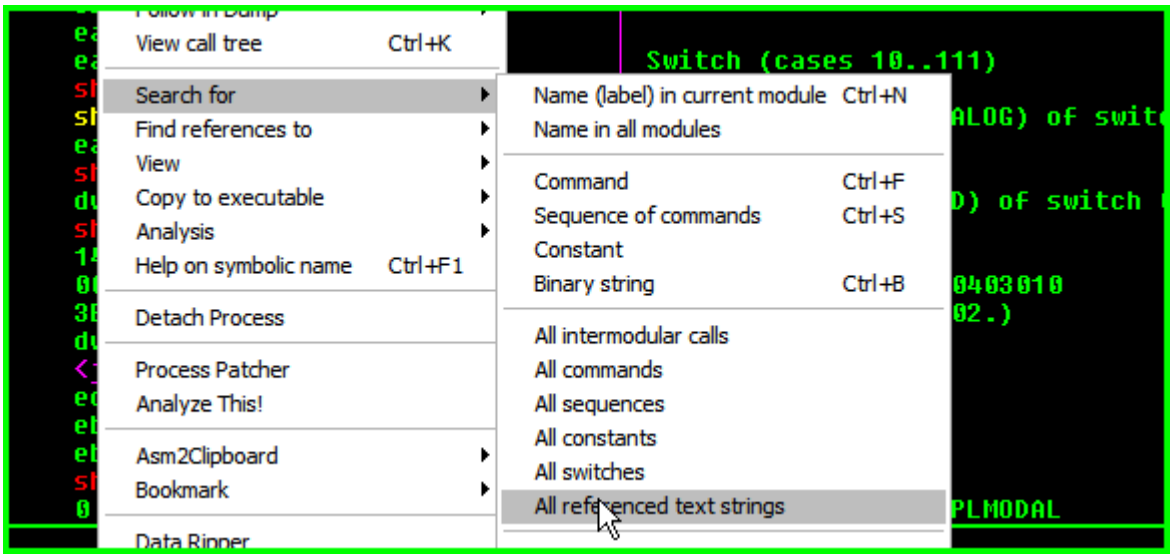
Sau khi load vào Olly, ta dừng lại tại EP của crackme như các bạn thấy ở hình minh họa trên. Dòm qua cái code thấy quá trong sáng, các bạn sẽ bắt gặp cái thông báo Error :

```
00401078  |.  68 35204000    push    00402035      ; |Text = "Mal Muy MAL"
```

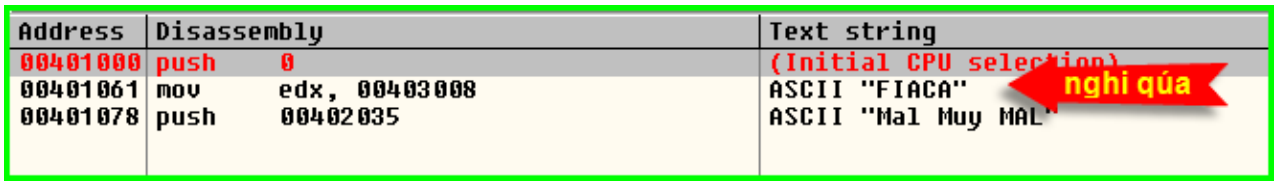
và một cái thông báo khác, tôi đoán chắc là thông báo Serial hợp lệ :

```
0040108B      |.   68   28204000   push   00402028      ;   |Text   =   "Muy
BIEN",A1,"",A1,"",A1,"",A1,""
```

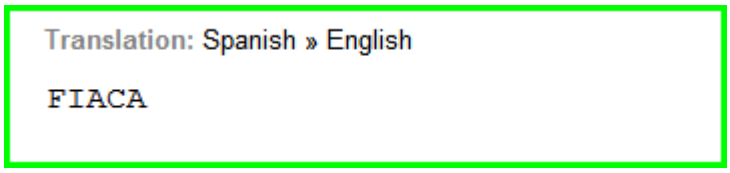
Vậy Serial này ở đâu? Vì nó là dạng HardedCode, tức là người code đã include nó vào trong chương trình rồi, ta thử mò bằng cách tìm toàn bộ các String xem thế nào. Chuột phải và chọn như sau :



Ta có được kết quả :

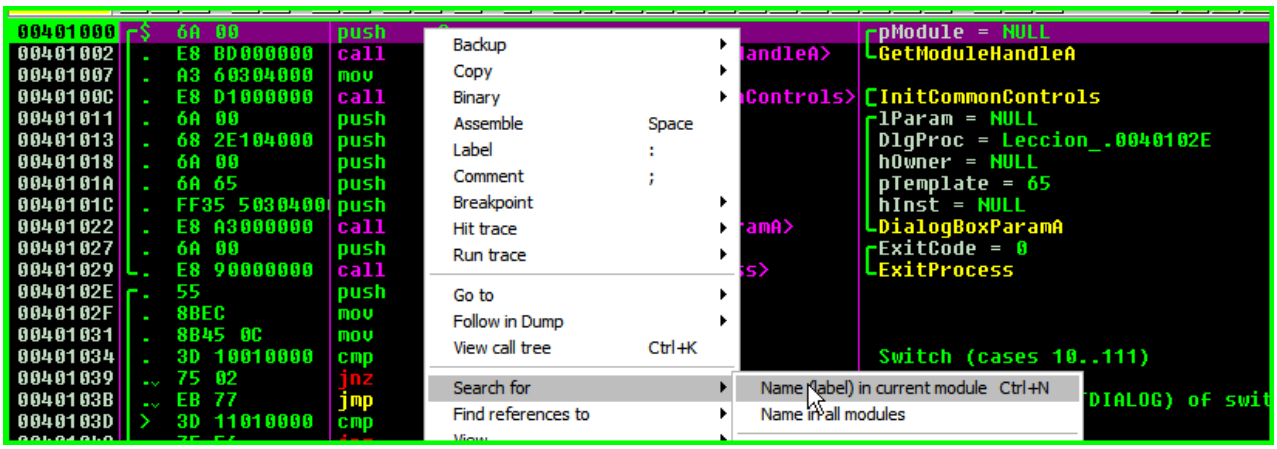


Ồ, kết quả có được cũng không nhiều, dòng thứ 3 thì ta đã biết được nó là thông báo lỗi rồi, còn dòng thứ hai là gì nhỉ, thử đưa lên anh google nhờ anh ấy dịch hộ xem nó có nghĩa gì không ☺ :

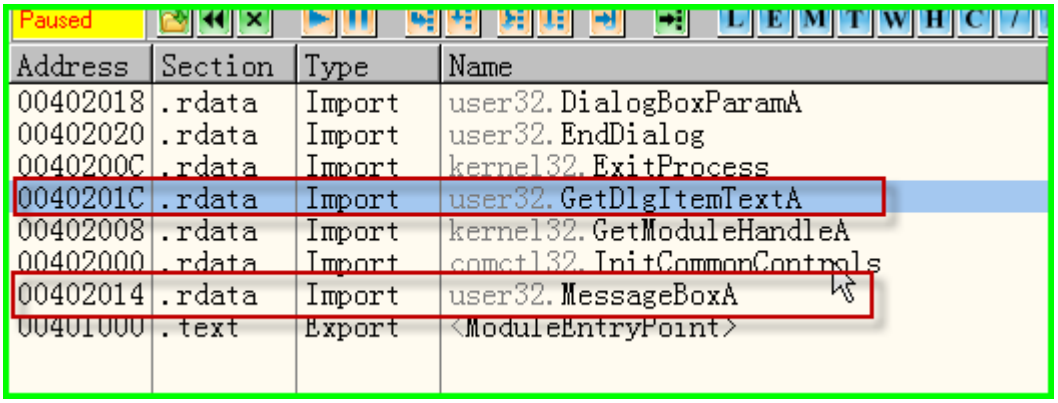


Khà khà kết quả là nó vẫn là chính nó lolz, vậy có thể nghi ngờ nó là valid Serial. Tuy nhiên đây mới chỉ là nghi ngờ thôi, tôi không khuyến cáo các bạn áp dụng cách này để tìm Serial đơn giản vì lý do sau :
"Do đây là crackme đơn giản cho nên tác giả đã code hết sức gọn nhẹ để chúng ta thực hành, cho nên kết quả trả về cho việc tìm kiếm chỉ là hai dòng text duy nhất. Nhưng với các target phức tạp khác, kết quả trả về có thể lên đến hàng trăm hoặc hàng nghìn dòng, việc dò và kiểm thử xem có đúng hay không là điều khó thực hiện và mất thời gian."

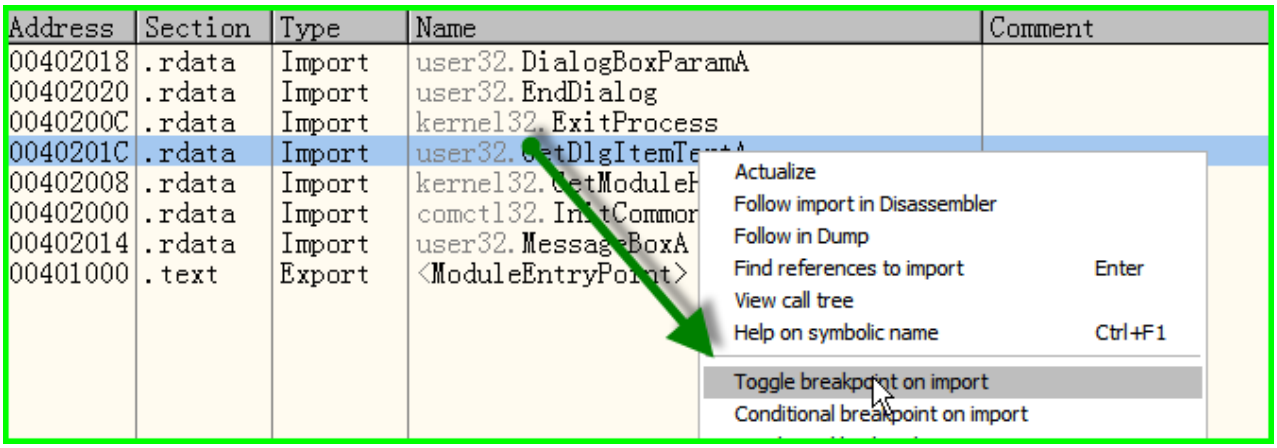
Giờ chúng ta đã có điểm nghi ngờ là dòng text thứ hai, nếu nhanh gọn các bạn có thể chạy crackme và nhập vào để kiểm tra tính hợp lệ của nó. Tuy nhiên để chắc chắn hơn chúng ta cần chứng minh đúng là nó chứ không phải dựa vào phán đoán nhất thời. Ta tiến hành tìm kiếm các hàm API mà crackme này sử dụng :



Olly trả về kết quả cho chúng ta như sau :

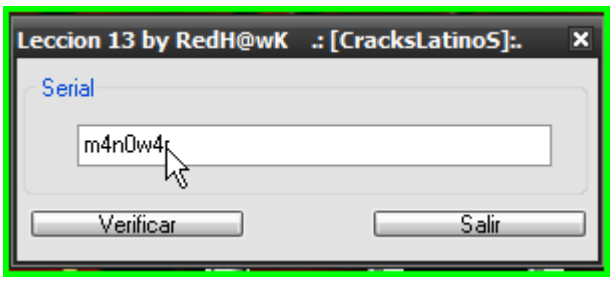


Chà trong danh sách này có hai API mà chúng ta đã khá quen thuộc từ các bài viết trước rồi, ta quan tâm nhất tới hàm `user32.GetDlgItemTextA`. Đặt BP tại hàm này như sau :

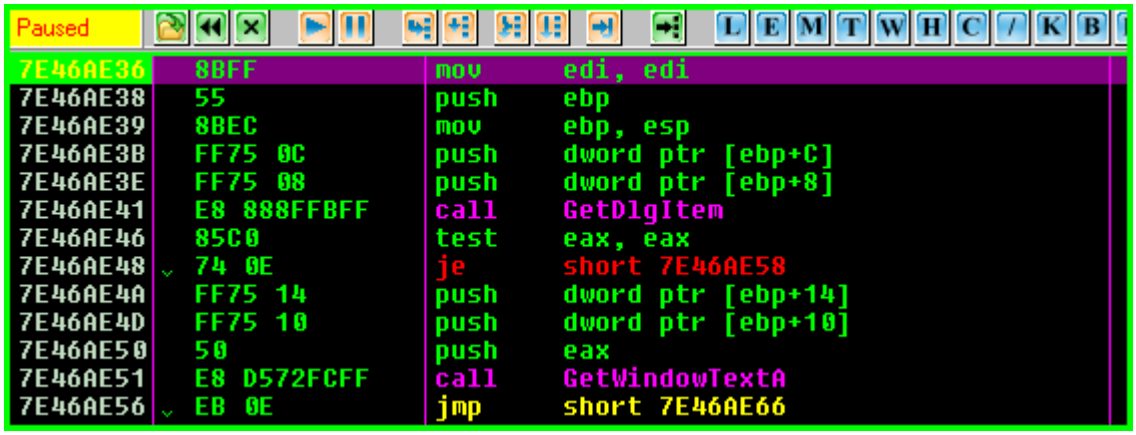


Kiểm tra xem BP đã được thiết lập chưa tại cửa sổ quản lý BP :

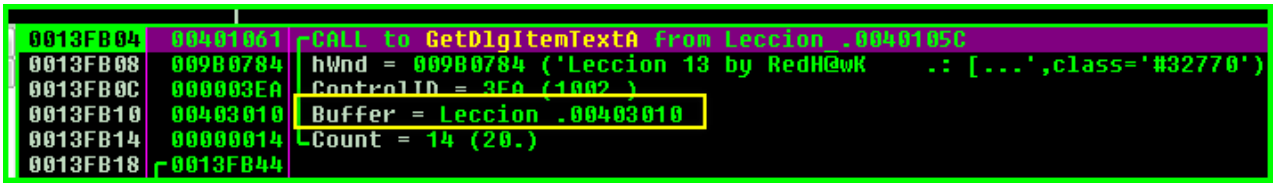
Ok, sau khi đặt BP xong ta run chương trình và tiến hành nhập Serial :



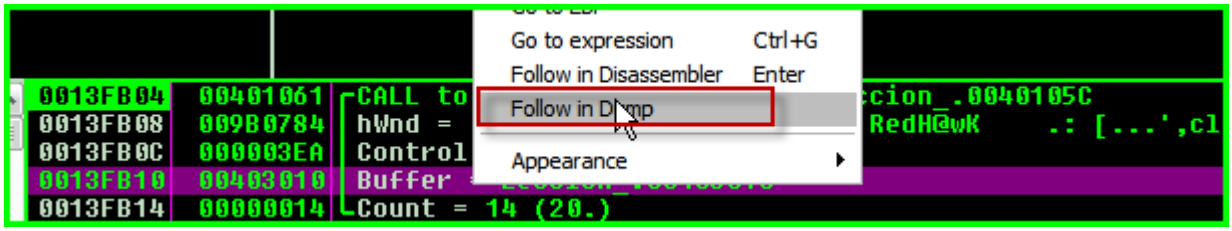
Sau khi nhập xong các bạn nhấn Verify, ngay lập tức Olly sẽ break :



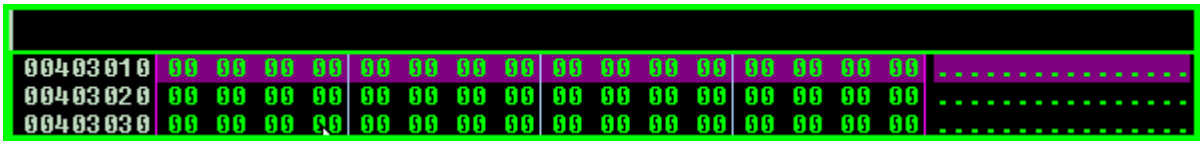
Ta đang dừng lại tại **GetDlgItemTextA**, dòm qua cửa sổ Stack ta có thông tin như sau :



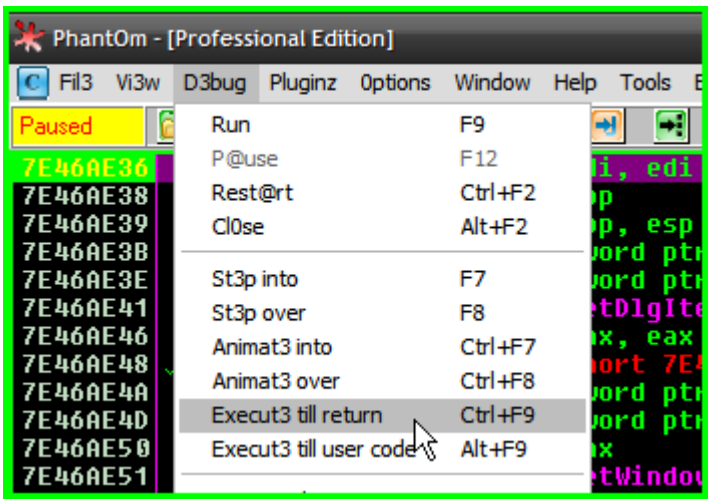
Ở đây ta cần quan tâm tới Buffer, đây sẽ là nơi nhận lưu Serial mà chúng ta đã nhập vào. Chuột phải vào Buffer và chọn Follow in Dump :



Do hàm API của chúng ta chưa được thực hiện nên vùng buffer hiện thời vẫn còn trống :

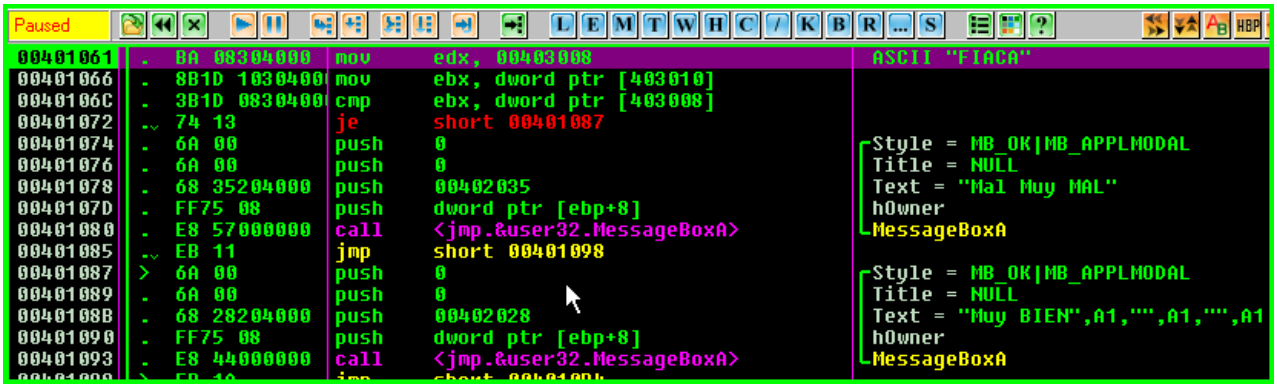


Giờ chúng ta cho thực thi hàm API **GetDlgItemTextA** như sau :



Olly sẽ dừng lại tại lệnh RET, lúc này quan sát vùng Buffer ta có được kết quả như sau :

Không cần giải thích chắc các bạn cũng biết được đó là gì rồi! Giờ nhấn F8 để trace qua lệnh Ret để trở về code chính của chương trình :



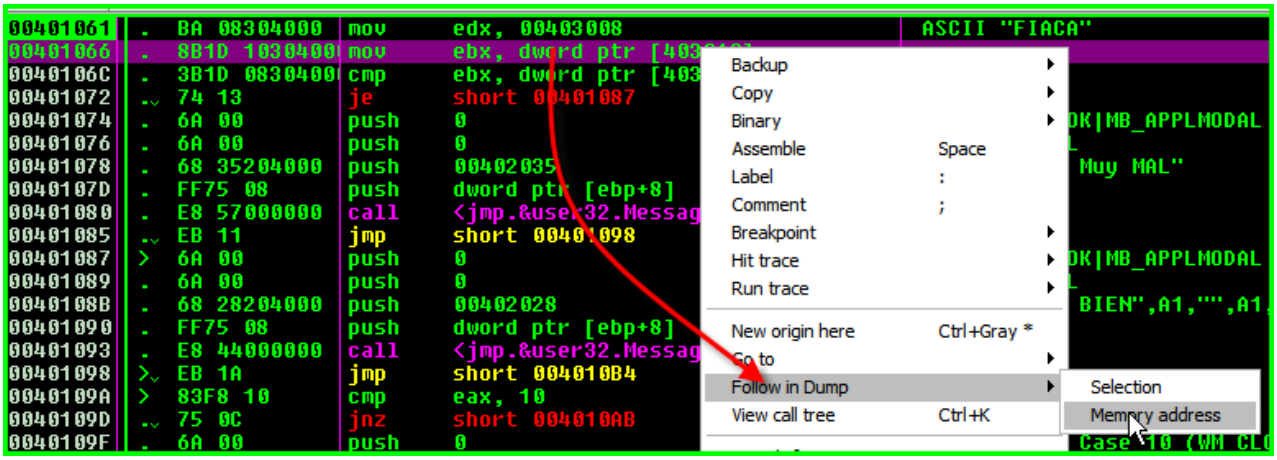
Ok, ta dừng lại quan sát và phân tích một chút. Để ý các bạn sẽ thấy có lệnh so sánh và lệnh nhảy, phụ thuộc vào kết quả so sánh mà lệnh nhảy sẽ đưa chúng ta đến một trong hai thông báo :

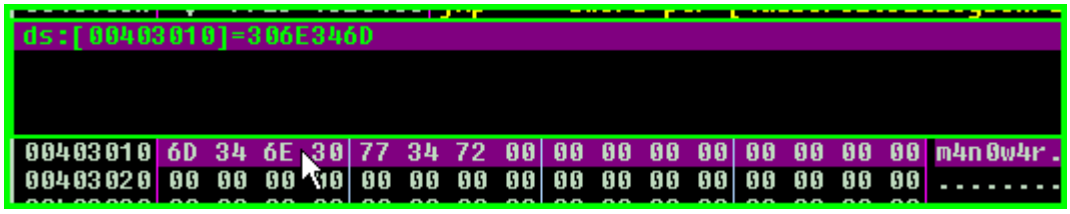
1. Nếu sai chúng ta sẽ được nhận thông báo "Mal Muy MAL" («bad, very bad» - approx. Ed.)
2. Nếu đùng chúng ta sẽ nhận thông báo "Muy BIEN" («very good »- Prim.per.).

Rõ ràng trong trường hợp của chúng ta chúng ta muốn nhận được thông báo **“Very good”.** Vậy ta trace code để xem điều gì sẽ xảy ra tiếp theo. Đầu tiên ta sẽ thấy chương trình chuyển vào thanh ghi edx một hardcoded string là FIACA :

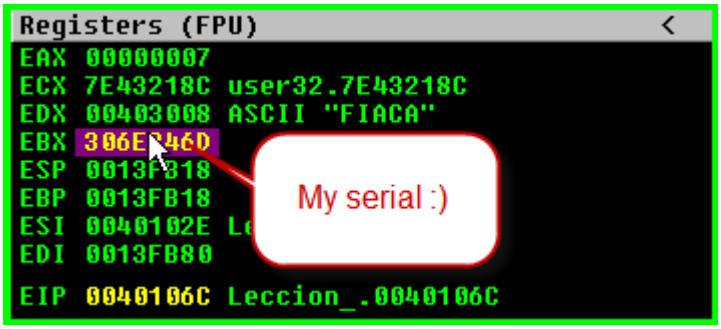
```
00401061  |.  BA 08304000   mov     edx, 00403008   ;  ASCII "FIACA"
```

Tiếp theo tại `0x00401066`, ta thấy thanh ghi ebx sẽ nhận được giá trị từ vùng nhớ [403010], mà theo phân tích ở trên thì đây chính là vùng Buffer nhận vào chuỗi Serial của chúng ta :

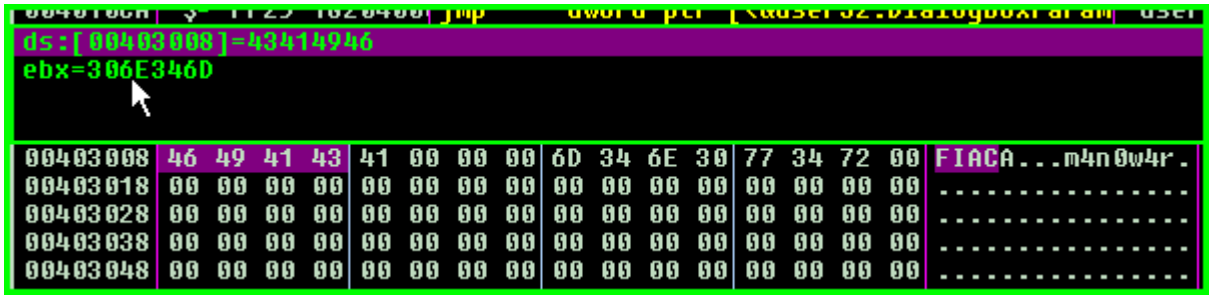




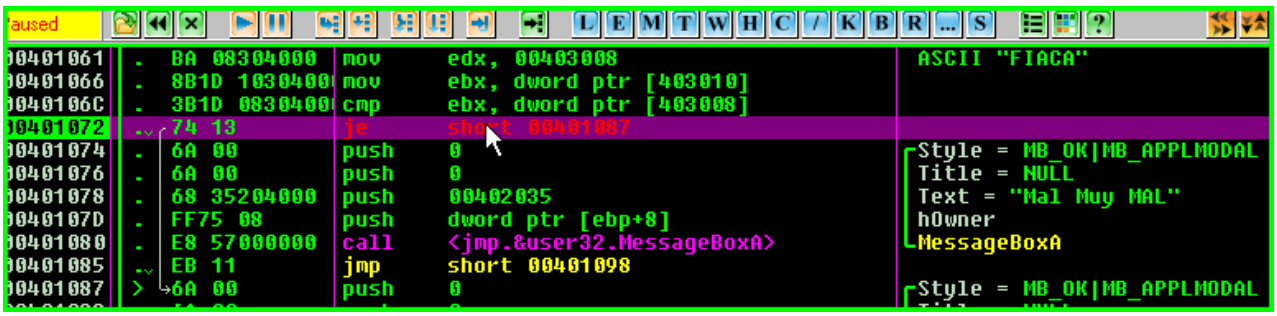
Trace qua lệnh này và quan sát thanh ghi EBX, kết quả như sau :



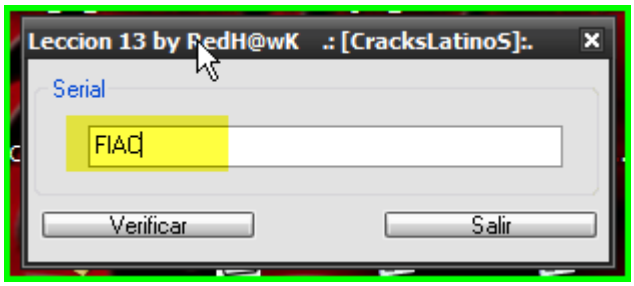
Sau khi trace qua lệnh `00401066 |. 8B1D 10304000 mov ebx, dword ptr [403010]`, ta sẽ đến lệnh CMP. Ta thấy nội dung của thanh ghi ebx sẽ được đem đi so sánh với vùng chứa chuỗi hardcoded.Ta follow in dump để quan sát giá trị :



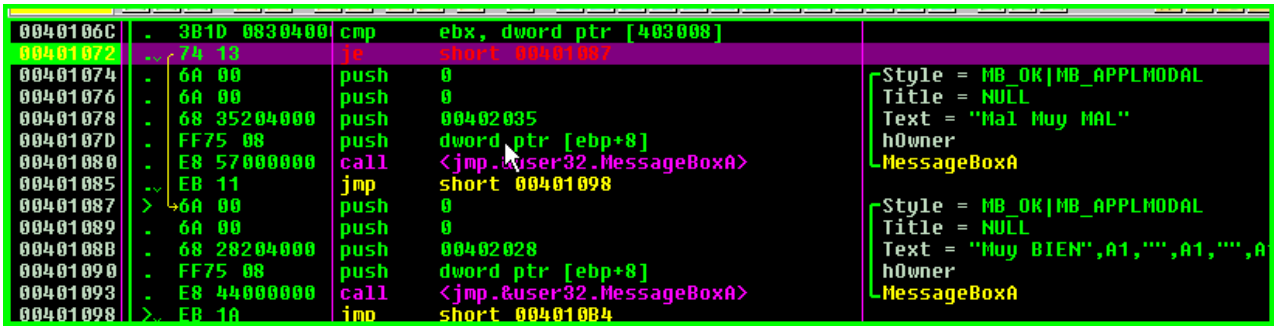
Theo như hình trên thì các bạn có thể thấy rằng, không phải toàn bộ chuỗi Hardcoded sẽ được đem đi so sánh với toàn bộ chuỗi serial mà ta nhập vào,mà ở đây chương trình chỉ lấy hai dword (tức là 4 chữ cái đầu tiên) đem so sánh với nhau. Nếu giống nhau thì quá tốt, còn không thì ☹.Giờ tôi trace qua lệnh CMP :



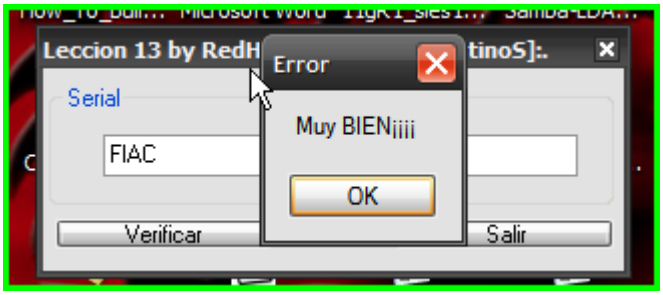
Đương nhiên là lệnh nhảy sẽ không thực hiện ví lý do Serial của tôi nhập vào không trùng khớp với Serial mà chương trình đưa ra. Bây giờ tôi đã biết được Serial của chương trình là gì rồi, đó chính là chuỗi "**FIAC**", không cần phải đầy đủ hết cả chuỗi vì như tôi đã phân tích ở trên. Chúng ta bỏ BP tại **GetDlgItemTextA** đi, sau đó đặt BP tại lệnh nhảy ở trên. Tiếp theo Restart Olly và cho thực thi chương trình :



Nhập valid Serial vào, sau đó nhấn Verify. Olly sẽ break :



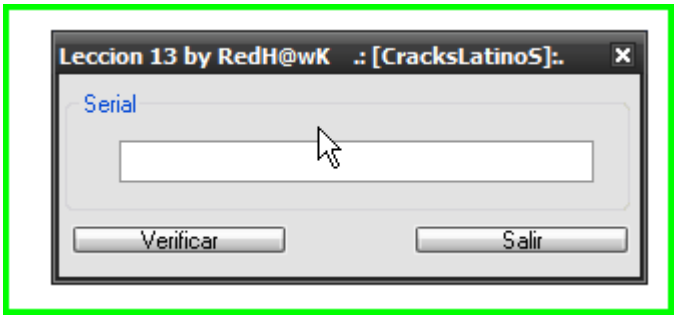
Khà khà Serial đã hợp lệnh, câu lệnh so sánh đã thành công và lệnh nhảy sẽ được thực hiện.Nhấn **F9** phát nữa nào :



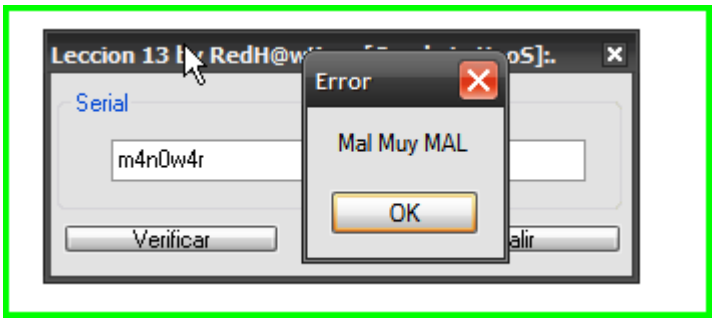
Hola, thành công rồi. Crackme 1 đến đây là kết thúc.

2. LESSON 13 HARDCODED 2

Xử xong crackme1 rồi, giờ chúng ta chuyển qua crackme2. Chạy thử nó xem thế nào đã, đây là một bước cần thiết để thu thập thông tin :



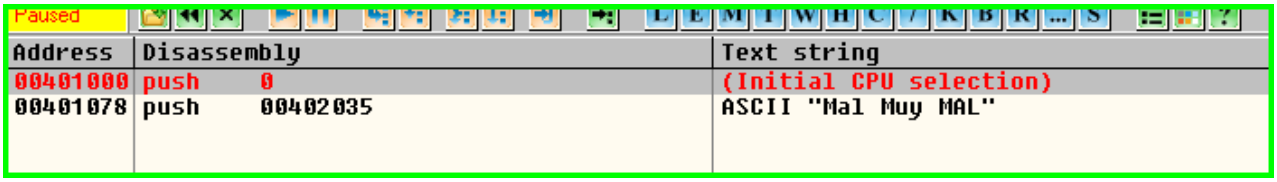
Nhập thử Serial và nhấn Verify :



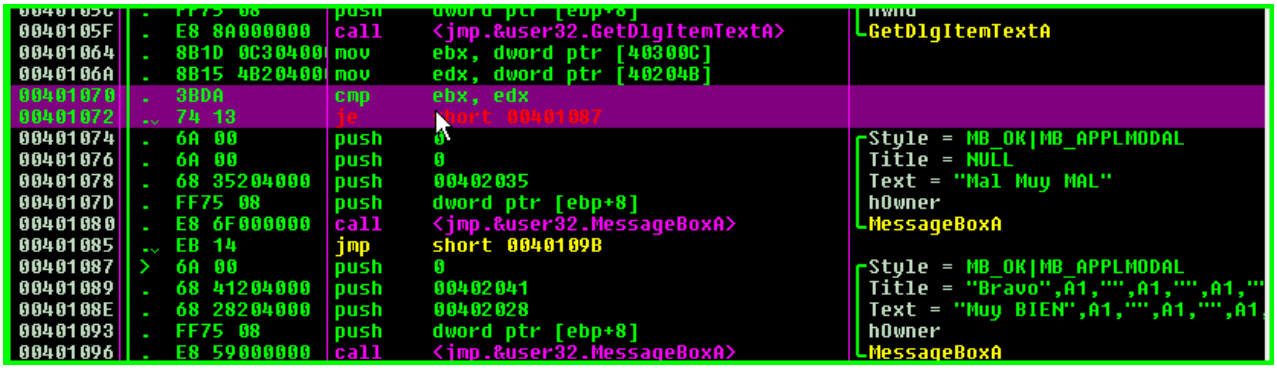
Chà thông báo trên thấy quen quá ☺. Giờ ta load Crackme vào Olly để tìm hướng giải quyết bài toán.Trong Olly ta có được như sau :

```
00401000  6A 00        push    0                       pModule = NULL
00401002  E8 D5000000  call    <jmp.&kernel32.GetModuleHandleA>  GetModuleHandleA
00401007  A3 08304000  mov     dword ptr [403008], eax
0040100C  E8 E9000000  call    <jmp.&comctl32.InitCommonControls>  InitCommonControls
00401011  6A 00        push    0                       lParam = NULL
00401013  68 2E104000  push    0040102E                DlgProc = Leccion_.0040102E
00401018  6A 00        push    0                       hOwner = NULL
0040101A  6A 65        push    65                      pTemplate = 65
0040101C  FF35 0830400 push    dword ptr [403008]      hInst = NULL
00401022  E8 BB000000  call    <jmp.&user32.DialogBoxParamA>  DialogBoxParamA
00401027  6A 00        push    0                       ExitCode = 0
00401029  E8 A8000000  call    <jmp.&kernel32.ExitProcess>  ExitProcess
0040102E  55           push    ebp
0040102F  8BEC         mov     ebp, esp
00401031  8B45 0C      mov     eax, dword ptr [ebp+C]
00401034  3D 10010000  cmp     eax, 110                Switch (cases 10..111)
00401039  75 05        jnz     short 00401040
0040103B  E9 8C000000  jmp     004010CC                Case 110 (WM_INITDIALOG) of swit
00401040  3D 11010000  cmp     eax, 111
00401045  75 6B        jnz     short 004010B2
00401047  817D 10 EB03 cmp     dword ptr [ebp+10], 3EB  Case 111 (WM_COMMAND) of switch
0040104E  75 4D        jnz     short 0040109D
00401050  6A 14        push    14                      Count = 14 (20.)
00401052  68 0C304000  push    0040300C                Buffer = Leccion_.0040300C
00401057  68 EA030000  push    3EA                     ControlID = 3EA (1002.)
0040105C  FF75 08      push    dword ptr [ebp+8]       hWnd
0040105F  E8 8A000000  call    <jmp.&user32.GetDlgItemTextA>  GetDlgItemTextA
```

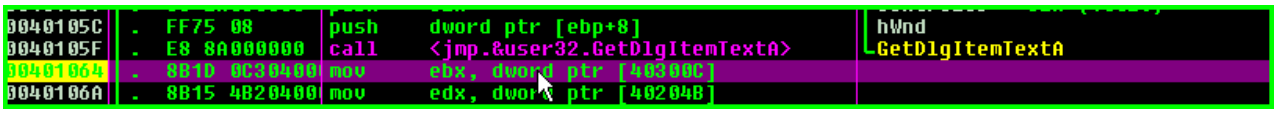
Như các bạn thấy code trông giống hệt crackme1, ta thử search string xem có tìm được vàng không nhé :



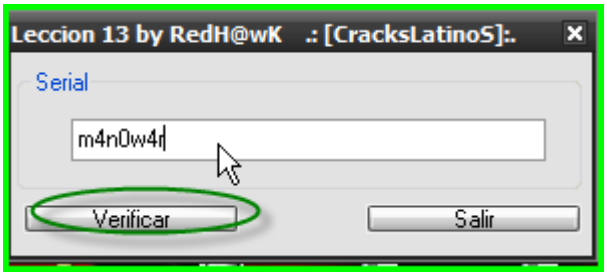
Ái chà, crackme2 này có vẻ khó hơn crackme1 rồi đây. Thông tin ta có được chỉ vẻn vẹn có mỗi một dòng thông báo lỗi, vậy là đầu mối tìm kiếm này đã không có tác dụng. Quay trở về cửa sổ CPU của Olly, cuộn chuột xuống ta sẽ thấy đoạn code so sánh :



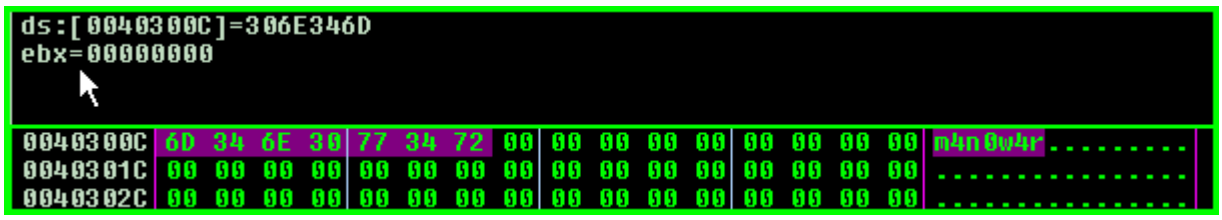
Tại đây ta cũng không thấy có dòng nào tương tự như crackme1, chẳng thấy có thông tin nào tương tư như 'FIACA' mà ta có được với crackme1 cả lolz ☺. Do code của crackme2 tương tự crackme1 cho nên phần phân tích tôi bỏ qua, ta đi thẳng vào vấn đề chính. Tôi đặt BP tại lệnh mov bên dưới lời gọi tới hàm **GetDlgItemTextA**.



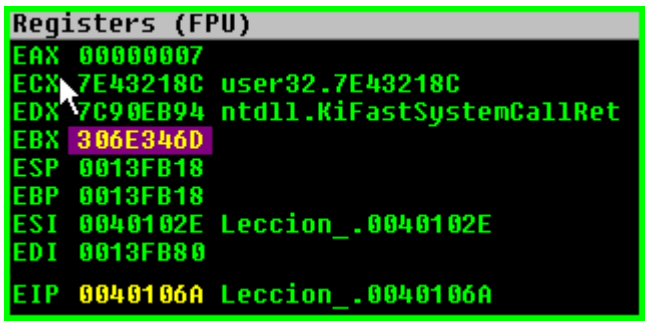
Nhấn **F9** để thực thi crackme, sau đó nhập đại một cái Serial vào rồi nhấn Verify :



Olly sẽ break tại chỗ mà ta đặt BP, ta Follow in Dump để biết nội dung tại [40300C] là gì :



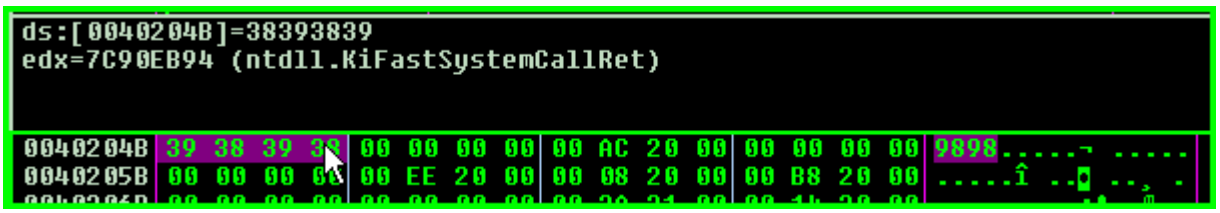
Ta thấy 4 bytes sẽ được đưa vào thanh ghi ebx. Quan sát thanh ghi :



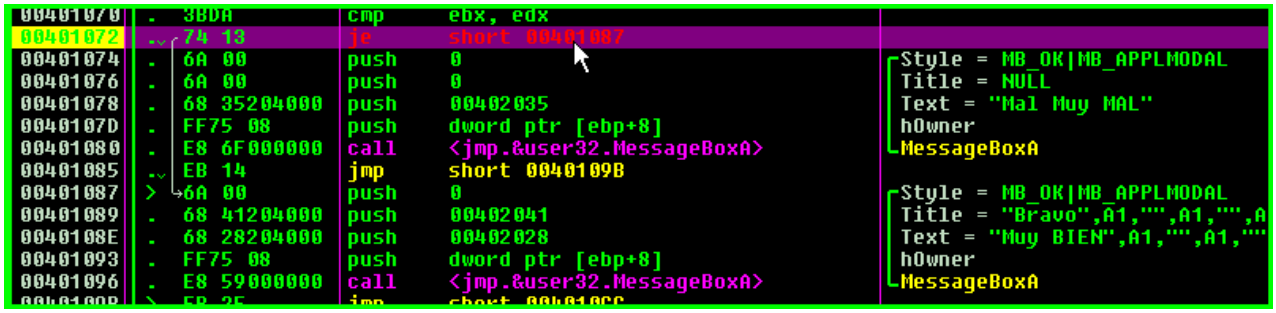
Vậy là thanh ghi ebx đã giữ thông tin về 4 kí tự đầu tiên trong chuỗi Serial mà tôi nhập vào. Bây giờ chúng ta đang dừng tại câu lệnh :

```
0040106A  |.  8B15 4B204000 mov     edx, dword ptr [40204B]
```

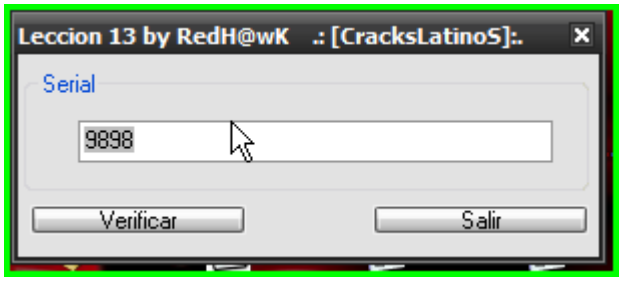
Không hiểu thanh ghi edx được mov thông tin gì vào nhỉ? Điểm nghi ngờ là ở đây, vì ở bên dưới tôi thấy lệnh cmp đem hai thanh ghi ebx và edx so sánh với nhau. Vậy ta có thể khẳng định chắc chắn rằng, vùng nhớ [40204B] đang nắm giữ một thông tin rất quan trọng, đó chính là valid Serial. Ta Follow in dump để biết thông tin đó là gì :



Khà khà, vậy là Serial mà chúng ta cần tìm là : **9898**. Quay lại cửa sổ code, bỏ BP đã đặt trước đó đi, sau đó bạn đặt BP tại lệnh nhảy bên dưới lệnh cmp :

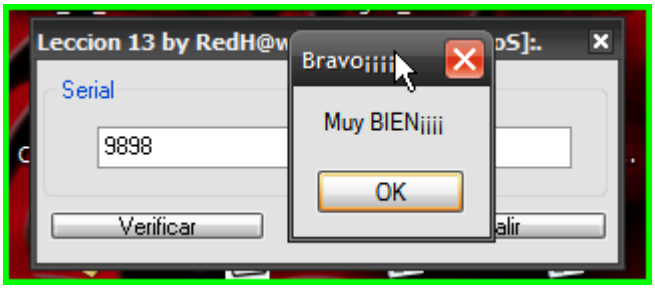


Restart lại Olly, sau đó nhấn F9 để run chương trình. Nhập valid Serial là 9898 vào và nhấn Verify:



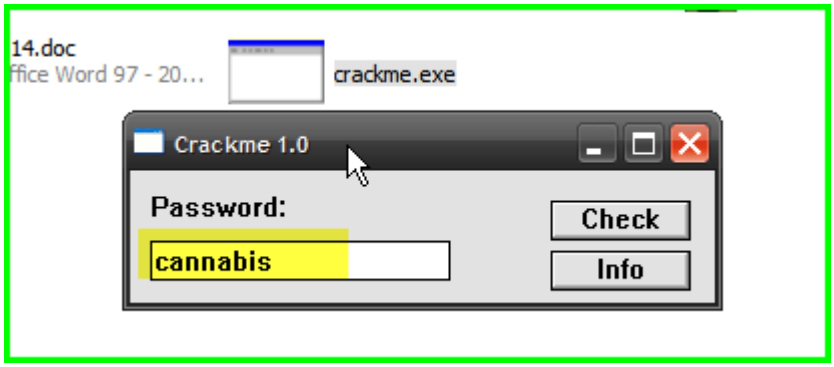
```
00401070  .  3BDA          cmp     ebx, edx
00401072  .  74 13         je      short 00401087
00401074  .  6A 00         push    0                          Style = MB_OK|MB_APPLMODAL
00401076  .  6A 00         push    0                          Title = NULL
00401078  .  68 35204000   push    00402035                   Text = "Mal Muy MAL"
0040107D  .  FF75 08       push    dword ptr [ebp+8]          hOwner
00401080  .  E8 6F000000   call    <jmp.&user32.MessageBoxA>  MessageBoxA
00401085  .  EB 14         jmp     short 0040109B
00401087  >  6A 00         push    0                          Style = MB_OK|MB_APPLMODAL
00401089  .  68 41204000   push    00402041                   Title = "Bravo",A1,"",A1,"",A1,"
0040108E  .  68 28204000   push    00402028                   Text = "Muy BIEN",A1,"",A1,"",A1
00401093  .  FF75 08       push    dword ptr [ebp+8]          hOwner
00401096  .  E8 59000000   call    <jmp.&user32.MessageBoxA>  MessageBoxA
```

Hehe, như các bạn thấy lệnh nhảy tới Good boy sẽ được thực hiện sau khi so sánh.Vậy là việc Fishing của chúng ta đã thành công, nhấn F9 để kiểm chứng kết quả cuối cùng :



Vậy là xong, crackme2 đã được xử lý nhẹ nhàng!!
Ok vậy là phần 14 của loạt tuts về Ollydbg đến đây là kết thúc, qua bài viết này tôi đã giới thiệu tới các bạn thế nào là Fishing Serial và hướng dẫn các bạn thực hành với hai Crackme rất đơn giản, ngoài ra trong phần 14 này cũng có thêm một target nữa của lão **Detten** để các bạn tự thực hành để kiểm chứng kiến thức và khả năng của mình. Hi vọng qua bài viết này của tôi đã cung cấp thêm một số kiến thức cơ bản giúp các bạn làm việc hiệu quả với Olly. Hẹn gặp lại các bạn trong các phần tiếp theo của loạt tutor này, By3 By3!! ☺

Solution for Detten's crackme :



Best Regards
[Kienmanowar]

--++--==[Greatz Thanks To]==--++--

My family, Computer_Angel, Moonbaby , Zombie_Deathman, Littleboy, Benina, QHQCrker, the_Lighthouse, Merc, Hoadongnoi, Nini ... all REA's members, TQN, HacNho, RongChauA, Deux, tlandn, light.phoenix, dump, dqtln, ARTEAM .... all my friend, and YOU.

--++--==[Thanks To]==--++--

iamidiot, WhyNotBar, trickyboy, dzungltvn, takada, hurt_heart, haule_nth, hytkl, moth, XIANUA, nhc1987, 0xdie, Unregistered!, akira, mranglex v..v.. các bạn đã đóng góp rất nhiều cho REA. Hi vọng các bạn sẽ tiếp tục phát huy ☺

I want to thank **Teddy Roggers** for his great site, Reversing.be folks(especially **haggar**), Arteam folks(**Shub-Nigurrath, MaDMAn_H3rCuL3s**) and all folks on crackmes.de, thank to all members of **unpack.cn** (especially **fly** and **linhanshi**). Great thanks to **lena151**(I like your tutorials).Thanx to Orthodox, kanxue, TiGa and finally, thanks to **RICARDO NARVAJA** and all members on **CRACKSLATINOS.**

>>>> If you have any suggestions, comments or corrections email me:

kienmanowar[at]reaonline.net