INTRODUCTION TO THE CRACKING WITH OLLYDBG

FROM CRACKLATINOS

(_kienmanowar_)

I. Lời nói đầu :

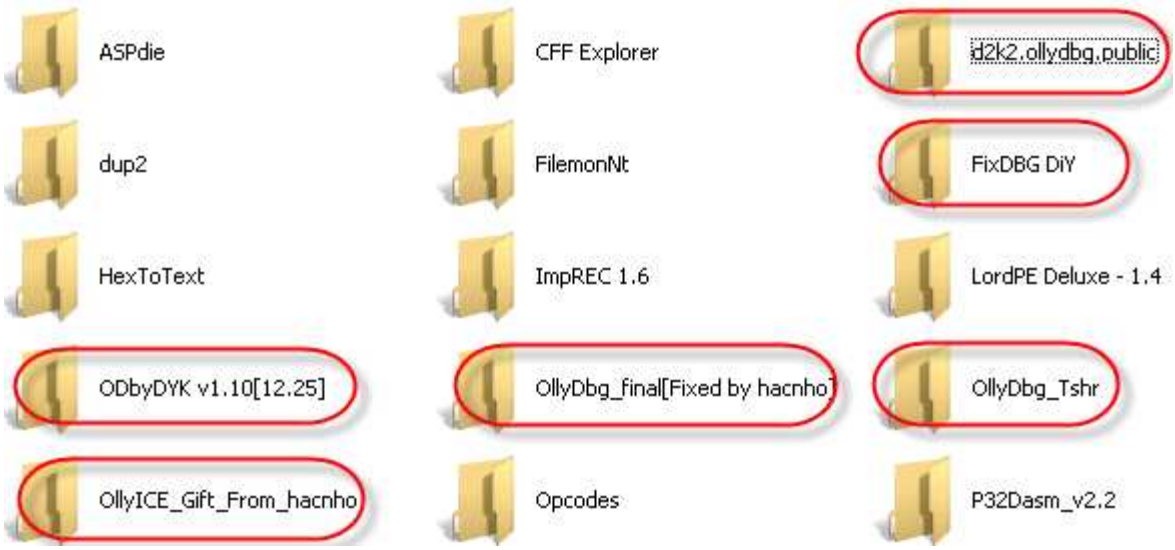
Một lần nữa gửi lời chào tới toàn thể anh em trong REA. Tình cờ qua bên site của lão **Ricardo Narvaja** thấy được bộ tut này khá hay và rất cơ bản cho tất cả những ai muốn tìm hiểu về cracking thông qua sự trợ giúp của chương trình debugger đã trở nên quá nổi tiếng, đó chính là **Ollydbg**. Tôi rất khoái các tut bên **Cracklatinos** nhưng ngặt nỗi toàn là tiếng TBN, nhưng thấy bộ tut này hay nên máu quá , quyết định trans từ TBN qua English, rồi từ Eng lại hì hục viết lại theo cách mình hiểu để truyền đạt những gì mình biết cho anh em. Ý tưởng chính của loạt tut này theo như tác giả của nó nói là nhằm cung cấp những kiến thức cơ bản nhất cho tất cả những ai chuẩn bị bắt đầu bước vào tìm hiểu nghệ thuật cracking với sự trợ giúp của Ollydbg. Mặc dù tiêu đề của tut là Introduction (tức là chỉ giới thiệu thôi) nhưng thực chất bộ tuts này sẽ cung cấp cho chúng ta một kiến thức nền tảng vững chắc để có thể đọc và hiểu được các tuts dành cho những người có trình độ advanced và đặc biệt là những tut sắp được giới thiệu trên Cracklatinos (hehe tác giả của nó quảng cáo ác quá), đồng thời thông qua loạt tuts này nó còn giúp chúng ta có khả năng áp dụng các kĩ thuật mới trong việc cracking.

II. Tại sao lại là Ollydbg ?

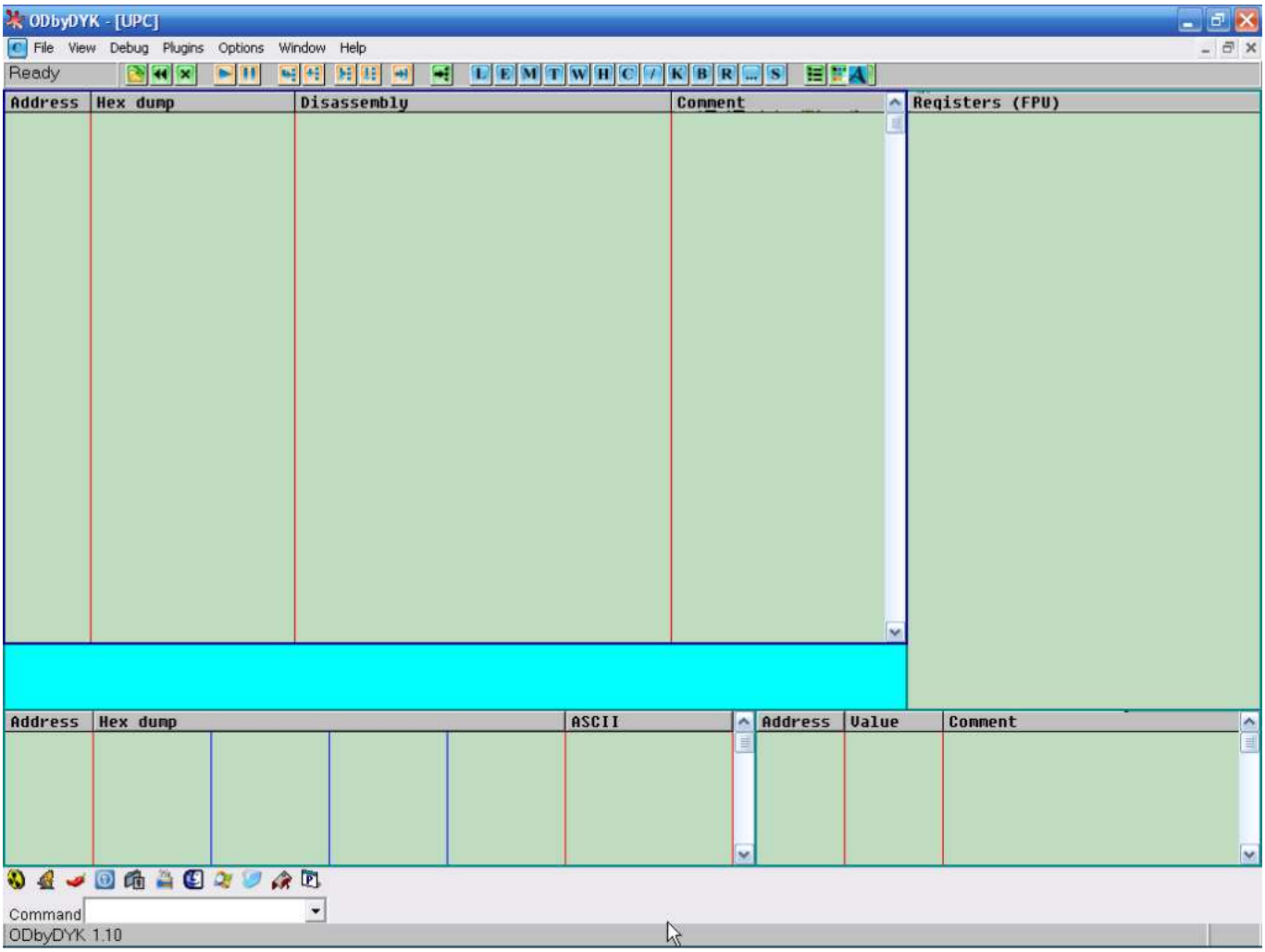
Tham gia vào REA điều đầu tiên có lẽ chúng ta thấy nhiều nhất đó là sự xuất hiện của **"Ollydbg"**, vậy tại sao lại là Ollydbg mà không phải là một công cụ nào khác. Ở đây chúng ta sẽ không bàn luận đến việc tạo ra một công cụ khác hay hơn, mạnh hơn Ollydbg cũng như không để cập tới việc chỉnh sửa lại một chương trình đã quá nổi tiếng từ lâu là SoftIce, chỉ đơn giản là những tín đồ cuồng tín của SoftIce đang dần dần chuyển qua xài Ollydbg bởi tính dễ dùng, không gây crash máy bất thình lình như SoftIce, được hỗ trợ bởi nhiều teams trên thế giới thông qua các Plugins cũng như các bản Ollydbg được mod lại nhằm chống lại các cơ chế anti-debug cũng như anti-Ollydbg, và vì một lý do đơn giản khác nữa đó là loạt tuts này dành riêng để nói về Ollydbg ☺.

III. Nhiệm vụ đầu tiên

Hì nhiệm vụ đầu tiên của chúng ta bây giờ là gì ? Do đây là tut viết về Olly nên việc chúng ta phải làm là đi tìm Olly ở đâu để còn load về mà xài. Thứ nhất bạn có thể lên home site của Olly là **ollydbg.de** để download, còn không thì trong REA có đưa rất nhiều link để download Ollydbg. Riêng bản thân tôi cũng sưu tầm được có lẽ gần chục bản Olly khác nhau, hic hic có lẽ là đợi ver 2.0 của Olly thôi ☺

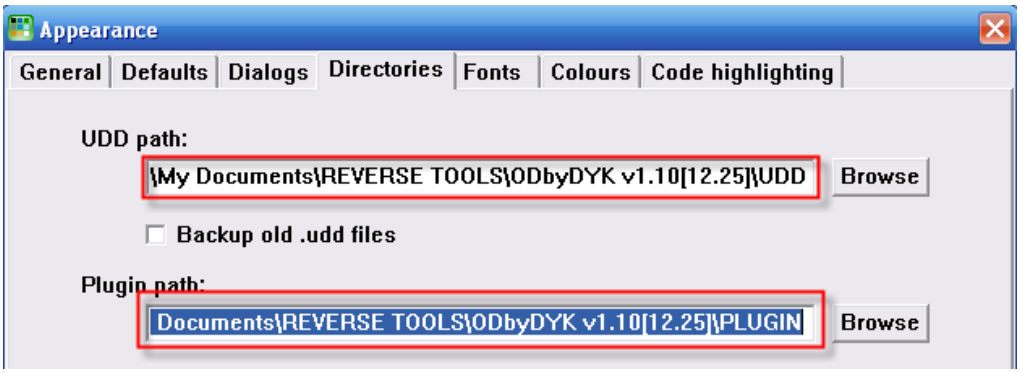


Khi download được Olly về rồi thì rất đơn giản chỉ việc extract nó ra rồi sử dụng, tôi khuyên bạn nên để chung tất cả công cụ liên quan đến RE, Cracking vào 1 thư mục, ví dụ như của tôi trên hình minh họa, như thế ta dễ dàng quản lý hơn. Okie coi như bạn đã có Ollydbg, chúng ta chỉ việc Run cái file **OLLYDBG.exe** là Olly hoạt động liền, không phức tạp về mặt cài đặt cũng như sử dụng như SoftIce. Giao diện của Ollydbg như sau :



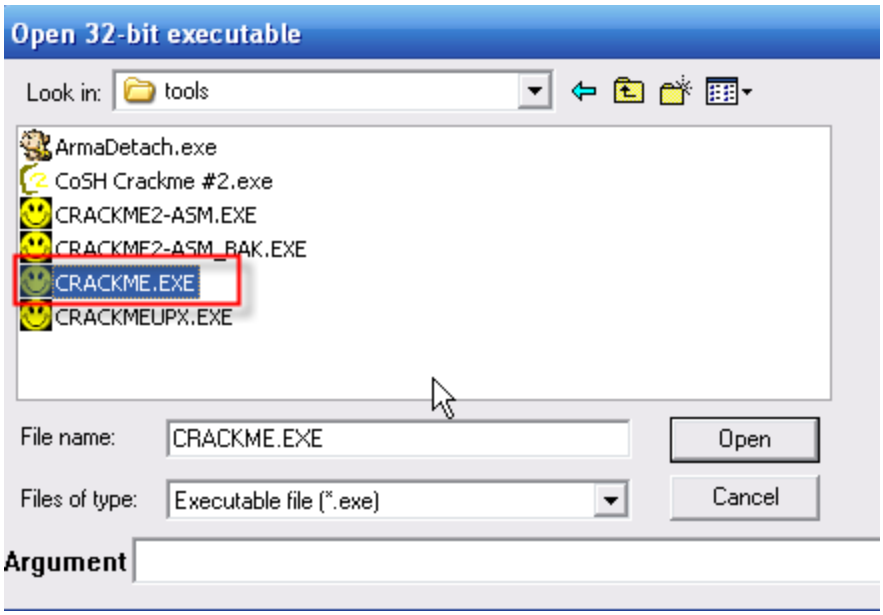
Đây là bản Ollydbg của tôi, đã được chỉnh sửa cũng như cấu hình lại. Nếu như các bạn download bản Ollydbg trên home site hoặc từ các nguồn khác có thể sẽ khác của tôi, và để có thể hiện thị menu Plugins thì các bạn làm như sau :

Chọn như hình trên hoặc vào **Options > Appearance** , chọn tab **Directories** và chỉnh lại đường dẫn tới thư mục **Plugins** và thư mục **UDD**.

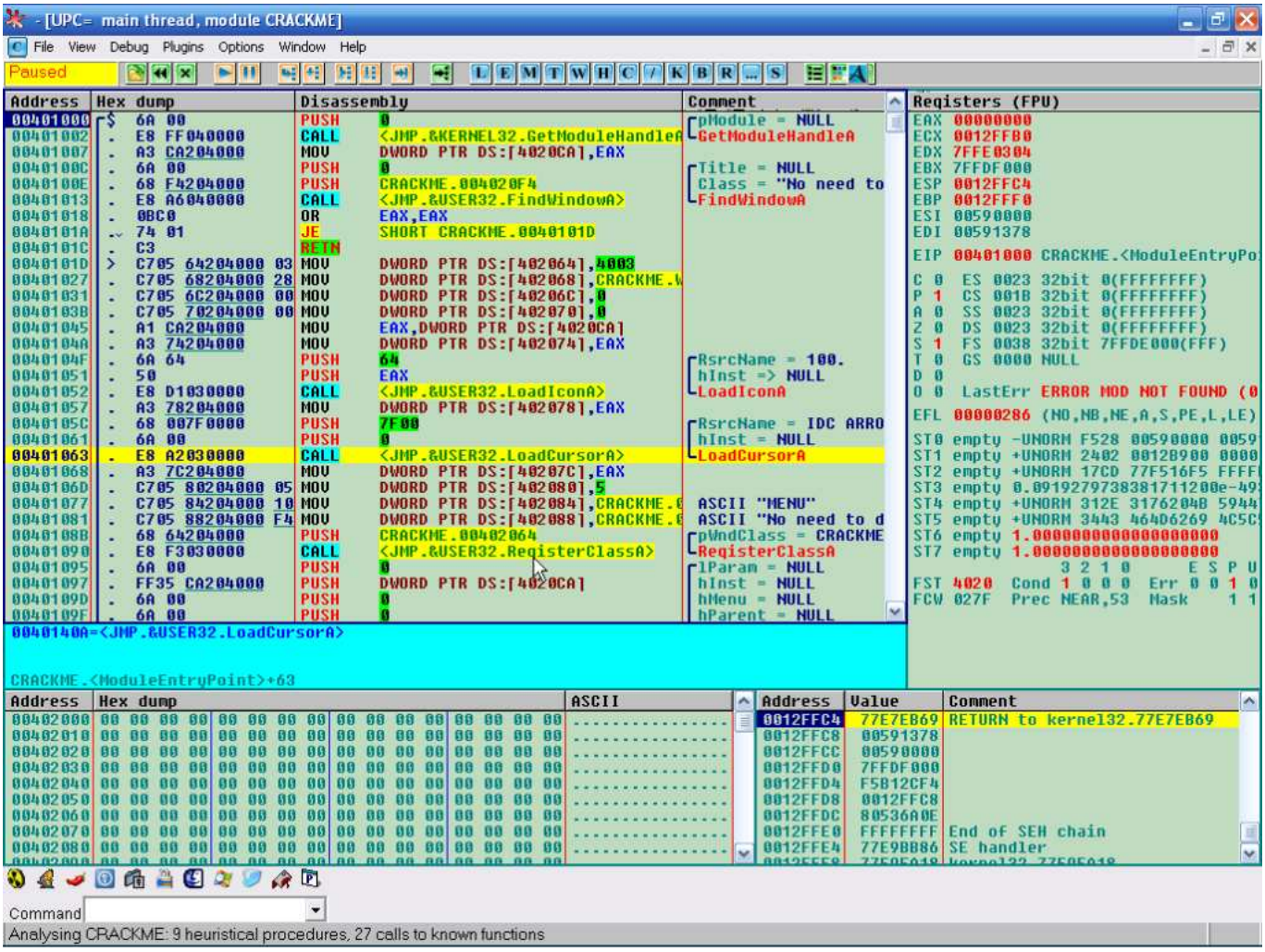


Sau đó nhấn Ok và chạy lại Olly thì sẽ thấy được menu Plugins. Phần tiếp theo, tôi sẽ giới thiệu tới các bạn chi tiết các cửa số chính trong Ollydbg và để minh họa cho các phần sau của bài viết, tôi sẽ sử dụng một Crackme rất nổi tiếng đó là : **CRACKME.EXE** của tác giả **CRUEHEAD**. Để load crackme này vào trong Olly ta nhấn chuột vào biểu tượng sau hoặc vào **File > Open (or F3)** :

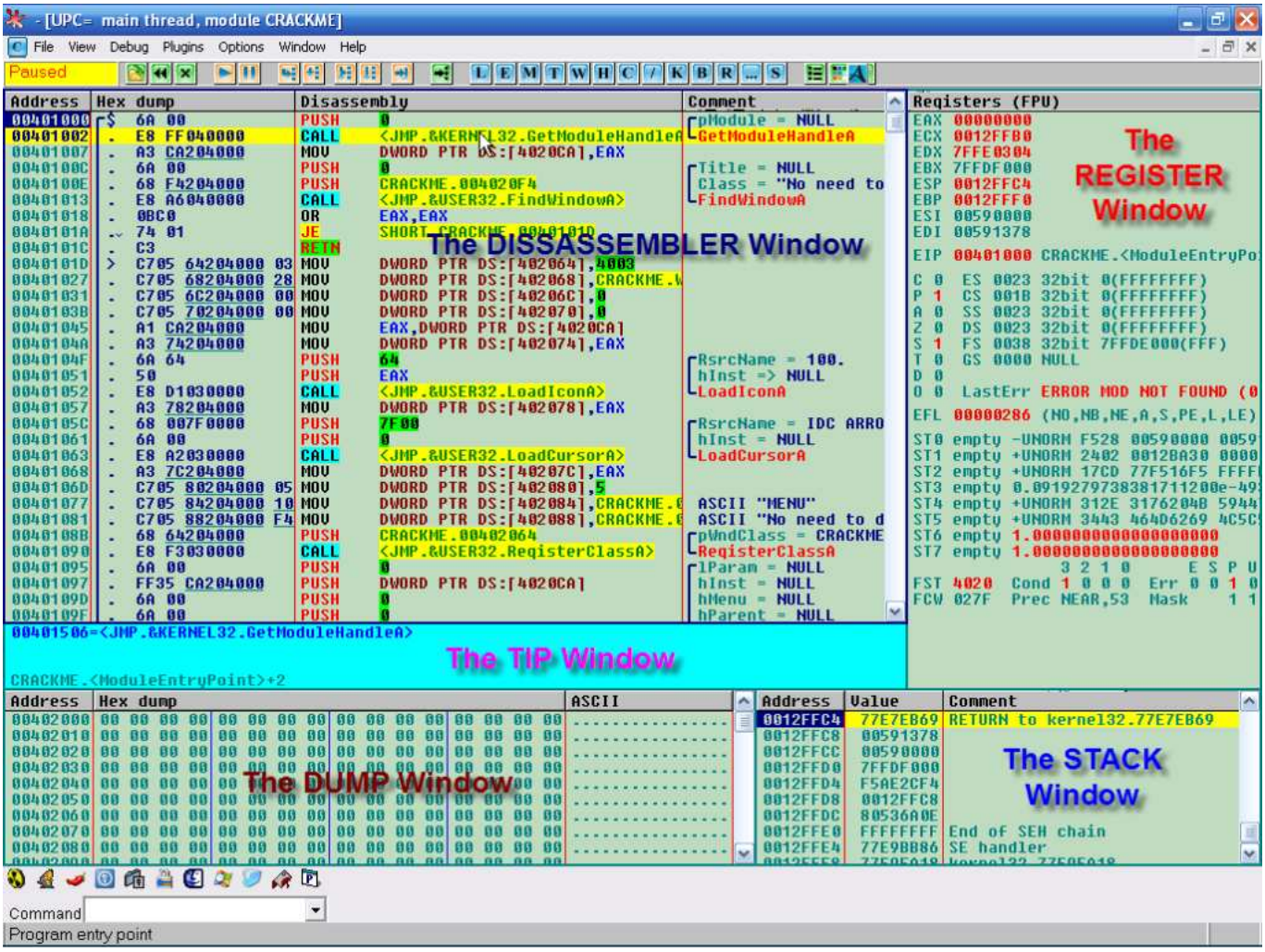
Sau đó chúng ta sẽ chọn chính xác crackme mà chúng ta dùng để minh họa cho bài viết này.



Kết quả sau khi load vào Olly chúng ta có được như sau :



Chắc các bạn nhìn vào sẽ cảm thấy choáng ngợp, không biết phải bắt đầu từ đâu. Hic ngày đầu tiên khi tôi load một target vào trong Olly, nhìn ngược nhìn xuôi cũng không hiểu gì hết luôn hehe, cứ ngồi ngắm mãi vì chẳng biết làm gì hơn. Nhưng không sao mọi thứ đều có cách giải quyết, khi chưa biết thì phải tìm tài liệu mà đọc, khi đọc mà không hiểu lúc đấy hẵng đi hỏi. Nhưng hỏi cũng phải biết đường mà hỏi, nếu không sẽ chẳng bao giờ bạn nhận được câu trả lời mà có khi còn khiến người khác cảm thấy bực mình. Tôi sẽ cùng các bạn tìm hiểu từng cửa sổ một của Olly. Như các bạn nhìn thấy ở trên màn hình chính của Olly được phân ra làm 5 cửa sổ chính, mỗi cửa sổ có một nhiệm vụ và một tên riêng :



Ở đây chúng ta thấy có 4 cửa sổ lớn :

- **The Disassembler Window :** Ở cửa sổ này các bạn có thể nhìn thấy các đoạn code của chương trình ở dạng ngôn ngữ asm, và đồng thời tại cửa sổ này các bạn cũng có thể chú thích cho từng từng dòng mã asm .

- **The Registers Window :** Đây là cửa số chứa thông tin chi tiết về các thanh ghi như eax, ebx, ecx v….v…..Các cờ trạng thái cũng được quản lý tại cửa sổ này

- **The Dump Window :** Tại cửa sổ này bạn có thể xem hoặc chỉnh sửa theo 2 dạng là hex và Ascii bộ nhớ của chương trình mà bạn muốn debug

- **The Stack Window :** Một cửa sổ không kém phần quan trọng , mọi thứ trước khi được thực hiện phải được nạp vào Stack.

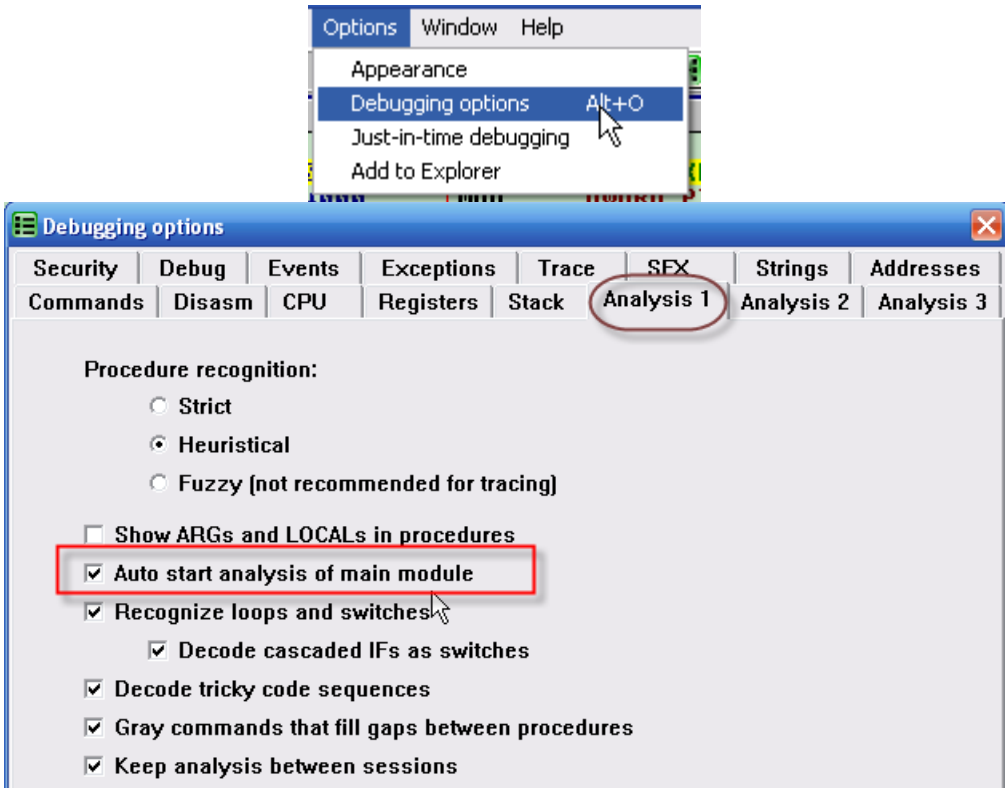
Cuối cùng có một cửa sổ nằm bên dưới cửa sổ **Disassembler Window** : Chúng ta gọi nó là **The Tip Window** . Đây không phải là tên gọi của nó nhưng với tôi, tôi thích gọi như vậy ☺ .Khi bạn đang ở tại một dòng code nào đó trong quá trình debug , **Olly** sẽ cho bạn thấy thông tin chi tiết về dòng code đó . Lấy ví dụ đơn giản như sau : nếu bạn debug tới dòng lệnh *" mov eax , dword ptr [123]"* . Thì cửa sổ này sẽ cho bạn biết được giá trị hay con số nào đang được lưu giữ tại [123] . Và còn nhiều điều thú vị khác nữa mà cửa sổ này sẽ mang lại cho chúng ta .

Trên đây là những gì tổng quan nhất mà các bạn nên biết. Phần dưới đây tôi sẽ đi vào giới thiệu về chức năng của từng cửa sổ một thông qua các hình minh họa, tất nhiên không thể giới thiệu chi tiết hết được, chúng ta sẽ tìm hiểu dần dần trong từng trường hợp cụ thể ở

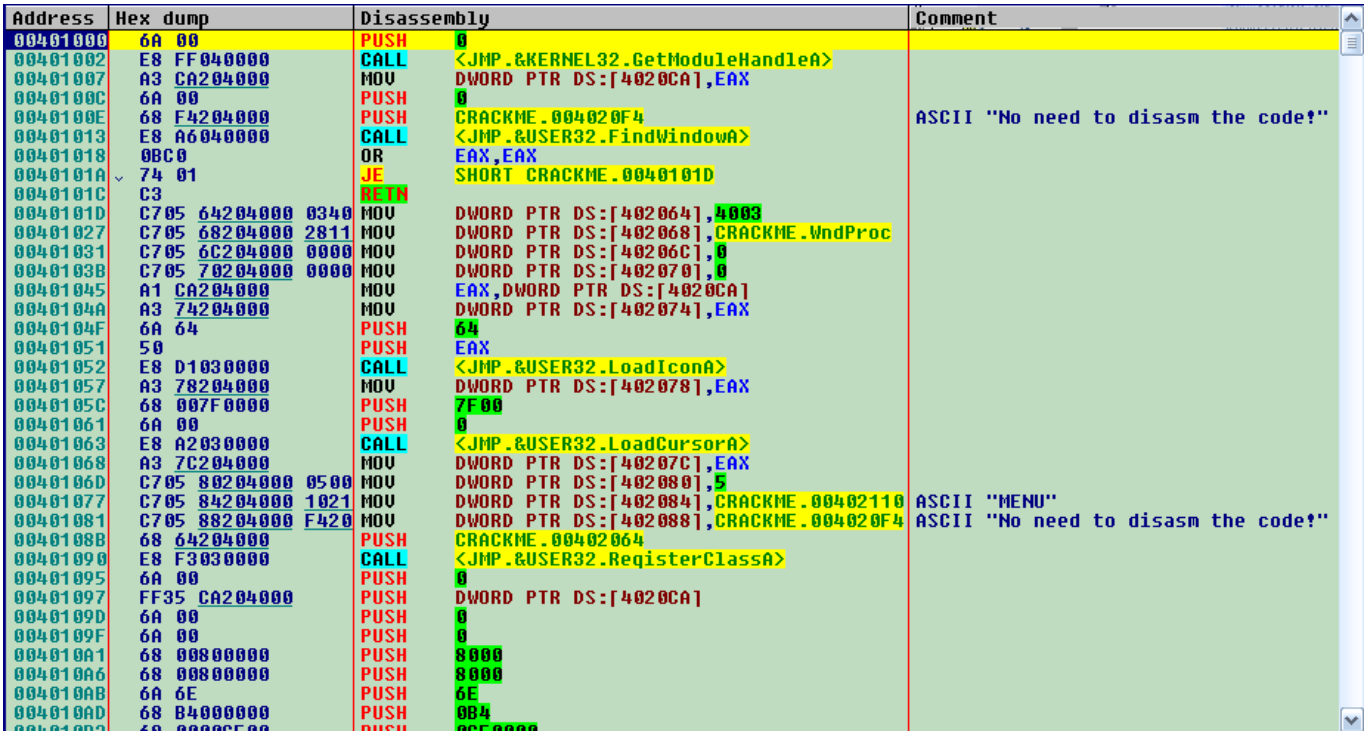
các loạt tuts sau thêm vào đó các bạn cũng nên chủ động tự mình tìm hiểu, đừng nên quá lệ thuộc vào bài viết này.

1. The DISASSEMBLER Window :

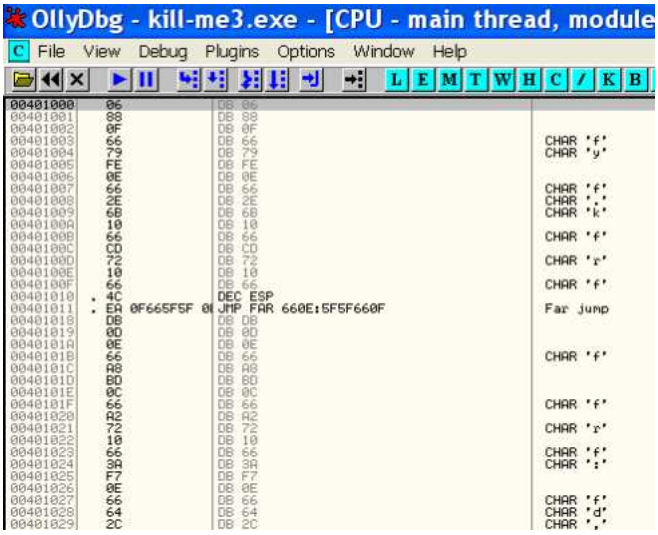
Đây là cửa sổ chính đầu tiên của Olly và là cửa sổ rất quan trọng, chúng ta sẽ làm việc rất nhiều trên cửa sổ này. Khi bạn muốn debug một chương trình, bạn load file thực thi của chương trình đó vào trong Olly.Các chương trình mà bạn load vào Olly là những chương trình có thể được code bằng những ngôn ngữ khác nhau như : **VB, VC++, Borland Delphi hay MASM** nhưng tại cửa sổ này toàn bộ code của chương trình sẽ được list ra dưới dạng các mã ASM. Theo mặc định của Olly thì bất cứ chương trình nào mà bạn load vào Olly sẽ được Olly tiến hành phân tích toàn bộ code chính của chương trình đó và đưa ra các comment thích hợp. Bạn có thể tùy biến chức năng này thông qua hình minh họa dưới đây :



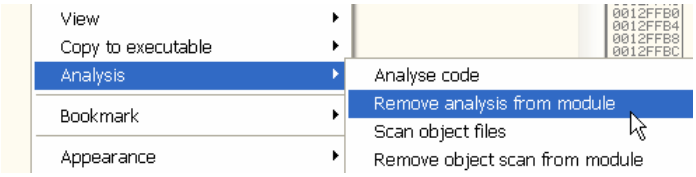
Nếu như bạn chọn sử dụng chức năng này của Olly thì những gì xuất hiện trên cửa sổ bạn sẽ giống với những hình minh họa trước. Còn nếu như bạn không chọn, chúng ta sẽ thấy ngay được sự khác biệt, Olly sẽ không tự động phân tích chương trình nữa công việc phân tích này chúng ta sẽ phải thực hiện một cách manual sau khi chương trình được load vào trong Olly. Okie, tôi thử bỏ chọn và load lại Crackme vào trong Olly, ta sẽ được như sau :



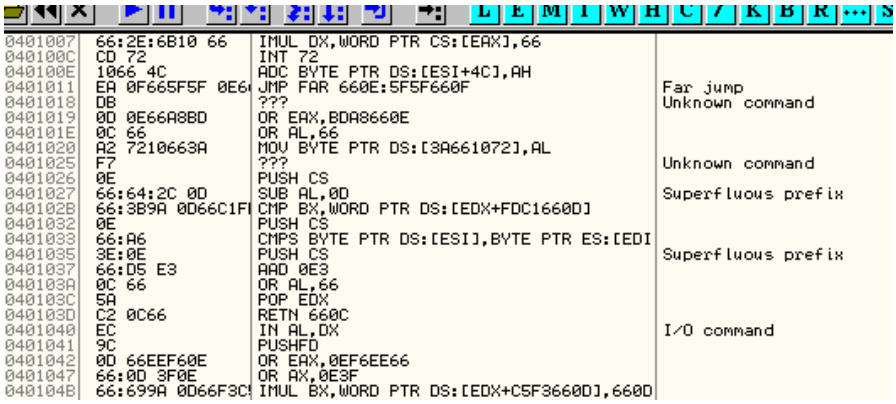
Như các bạn thấy trên hình trên, nếu như chúng ta không chọn chức năng tự động phân tích của Olly thì sẽ thấy các thông tin trong phần **Comment** đã bị lược bỏ đi khá nhiều, điều này dẫn đến việc khó khăn trong quá trình debug chương trình. Tuy nhiên không phải lúc nào chức năng này cũng hoạt động một cách hiệu quả, nhiều khi chúng ta để cho Olly tự động phân tích sẽ lại dẫn đến một kết quả hoàn toàn ngược lại, đoạn code được phân tích và thể hiện ra không được chính xác, ví dụ như trường hợp dưới đây chúng ta sẽ nhận được đoạn code toàn chứa DB :



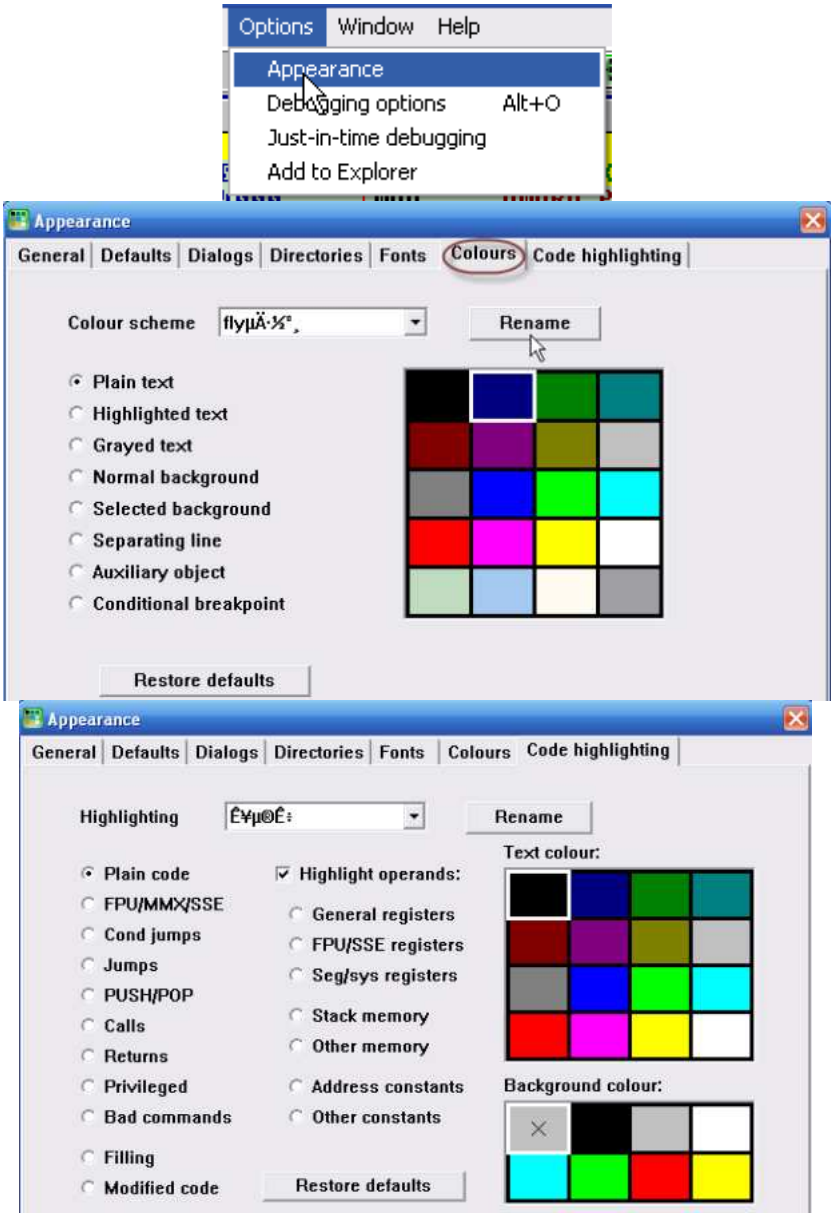
Trong trường hợp như thế này chúng ta có thể thực hiện một cách manual để remove những gì mà Olly đã tiến hành phân tích chỉ đơn giản bằng cách nhấn chuột phải tại màn hình này và chọn **Analysis > Remove analysis from module**



Và kết quả là chúng ta có được đoạn code chính xác như sau :

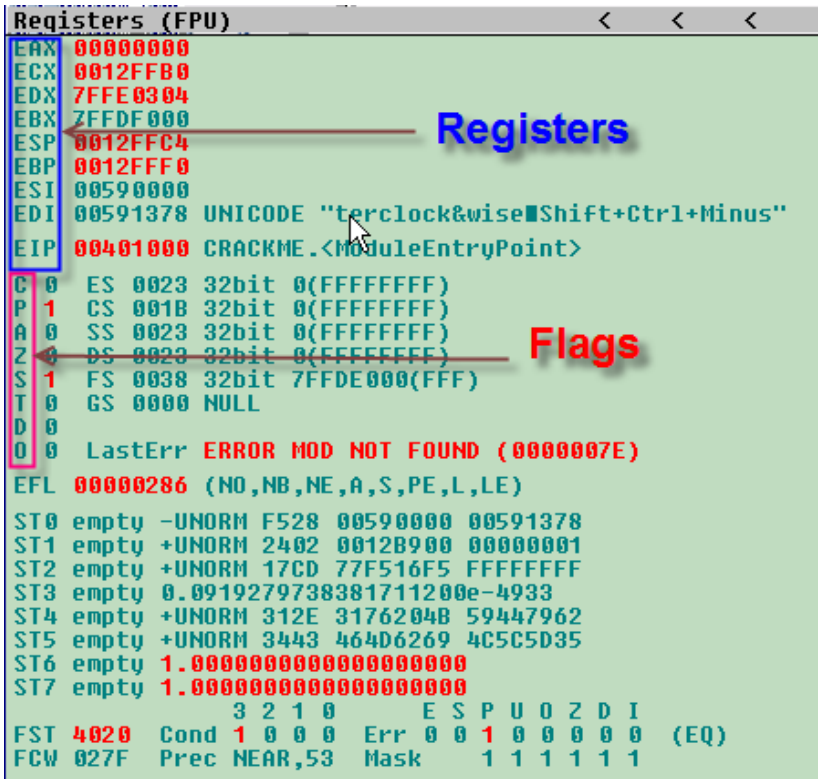


Do đó trong quá trình làm việc với Olly các bạn nên linh hoạt trong quá trình sử dụng chức năng này. Ngoài ra còn một phần khác cũng không kém phần quan trọng, như các bạn thấy trên hình minh họa Olly của tôi các câu lệnh được phân biệt màu sắc một cách rõ ràng, có thể các bạn không chú trọng đến vấn đề này nhưng theo tôi việc chúng ta phân biệt cũng như tinh chỉnh lại màu sắc trong Olly sẽ khiến cho chúng ta nhận biệt các câu lệnh dễ dàng hơn cũng như phần nào thể hiện năng khiếu thẩm mĩ của bạn ☺. Để tinh chỉnh lại màu sắc trong Olly các bạn vào các Tabs sau :



2. The REGISTERs Window :

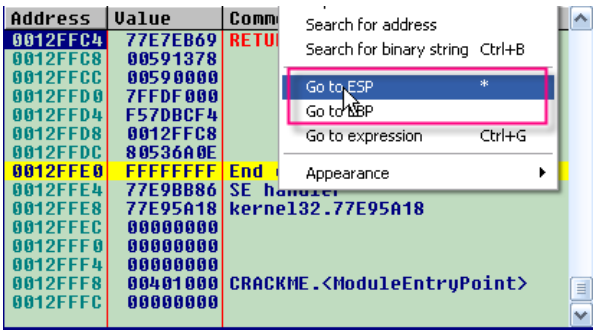
Một cửa sổ quan trọng tiếp theo, đó chính là cửa sổ Register. Như đã nói đây là cửa sổ chứa thông tin chi tiết về các thanh ghi như eax, ebx, ecx v…v… Các cờ trạng thái cũng được quản lý tại cửa sổ này.



Cửa số này sẽ cung cấp cho chúng ta rất nhiều thông tin trong quá trình chúng ta làm việc cùng Olly. Nếu như chỉ nhìn vào hình minh họa ở trên các bạn chắc cũng sẽ như tôi cảm thấy rằng nó sẽ không có ý nghĩa nhiều lắm, nhưng kì thực đây là nơi cung cấp nhiều thông tin rất hữu ích.

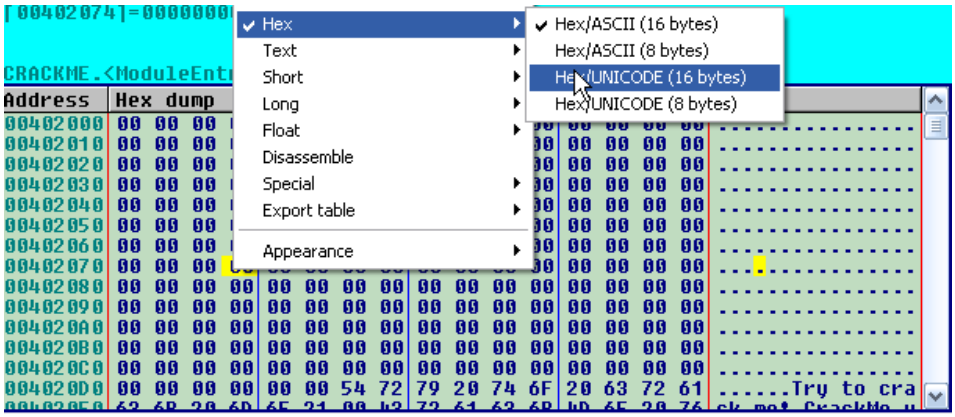
3. The STACK Window :

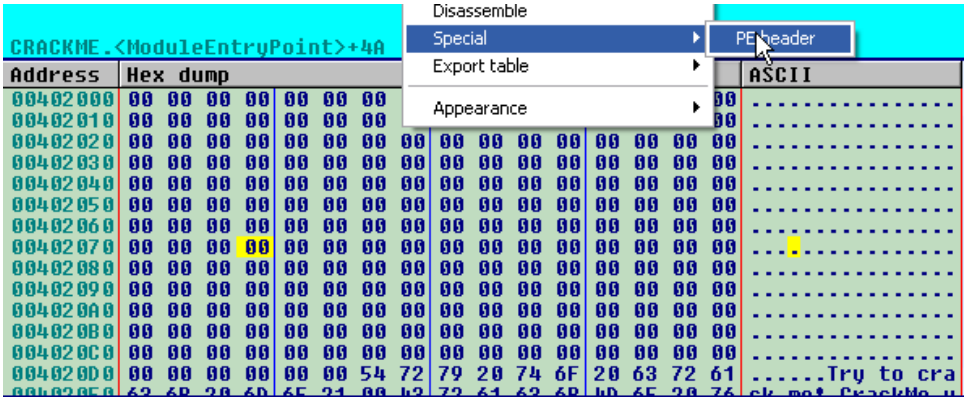
Trước tiên chúng ta sẽ đi tìm hiểu sơ qua về Stack. Đây là nơi lưu trữ tạm thời các dữ liệu và địa chỉ, nó là một cấu trúc dữ liệu một chiều. Các phần tử được cất vào và lấy ra từ một đầu của cấu trúc này, tức là nó được xử lý theo phương thức **"vào trước, ra sau"** (**LIFO : Last In First Out**). Phần tử được cất vào cuối cùng gọi là đỉnh của Stack. Các bạn có thể hình dung Stack như là một chồng đĩa, chiếc đĩa được đặt lên cuối cùng sẽ nằm trên đỉnh và chỉ có nó mới có thể được lấy ra đầu tiên. Hai thanh ghi chính làm việc với Stack là ESP và EBP. Theo mặc định trong Olly, Stack được biểu diễn theo thanh ghi ESP tuy nhiên chúng ta có thể luân chuyển qua lại giữa ESP và EBP bằng cách nhấn chuột phải và chọn như hình sau :



4. The DUMP Window :

Đây là cửa số hiện thị nội dung của bộ nhớ hoặc file. Ta có thể chọn nhiều định dạng khác nhau để biểu diễn nội dung của memory trong cửa số này : byte, text, integer, float, address, disassembly hoặc PE Header. Cửa sổ này cho phép chúng ta tìm kiếm cũng như thực hiện các chức năng chỉnh sửa, thiết lập các Break points v..v...





Vậy là chúng ta đã dạo qua 1 vòng các cửa sổ chính của Olly, tuy nhiên bên cạnh đó Olly còn có rất nhiều cửa sổ khác mà chúng ta không nhìn thấy một cách trực tiếp như các cửa sổ trên được.Chúng ta phải truy cập vào các cửa sổ đó thông qua Menu như hình minh họa dưới đây :

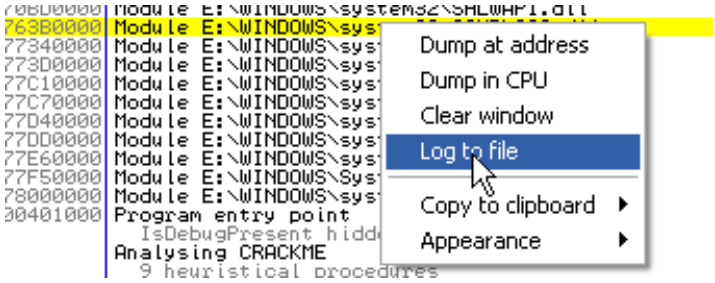
Chúng ta sẽ lướt qua chức năng của từng cửa sổ một.

_ Nút **L** dùng để mở cửa sổ Log của Olly, cửa sổ này cho chúng ta thấy những thông tin mà Olly ghi lại. Theo mặc định thì cửa số này sẽ lưu các thông tin về các module, import library hoặc các Plugins được load cùng chương trình tại thời điểm đầu tiên khi ta load chương trình vào Olly. Bên cạnh đó cửa sổ này cũng ghi lại các thông tin về các Break points mà chúng ta đặt trong chương trình. Trong trường hợp crackme của chúng ta, ta có được thông tin như sau :

```
Olly TBar Manager  Plug-in compiled on Jun  3 2006
anorganix!! thanks alot for your kind help
   Copyright (c) 2006 arjuns,a41arjuns@hotmail.com
Scanning import library 'E:\Documents and Settings\Tran Trung Kien\My Documents\REVERSE TOOLS\ODbyDYK v1.10[12.25]\Lib\mfc71.Lib'
  Resolved 6442 ordinals
Scanning import library 'E:\Documents and Settings\Tran Trung Kien\My Documents\REVERSE TOOLS\ODbyDYK v1.10[12.25]\LibMFC42.Lib'
  Unable to open import library

New process with ID 00000100 created
00401000 Main thread with ID 00000400 created
00400000 Module E:\Documents and Settings\Tran Trung Kien\Desktop\tools\CRACKME.EXE
70BD0000 Module E:\WINDOWS\system32\SHLWAPI.dll
763B0000 Module E:\WINDOWS\system32\COMDLG32.dll
77340000 Module E:\WINDOWS\system32\COMCTL32.DLL
773D0000 Module E:\WINDOWS\system32\SHELL32.dll
77C10000 Module E:\WINDOWS\system32\msvcrt.dll
77C70000 Module E:\WINDOWS\system32\GDI32.dll
77D40000 Module E:\WINDOWS\system32\USER32.dll
77DD0000 Module E:\WINDOWS\system32\ADVAPI32.dll
77E60000 Module E:\WINDOWS\system32\kernel32.dll
77F50000 Module E:\WINDOWS\System32\ntdll.dll
78000000 Module E:\WINDOWS\system32\RPCRT4.dll
00401000 Program entry point
  IsDebugPresent hidden
Analysing CRACKME
  9 heuristical procedures
  27 calls to known functions
  5 loops
```

Một tính năng nữa của cửa sổ này là khi chúng ta muốn lưu lại nhưng thông tin về Log cửa số này cũng cung cấp cho chúng ta khả năng ghi ra file.



_ Nút **E** dùng để mở cửa sổ Executables, cửa sổ này sẽ đưa ra danh sách những file có khả năng thực thi được chương trình sử dụng như file exe, dlls, ocxs , v..v..

Base	Size	Entry	Name	(system)	File version	Path
00400000	00008000	00401000	CRACKME			E:\Documents and Settings\Tran Trung Kien\Desktop\tools\CRACKME.EXE
70BD0000	00065000	70BEEEB3	SHLWAPI	(system)	6.00.2800.1106	E:\WINDOWS\system32\SHLWAPI.dll
71950000	000E4000	7195F225	comctl_1		6.0 (xpclient.010817-1148)	E:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.0.0_x-ww_1382
763B0000	00045000	763B1600	COMDLG32	(system)	6.00.2600.0000 (xpclient.010817	E:\WINDOWS\system32\COMDLG32.dll
77340000	0008B000	77341A59	COMCTL32	(system)	5.82 (xpclient.010817-1148)	E:\WINDOWS\system32\COMCTL32.DLL
773D0000	007F4000	773F2014	SHELL32	(system)	6.00.2600.0000 (xpclient.010817	E:\WINDOWS\system32\SHELL32.dll
77C10000	00053000	77C1E94F	msvcrt	(system)	7.0.2600.0 (xpclient.010817-114	E:\WINDOWS\system32\msvcrt.dll
77C70000	00040000		GDI32	(system)	5.1.2600.0 (xpclient.010817-114	E:\WINDOWS\system32\GDI32.dll
77D40000	0008D000	77D4514B	USER32	(system)	5.1.2600.0 (xpclient.010817-114	E:\WINDOWS\system32\USER32.dll
77DD0000	0008B000	77DD1CFB	ADVAPI32	(system)	5.1.2600.0 (XPClient.010817-114	E:\WINDOWS\system32\ADVAPI32.dll
77E60000	000E5000	77E7A241	kernel32	(system)	5.1.2600.0 (xpclient.010817-114	E:\WINDOWS\system32\kernel32.dll
77F50000	000A9000		ntdll	(system)	5.1.2600.0 (xpclient.010817-114	E:\WINDOWS\System32\ntdll.dll
78000000	0006E000	780072BA	RPCRT4	(system)	5.1.2600.109 (xpclnt_qfe.021108	E:\WINDOWS\system32\RPCRT4.dll

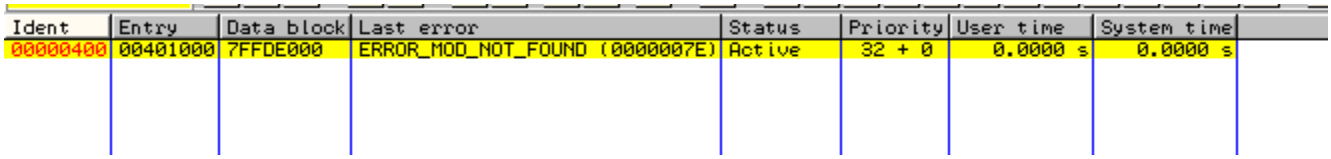
Tại cửa sổ này nếu như bạn click chuột phải sẽ thấy có rất nhiều tùy chọn khác nhau, trong khuôn khổ có hạn của bài viết không thể nói hết được. Sẽ có những phần tiếp theo đề cập đến chúng.

_ Nút **M** dùng để mở cửa sổ Memory, cửa sổ này sẽ cho chúng ta thông tin về bộ nhớ đang được sử dụng bởi chương trình của chúng ta và còn nhiều thông tin bổ ích khác nữa :

003F0000	00001000				Priv	RW	RW	
00400000	00001000	CRACKME		PE header	Imag	R	RWE	
00401000	00001000	CRACKME	CODE	code	Imag	R	RWE	
00402000	00001000	CRACKME	DATA	code,data	Imag	R	RWE	
00403000	00001000	CRACKME	.idata	code,import	Imag	R	RWE	
00404000	00001000	CRACKME	.edata	code,export	Imag	R	RWE	
00405000	00001000	CRACKME	.reloc	code,reloca	Imag	R	RWE	
00406000	00002000	CRACKME	.rsrc	code,resour	Imag	R	RWE	
00410000	00103000				Map	R	R	
00520000	00060000				Map	R E	R E	
00820000	00001000				Priv	RW	RW	
00830000	00004000				Priv	RW	RW	
00840000	00003000				Map	R	R	\Device\HarddiskVolume3\WINDOWS\System32\CTYPE.NLS
00860000	00002000				Map	R	R	
01040000	00002000				Map	R	R	
70BD0000	00001000	SHLWAPI		PE header	Imag	R	RWE	
70BD1000	0005A000	SHLWAPI	.text	code,import	Imag	R	RWE	
70C2B000	00002000	SHLWAPI	.data	code,data	Imag	R	RWE	
70C2D000	00002000	SHLWAPI	.rsrc	code,resour	Imag	R	RWE	
70C2F000	00006000	SHLWAPI	.reloc	code,reloca	Imag	R	RWE	

Tại cửa sổ này chúng ta cũng có thể sử dụng tính năng Search để tìm kiếm thông tin về các strings, các đoạn hexa cụ thể hay unicode v..v.. thêm vào đó nó còn cung cấp cho chúng ta những kiểu thiết lập Break points khác nhau tại các Sections. Việc thiết lập các BPs là tùy thuộc vào yêu cầu và mục đích của chúng ta.

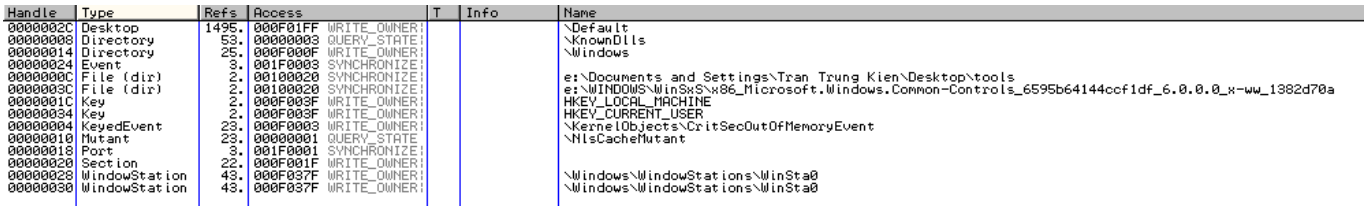
_ Nút **T** dùng để mở cửa sổ Threads, cửa sổ này liệt kê các Threads của chương trình :



Ident	Entry	Data block	Last error	Status	Priority	User time	System time
00000400	00401000	7FFDE000	ERROR_MOD_NOT_FOUND (0000007E)	Active	32 + 0	0.0000 s	0.0000 s

_ Nút **W** dùng để mở cửa sổ Windows

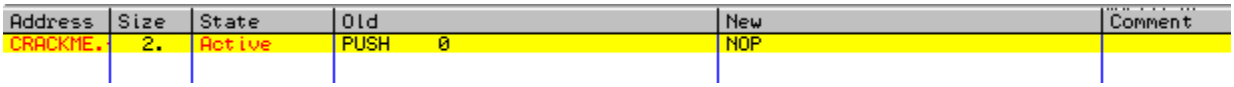
_ Nút **H** dùng để mở cửa sổ Handles



Handle	Type	Refs	Access	T	Info	Name
0000002C	Desktop	1495.	000F01FF WRITE_OWNER			\Default
00000008	Directory	53.	00000003 QUERY_STATE			\KnownDlls
00000014	Directory	25.	000F000F WRITE_OWNER			\Windows
00000024	Event	3.	001F0003 SYNCHRONIZE			
0000000C	File (dir)	2.	00100020 SYNCHRONIZE			e:\Documents and Settings\Tran Trung Kien\Desktop\tools
0000003C	File (dir)	2.	00100020 SYNCHRONIZE			e:\WINDOWS\WinSxS\x86_Microsoft.Windows.Common-Controls_6595b64144ccf1df_6.0.0.0_x-ww_1382d70a
0000001C	Key	2.	000F003F WRITE_OWNER			HKEY_LOCAL_MACHINE
00000034	Key	2.	000F003F WRITE_OWNER			HKEY_CURRENT_USER
00000004	KeyedEvent	23.	000F0003 WRITE_OWNER			\KernelObjects\CritSecOutOfMemoryEvent
00000010	Mutant	23.	00000001 QUERY_STATE			\NlsCacheMutant
00000018	Port	3.	001F0001 SYNCHRONIZE			
00000020	Section	22.	000F001F WRITE_OWNER			
00000028	WindowStation	43.	000F037F WRITE_OWNER			\Windows\WindowStations\WinSta0
00000030	WindowStation	43.	000F037F WRITE_OWNER			\Windows\WindowStations\WinSta0

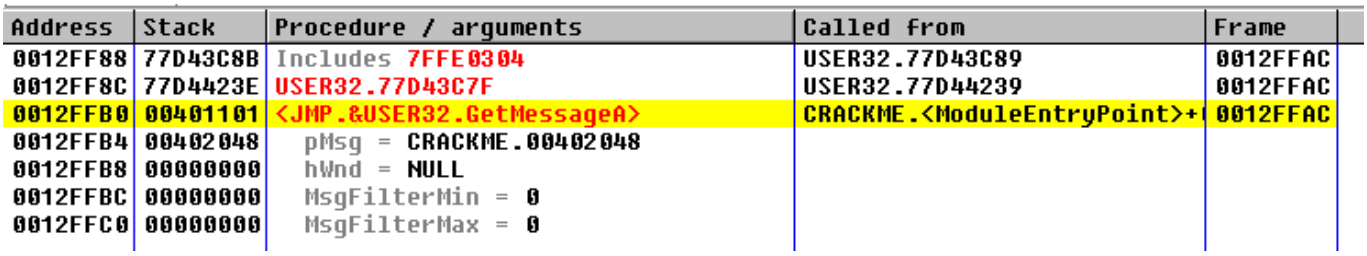
_ Nút **C** thì khỏi nói , bạn cứ nhấn vào là khắc biết ngay ☺

_ Nút **/** để mở cửa sổ Patches, cửa sổ này sẽ cho chúng ta các thông tin về những gì mà chúng ta đã edit trong chương trình.



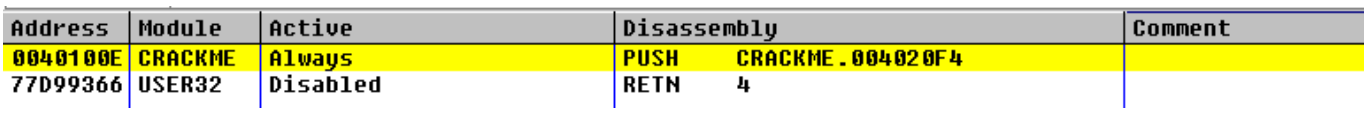
Address	Size	State	Old	New	Comment
CRACKME.	2.	Active	PUSH 0	NOP	

_Nút **K** để mở cửa sổ Call Stack, hiển thị một danh sách các lệnh call mà chương trình của chúng ta đã thực hiện khi chúng ta Run bằng F9 và dùng F12 để tạm dừng chương trình.



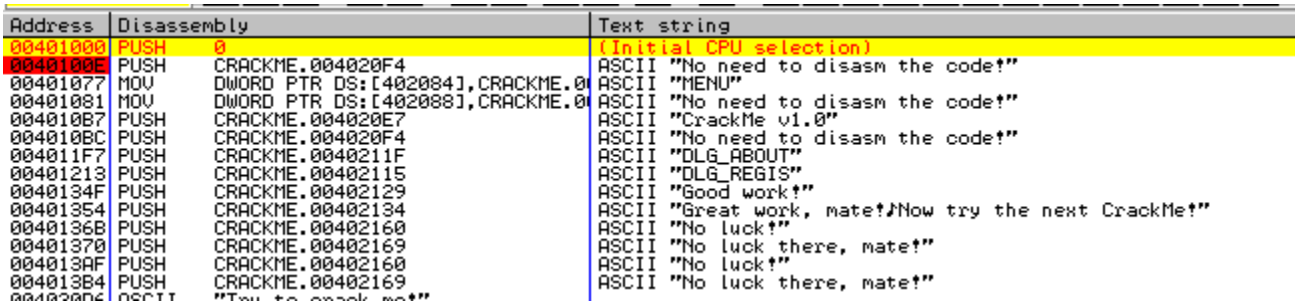
Address	Stack	Procedure / arguments	Called from	Frame
0012FF88	77D43C8B	Includes 7FFE0304	USER32.77D43C89	0012FFAC
0012FF8C	77D4423E	USER32.77D43C7F	USER32.77D44239	0012FFAC
0012FFB0	00401101	<JMP.&USER32.GetMessageA>	CRACKME.<ModuleEntryPoint>+	0012FFAC
0012FFB4	00402048	pMsg = CRACKME.00402048		
0012FFB8	00000000	hWnd = NULL		
0012FFBC	00000000	MsgFilterMin = 0		
0012FFC0	00000000	MsgFilterMax = 0		

_ Nút **B** để mở cửa sổ Break Points, cửa sổ này sẽ hiển thị tất cả các BPs mà chúng ta đặt trong chương trình. Tuy nhiên nó chỉ hiện thị các BPs được set bằng cách nhấn F2 thôi, còn các dạng BPs khác như : hardware breakpoint hoặc memory breakpoints thì không được liệt kê ra ở đây:



Address	Module	Active	Disassembly	Comment
0040100E	CRACKME	Always	PUSH CRACKME.004020F4	
77D99366	USER32	Disabled	RETN 4	

_ Nút **R** để mở cửa sổ References, cửa sổ này là kết quả cho những gì chúng ta thực hiện chức năng Search trong Olly, kết quả sẽ được hiện ra ở đây :



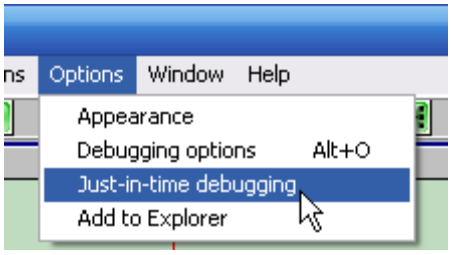
Phù khá nhiều cửa sổ phải không các bạn, tôi sẽ không đi vào chi tiết thêm nữa bởi vì chúng ta sẽ còn gặp lại trong các tuts tiếp theo, 1 yêu cầu rất quan trọng ngoài việc bạn biết sử dụng Olly ra thì bạn còn phải biết về Asm language, nếu không biết về nó thì hii các bạn nên dành thời gian để tìm hiểu một số kiến thức cơ bản trước khi đọc tiếp các phần sau của bài viết. Ngoài ra để các bạn dễ làm quen hơn trong các phần sau tôi sẽ cố gắng hệ thống lại ☺.

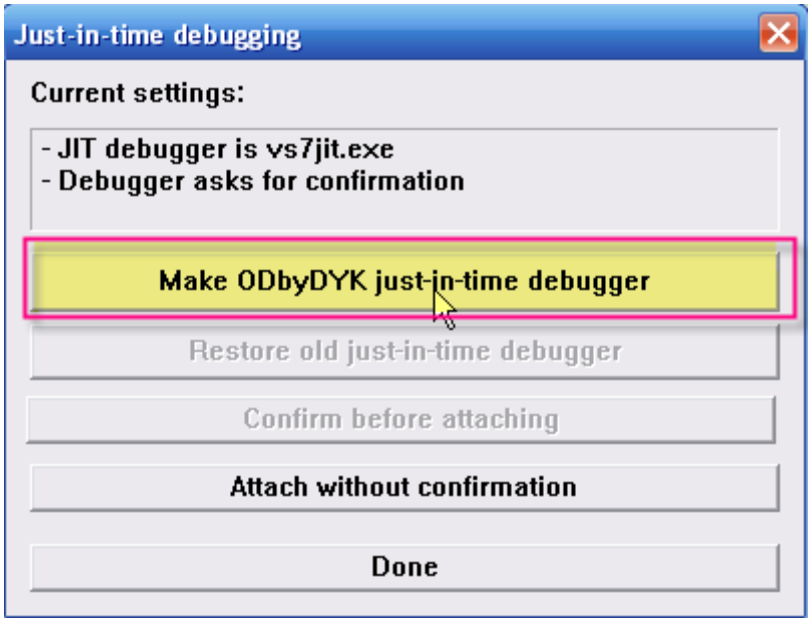
IV. Cấu hình Olly thành JIT (Just-in-time debugging)

Khi một số chương trình thực thi và nó tạo ra Exception, Windows có thể gọi Registered Debugger (các debuggers được cấu hình thành JIT) và attach nó vào chương trình. Tính năng này được gọi là Just-in-time debugging.

Một vài JIT debuggers dừng lại tại System breakpoint. Ollydbg thì tiếp tục thực thi cho đến khi nó đi đến câu lệnh đã tạo ra Exception.

Để cấu hình Ollydbg trở thành 1 JIT bạn làm như sau :





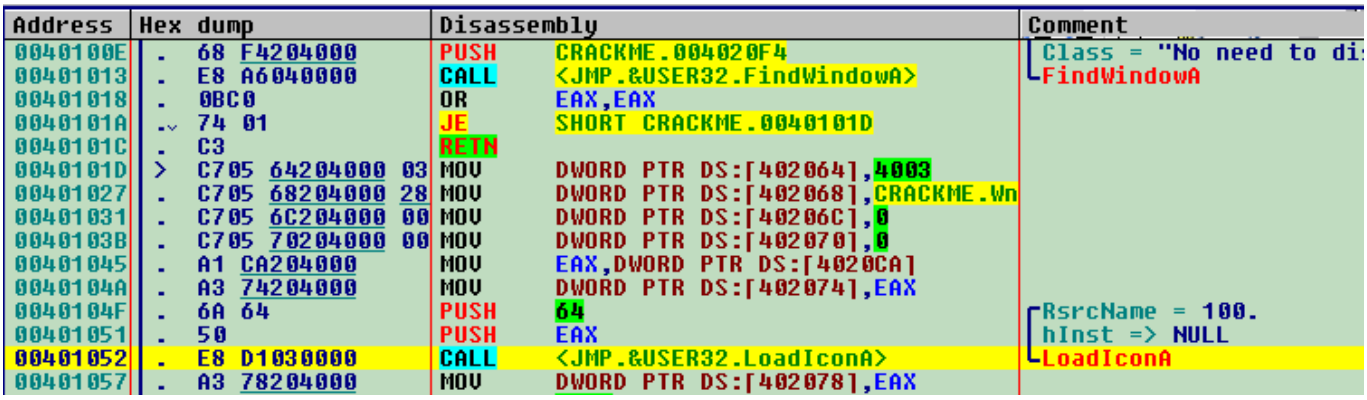
Nếu như bạn không muốn sử dụng tính năng này thì bạn có thể Restore lại.

V. Một số phím cơ bản để làm việc với Olly :

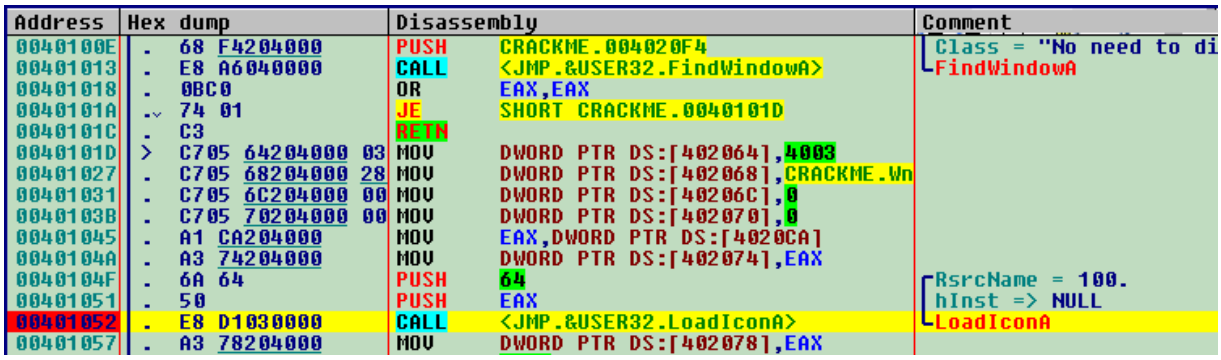
F7 : Khi bạn nhấn F7 sẽ thực thi từng dòng lệnh 1. Nếu trong quá trình Trace mà gặp lệnh Call thì sẽ đi vào trong lòng của lệnh Call đó và thực thi từng câu lệnh trong lệnh Call này cho đến khi gặp lệnh Retn để trở lại chương trình chính, tức là câu lệnh tiếp theo sau lệnh Call.

F8 : Cũng tương tự như F7 nhưng có 1 điểm khác biệt là khi Trace code, nếu như gặp lệnh Call nó bỏ qua không cần quan tâm các lệnh bên trong lệnh Call mà thực thi luôn lệnh Call đó và dừng lại tại câu lệnh tiếp theo dưới lệnh Call.

F2 : Đặt một Break point trong chương trình. Vậy Break point là gì , đơn giản nó chỉ là việc chúng ta tạo 1 điểm ngắt trong chương trình theo một điều kiện nào đó để khi thực thi chương trình, nếu thỏa điều kiện mà chúng ta đặt ra thì chương trình sẽ dừng lại tại vị trí mà chúng ta đã đặt BP. Ví dụ, trong hình minh họa dưới đây :

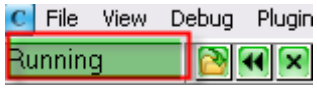


Bây giờ tôi muốn đặt một BP tại hàm Call gọi tới API: **LoadIconA**. Tức là khi tôi thực thi chương trình, chương trình gọi tới hàm này thì ngay lập tức nó sẽ dừng lại tại đây.Việc tiếp theo là tôi có thể tùy biến lại hàm này theo mục đích của tôi, chẳng hạn tôi NOP nó để chương trình không còn gọi đến hàm này nữa v..v.. Để làm được điều này bạn nhấn chuột tại vị trí cần Set BP, sau đó nhấn F2. Chỗ chúng ta Set BP sẽ được đánh dấu màu đỏ :



Để bỏ BP mà chúng ta đã set thì chỉ việc chọn vị trí đánh dấu màu đỏ và nhấn F2.

F9 : Cho phép thực thi chương trình trong chế độ Debug, tương tự như việc chúng ta nhấp đúp chuột vào chương trình để thực thi nó. Tuy nhiên khác với việc nhấp đúp chuột, nếu chúng ta nhấn F9 thì Olly sẽ tìm xem có BP nào được Set hay không, chương trình có tung ra các Exception gì không, hay nếu chương trình có cơ chế chống Debug thì nó sẽ terminate ngay lập tức. Nếu như không có bất kì cản trở nào thì chương trình sẽ Run hoàn toàn và trên status bar của Olly sẽ báo cho chúng ta biết điều này :



F12 : Tạm dừng chương trình lại.

VI. Lời kết :

Trên đây là những gì tổng quan nhất về Olly, như đã nói các bạn không nên quá lệ thuộc vào bài viết này của tôi, các bạn có thể tự mình tìm hiểu thêm những tính năng khác của Olly. Các phần sau của loạt tuts này làm việc trên Crackme của tác giả CRUEHEAD, để tiện cho các bạn đỡ mất công tìm kiếm tôi đã kèm luôn target cùng với bài viết này. Hi vọng những gi tôi đã viết ở trên đã giúp cho các bạn phần nào hiểu được tại sao Ollydbg đang ngày càng trở nên phổ biến.

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, dqtln, ARTEAM .... all my friend, and YOU.

--++--==[Thanks To]==--++--
iamidiot, WhyNotBar, trickyboy, dzungltvn, takada, hurt_heart, haule_nth, hytkl 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). And finally, thanks to **RICARDO NARVAJA** and all members on **CRACKSLATINOS.**

>>>> If you have any suggestions, comments or corrections email me:
kienbigmummy[at]gmail.com