# INTRODUCTION TO THE CRACKING WITH OLLYDBG

# FROM CRACKLATINOS

**(_kienmanowar_)**

## I. Lời nói đầu

Chào mọi người, sau tut đầu tiên của tôi giới thiệu tới các bạn về Ollydbg, bẵng đi một thời gian do công việc bận rộn tôi đành gác bút chưa thể viết tiếp được. Bây giờ mọi việc có vẻ ổn định rồi, tôi sẽ dành chút thời gian để tiếp tục bộ tut này. Mặc dù có bạn đã làm tiếp công việc của tôi là dịch và viết đến tut thứ 16, nhưng tôi sẽ vẫn viết lại theo cách viết và phong cách của tôi. Đây vừa là những bài viết mà tôi chia sẻ đến các bạn cũng đồng thời là việc tôi đúc kết và lưu trữ những gì mình đã làm được. Ở phần trước sau khi các bạn đã có một cái nhìn tổng quan nhất về công cụ Ollydbg về các thành phần cũng như chức năng chính của nó, thì trong phần thứ hai này tôi sẽ đề cập đến việc sử dụng các hệ thống số trong Olly, thêm vào đó là một chút kiến thức cơ bản về Stack. Okie, L3t's G0!!

## II. Các hệ thống số

Có ba hệ thống số được sử dụng nhiều nhất đó là Hệ nhị phân, Hệ mười và cuối cùng là hệ thập lục phân.Chúng ta sẽ đi lần lượt định nghĩa về từng hệ thống này.

**Hệ nhị phân :** Trong hệ đếm nhị phân cơ số là 2 và nó chỉ có hai chữ số là 0 và 1.

**Hệ mười (thập phân) :** Có thể nói đây là một hệ thống được chúng ta sử dụng nhiều nhất trong đời sống hàng ngày.Hệ này bao gồm mười chữ số bắt đầu từ 0 đến 9. Hệ đếm này là hệ đếm mà chúng ta quen thuộc nhất.

**Hệ mười sáu :** Các số dưới dạng nhị phân thường là dài và khó nhớ. Việc chuyển đổi các số thập phân sang nhị phân thường khó. Khi chúng ta viết chương trình hợp ngữ chúng ta thường sử dụng cả hai hệ đếm là : nhị phân và thập phân, và cả một hệ đếm thứ ba là hệ 16 hay còn gọi tắt là số hex. Số hex cho phép chúng ta chuyển đổi một cách dễ dàng sang số ở hệ nhị phân và ngược lại.

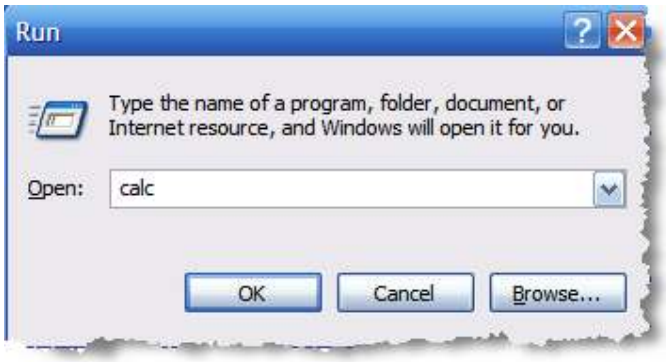
*Note : Để đổi số hex sang số nhị phân chúng ta chỉ việc biểu diễn các chữ số của nó dưới dạng nhị phân. Còn đổi số nhị phân sang số hex, thì ta nhóm 4 chữ số của số nhị phân lại theo thứ tự lần lượt từ phải qua trái. Sau đó chuyển thành số hex tương ứng.*

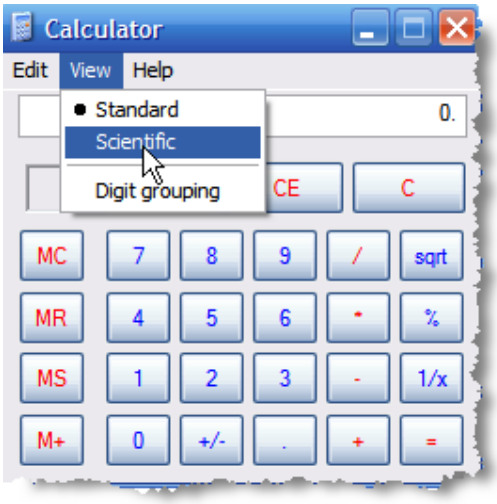
Hệ đếm hex là hệ đếm có cơ số 16 cho nên các chữ số của nó là : 0-9, A-F. (Vì hết các kí hiệu chữ số để biểu diễn nên người ta dùng thêm các chữ cái để biểu diễn: các chữ cái từ A – F tương ứng biểu diễn các số từ 10 – 15).

Khi bạn muốn làm quen với công việc debug trong Olly thì điều đầu tiên tôi khuyên bạn nên làm quen với các hệ thống số ở trên, Olly chủ yếu sử dụng hệ 16. Bên cạnh đó các bạn cũng phải học các phương pháp chuyển đổi đơn giản giữa các hệ số với nhau để tiện cho quá trình bạn làm việc. Có thể các bạn sẽ cho lời tôi nói là thừa bởi vì ngày nay có quá

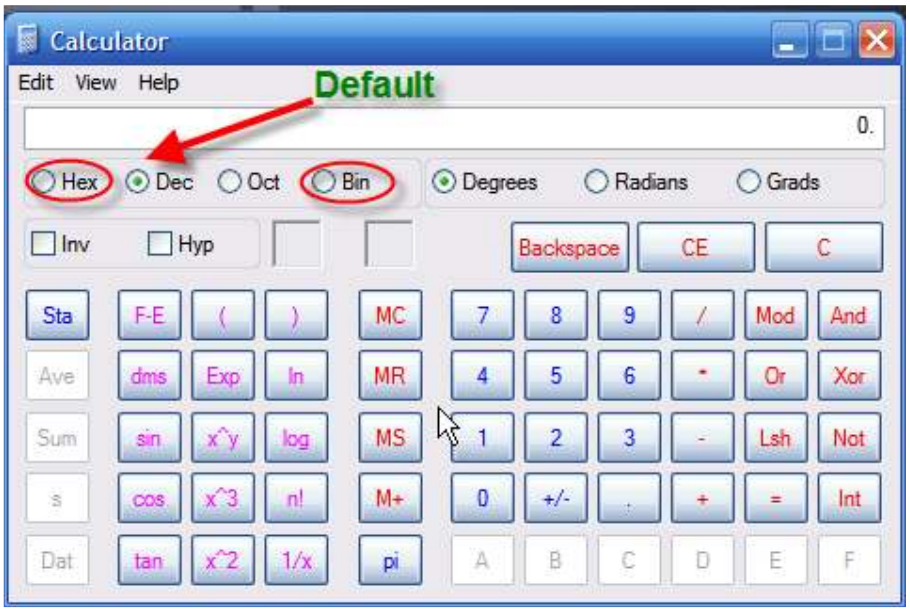
nhiều công cụ hỗ trợ cho chúng ta làm việc này, nhưng theo tôi đây vẫn là những kiến thức tiên quyết vì công cụ chỉ là hỗ trợ để chúng ta làm việc nhanh chóng mà thôi, còn muốn hiểu sâu, rộng thì chúng ta không nên bỏ qua những chi tiết dù là vụn vặt nhất.

Ở đây trong bài viết này, tôi coi như các bạn đã tự mình trang bị những kiến thức cơ bản rồi. Do đó để dễ dàng hơn cho chúng ta khi làm việc với các hệ thống số, Windows cung cấp cho chúng ta một công cụ khá mạnh mà đôi khi ít người để ý mà thậm chí có khi còn không biết là nó hỗ trợ cho chúng ta các tính năng liên quan đến việc chuyển đổi ☺, đó chính là tiện ích **Calculator**. Có nhiều cách thức để mở chương trình này nhưng cách nhanh nhất là vào **menu Run** và gõ **Calc.exe** (thậm chí chỉ cần gõ Calc cũng mở được).

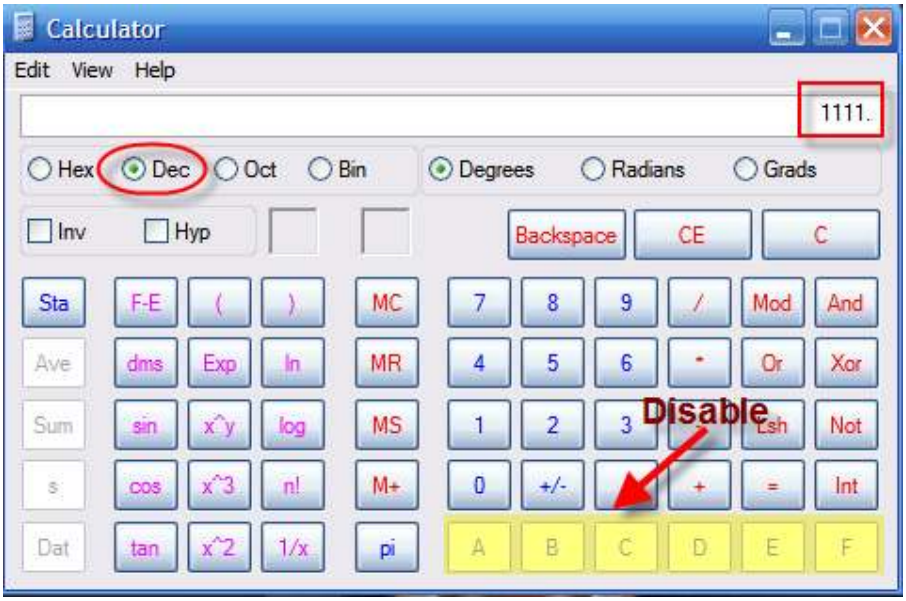


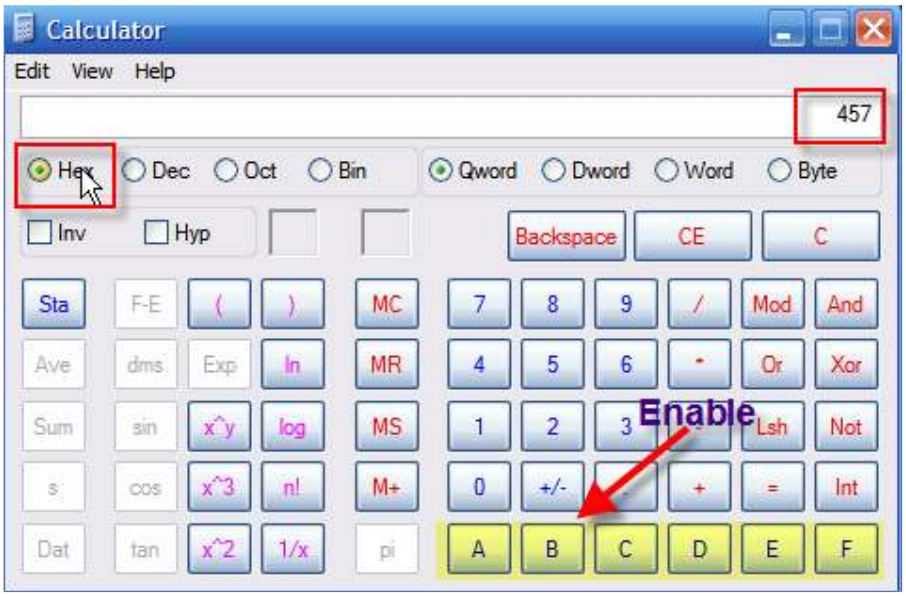

Như bạn thấy trên hình sau khi chúng ta gõ **Calc** thì ngay lập tức công cụ Calculator sẽ hiện ra dưới dạng một máy tính chuẩn hệt như cái máy tính bình thường mà bạn hay sử dụng. Để có thể chuyển sang sử dụng các tính năng chuyên nghiệp hơn liên quan tới các số hệ nhị phân và hệ 16 cũng như các phép tính liên quan tới hai hệ số này, bạn làm như trên hình vẽ (**View > Scientific**). Ta có được như sau :




Trong hình minh họa bên trên, bạn thấy hệ thống số được sử dụng mặc định là hệ 10 (Dec).Tại sao nó lại mặc định như vậy? Một câu trả lời rất đơn giản là vì từ lúc cha sinh mẹ đẻ chúng ta tới giờ chúng ta sử dụng hệ 10, hệ đếm chuẩn của loài người ☺ nên chương trình để default như vậy là hoàn toàn hợp lý. Các bạn có thể luân chuyển sang các hệ khác rất đơn giản thông qua các tùy chọn. Lấy một ví dụ, tôi muốn chuyển một con số từ hệ 10 sang hệ 16 thì tôi làm thế nào? Tại màn hình **Calculator** bạn chọn **Dec** và gõ vào một con số bất kì, ví dụ : **1111**)




Để chuyển sang hệ Hex bạn chỉ việc nhấp chọn vào tùy chọn Hex tại cửa màn hình của Calculator, ngay lập tức số ở hệ 10 của bạn sẽ được chuyển sang số ở hệ 16 một cách chính xác.




Trên hình trên bạn đã thấy khi ở hệ 10 thì các chữ cái từ A – F đều bị disable. Khị bạn chọn chuyển sang hệ hex thì các chữ cái này sẽ được enable lên để phục vụ cho các bạn làm việc ở hệ hex. Việc chuyển đôi qua lại các hệ số khác cũng làm tương tự như trên, qua đó bạn thấy công cụ này đã đơn giản hóa cho chúng ta rất nhiều các công việc liên quan đến việc chuyển đổi bằng tay.Tất cả những gì bạn phải làm là gõ số và nhấn chọn khà khà ☺.

## III. Số có dấu trong hệ 16

Phần cứng của máy tính cần giới hạn kích thước của các số để có thể lưu nó trong các thanh ghi hay các ô nhớ. Trong hệ hex vấn đề sẽ nảy sinh khi chúng ta muốn biểu diễn một số âm ví dụ như -1 chẳng hạn, chúng ta không thể làm bằng cách thêm một dấu trừ ở phía trước con số giống như ở trong hệ 10 được.Vì nếu làm thế thì đơn giản quá rồi, đâu cần phải đề cập đến vấn đề này làm gì và vì hệ thống máy tính mà chúng ta đang sử dụng chỉ làm việc với hai số 0 và 1 mà thôi, cho nên để biểu diễn một số có dấu phải có qui định khác.

Do chúng ta đang làm việc với hệ thống 32 bít cho nên dải số của nó sẽ được biểu diễn ở hệ hex là từ 00000000 – FFFFFFFF.Dải này sẽ được cắt nửa ra, một nửa dùng để biểu diễn số dương và một nửa dùng để biểu diễn số âm. Vậy số dương sẽ bắt đầu từ 00000000 và kết thúc là 7FFFFFFF, còn số âm sẽ bắt đầu từ 80000000 và kết thúc là FFFFFFFF. Vậy làm thế nào để nhận biết đâu là số âm và đâu là số dương? Các bạn hãy để ý đến một bit đặc biệt, đó là bit nằm ở tận cùng bên trái hay còn được gọi với một cái tên khác là bit có trọng số nặng nhất (**MSB - Most significant bit**). Tương tự như vậy ta cũng có một bit có trọng số thấp nhất hay còn gọi là bít nhẹ nhất đó là số nằm ở tận cùng bên phải (**LSB – Least Significant bit**).

Nếu như bit có trọng số cao nhất là 0 thì số đó được hiểu là số dương. Còn nếu như bít có có trọng số cao nhất là 1 thì số được được hiểu là số âm. Bằng 0 hay bằng 1 là khi chúng ta biểu diễn số đó dưới dạng nhị phân. Các số âm trong máy tính được lưu ở dạng số bù 2 (Note: số bù 2 có được bằng cách đảo bít của một số nguyên và cộng với 1).

Theo đó ta có được dải biểu diễn như sau :

**<u>SỐ DƯƠNG</u>** :

00000000h hệ 16 – 0 hệ 10
00000001h hệ 16 – 1 hệ 10

…….
7FFFFFFFh hệ 16 – 2147483647 hệ 10 (Số dương lớn nhất)

**SỐ ÂM :**

FFFFFFFFh hệ 16 - -1 hệ 10
FFFFFFFEh hệ 16 - -2 hệ 10
…….
80000000h hệ 16 - -2147483647 hệ 10 (Số âm nhỏ nhất)

Tôi sẽ làm một ví dụ chuyển đổi sang số bù 2 để các bạn thấy được một cách trực quan nhất. Giả sử tôi có số dương là 1 , giờ tôi muốn biểu diễn số -1 tôi sẽ làm thế nào. Để đơn giản tôi chỉ làm mẫu với số 16 bit.

**_ Đầu tiên ta tìm số bù 1 của 1 (có được bằng cách đảo bít) :**

1. Biểu diễn 1 ở dạng nhị phân : 0000 0000 0000 0001
2. Tìm số bù 1 của 1 : 1111 1111 1111 1110

**_ Tìm số bù 2 của 1 bằng cách lấy bù 1 đem cộng với 1 :**

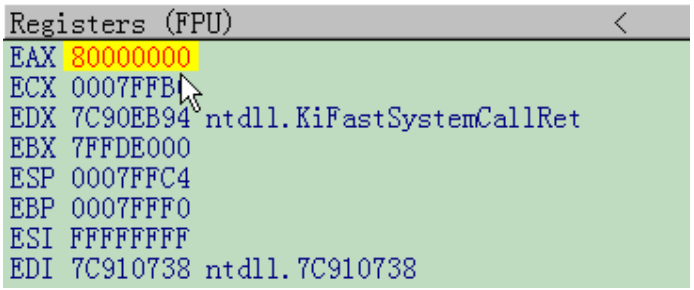
1. Theo kết quả ở trên, bù 1 của 1 : 1111 1111 1111 1110
2. Cộng với 1 : +1
3. Kết quả là số bù 2 : 1111 1111 1111 1111

Đem số bù hai này chuyển qua hệ Hex các bạn sé có được là : FFFFh

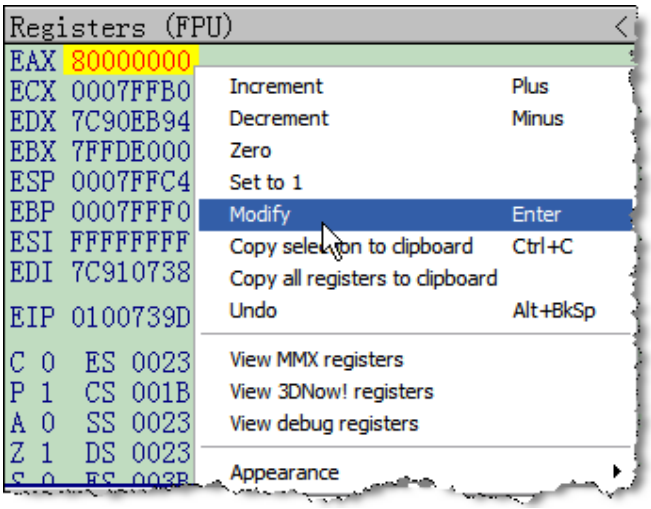
Trong Olly chúng ta có thể giải quyết mọi vấn đề liên quan thông qua Plug-in : **Command Bar**. Để sử dụng nó cũng rất đơn giản, bạn làm như hình minh họa dưới đây :


Rất trực quan và dễ hiểu, bạn không biết giá trị ở hệ 10 của 7FFFFFFFh là bao nhiêu. Trong Plug-in **Command Bar** bạn chỉ việc gõ **?** và theo sau là biểu thức hay giá trị mà bạn cần biết thông tin. Ta thử thực hiện phép chuyển đổi với giá trị 80000000h xem sao? Như ta biết ở trên, giá trị 80000000h biểu diễn một số âm, nhưng khi sử dụng Command Bar để chuyển đổi thì kết quả ta có được không như những gì chúng ta mong đợi, đây là một bug của Plug-in Command Bar.

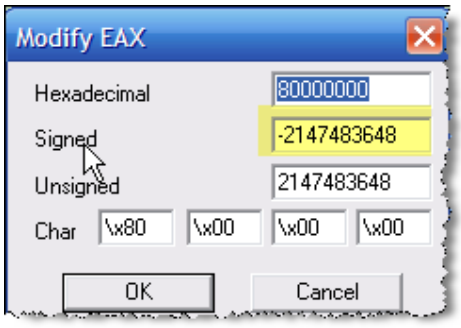

Chúng ta có thể giải quyết vấn đề này thông qua cửa số Register. Giả sử tại cửa số này tôi có giá trị thanh ghi **EAX** là 80000000h. Tôi muốn xem giá trị của nó ở hệ mười thì phải làm thế nào và giá trị âm dương của nó ra sao?



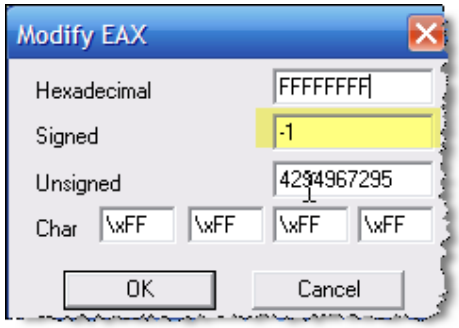

Để làm được điều này, nhấn chuột phải lên thanh ghi EAX và chọn **Modify**.Như hình minh họa dưới đây :




Cửa sổ Modify sẽ hiện ra cho phép chúng ta muốn thay đổi thanh ghi EAX thế nào tùy thích ☺. Trong trường hợp này kết quả của 80000000h đúng như những gì chúng ta trông đợi đó là -214783648.




Chúng ta thử sửa giá trị 80000000 đi và thay vào đó là một giá trị khác xem thế nào :




Ok, sau khi chỉnh sửa các bạn có thể lưu lại giá trị mà bạn đã chỉnh hoặc bỏ bằng cách nhấn Cancel.

## IV. Bảng mã ACSII

Không phải mọi số liệu mà máy tính xử lý đều là các con số, các thiết bị ngoại vi như màn hình, bàn phím, máy in đều có xu hướng làm việc với kí tự.Cũng như tất cả mọi loại dữ liệu khác, các kí tự cần phải được biểu diễn thành dạng nhị phân để máy tính có thể xử lý chúng. Một kiểu mã hóa thông dụng nhất cho các kí tự đó là mã ASCII. Khi làm việc trong Ollydbg bắt buộc bạn cũng phải tìm hiểu sơ qua về bảng mã này. Bạn phải hiểu nó để có thể làm các bước chuyển đổi giữa kí tự ở dạng hex sang kí tự cũng như những symbols tương ứng. Dưới đây là bảng mã ACSII mà bạn có thể tham khảo :

| Dec. | Hex. | Carac | Dec. | Hex. | Carac | Dec. | Hex. | Caract |
|---|---|---|---|---|---|---|---|---|
| 32 | 20 | esp | 64 | 40 | @ | 96 | 60 | ` |
| 33 | 21 | ! | 65 | 41 | A | 97 | 61 | a |
| 34 | 22 | " | 66 | 42 | B | 98 | 62 | b |
| 35 | 23 | # | 67 | 43 | C | 99 | 63 | c |
| 36 | 24 | $ | 68 | 44 | D | 100 | 64 | d |
| 37 | 25 | % | 69 | 45 | E | 101 | 65 | e |
| 38 | 26 | & | 70 | 46 | F | 102 | 66 | f |
| 39 | 27 | ' | 71 | 47 | G | 103 | 67 | g |
| 40 | 28 | ( | 72 | 48 | H | 104 | 68 | h |
| 41 | 29 | ) | 73 | 49 | I | 105 | 69 | i |
| 42 | 2A | * | 74 | 4A | J | 106 | 6A | j |
| 43 | 2B | + | 75 | 4B | K | 107 | 6B | k |
| 44 | 2C | , | 76 | 4C | L | 108 | 6C | l |
| 45 | 2D | - | 77 | 4D | M | 109 | 6D | m |
| 46 | 2E | . | 78 | 4E | N | 110 | 6E | n |
| 47 | 2F | / | 79 | 4F | O | 111 | 6F | o |
| 48 | 30 | 0 | 80 | 50 | P | 112 | 70 | p |
| 49 | 31 | 1 | 81 | 51 | Q | 113 | 71 | q |
| 50 | 32 | 2 | 82 | 52 | R | 114 | 72 | r |
| 51 | 33 | 3 | 83 | 53 | S | 115 | 73 | s |
| 52 | 34 | 4 | 84 | 54 | T | 116 | 74 | t |
| 53 | 35 | 5 | 85 | 55 | U | 117 | 75 | u |
| 54 | 36 | 6 | 86 | 56 | V | 118 | 76 | v |
| 55 | 37 | 7 | 87 | 57 | W | 119 | 77 | w |
| 56 | 38 | 8 | 88 | 58 | X | 120 | 78 | x |
| 57 | 39 | 9 | 89 | 59 | Y | 121 | 79 | y |
| 58 | 3A | : | 90 | 5A | Z | 122 | 7A | z |
| 59 | 3B | ; | 91 | 5B | [ | 123 | 7B | { |
| 60 | 3C | < | 92 | 5C | \ | 124 | 7C | \| |
| 61 | 3D | = | 93 | 5D | ] | 125 | 7D | } |
| 62 | 3E | > | 94 | 5E | ^ | 126 | 7E | ~ |
| 63 | 3F | ? | 95 | 5F | | 127 | 7F | |

Một ví dụ với sự giúp đỡ của Plug-in **Command Bar** sẽ cho bạn thấy được kết quả trực quan :

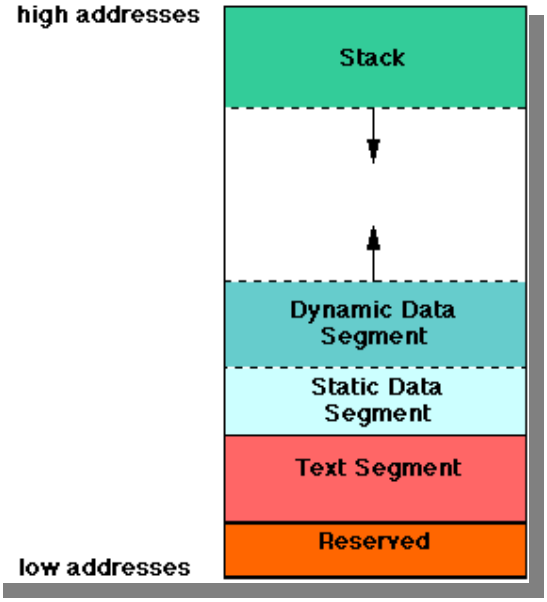

Ngoài ra cửa sổ **Dump** trong Olly cũng giúp bạn có được những thông tin quan trọng trong quá trình bạn Debug target :

| Address | Hex dump | | | | | | | | | | | | | | | | ASCII |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00403000 | 52 | 65 | 76 | 65 | 72 | 73 | 65 | 20 | 45 | 6E | 67 | 69 | 6E | 65 | 65 | 72 | Reverse Engineer |
| 00403010 | 69 | 6E | 67 | 20 | 41 | 73 | 73 | 6F | 63 | 69 | 61 | 74 | 69 | 6F | 6E | 00 | ing Association. |
| 00403020 | 43 | 6F | 6E | 67 | 72 | 61 | 74 | 75 | 6C | 61 | 74 | 69 | 6F | 6E | 21 | 20 | Congratulation! |
| 00403030 | 59 | 6F | 75 | 27 | 76 | 65 | 20 | 64 | 6F | 6E | 65 | 20 | 77 | 69 | 74 | 68 | You've done with |
| 00403040 | 20 | 69 | 74 | 00 | 4E | 6F | 2C | 6E | 6F | 21 | 20 | 54 | 72 | 79 | 20 | 69 | it.No,no! Try i |
| 00403050 | 74 | 20 | 61 | 67 | 61 | 69 | 6E | 21 | 00 | 25 | 78 | 00 | 00 | 00 | 00 | 00 | t again!.%x..... |

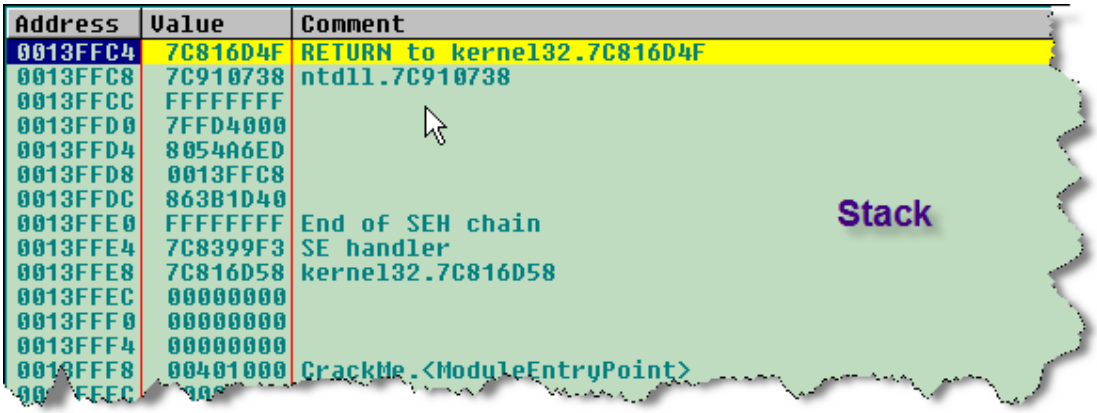
## V. STACK

Như trong phần đầu tiên tôi đã nói sơ quan vế **STACK**, nó là một vùng của bộ nhớ dùng để lưu trữ tạm thời các dữ liệu và địa chỉ. Stack làm việc theo nguyên lý **LIFO (Last In, First Out)**, tức là phần tử nào được cất vào cuối cùng trong stack sẽ là phần tử được lấy ra đầu tiên. Bạn cứ tưởng tượng như bạn đang xếp một chồng đĩa, thì chiếc đĩa cuối cùng mà bạn xếp sẽ nằm trên cùng, tức là đỉnh của Stack nó sẽ là chiếc đĩa được lấy ra đầu tiên nếu như bạn muốn lấy tiếp chiếc đĩa thứ hai bên dưới nó. Cấu trúc dữ liệu làm việc theo kiểu LIFO này là ý tưởng cho việc lưu trữ những dữ liệu tạm thời, hoặc những thông tin không cần thiết phải được lưu trữ trong một thời gian dài. Stack thường là nơi lưu trữ các local variables, những lời gọi hàm (function calls) và các thông tin khác được sử dụng để dọn dẹp stack sau khi một hàm hay một thủ tục được gọi.

Một tính năng quan trọng khác của stack là nó *grows down* theo không gian địa chỉ: có nghĩa là càng nhiều dữ liệu được thêm vào trong stack, nó được thêm vào tại các giá trị địa chỉ thấp hơn theo cơ chế tăng dần. Xem hình minh họa  về sơ đồ không gian bộ nhớ :




Làm việc với Stack có 2 thanh ghi chính là ESP và EBP, và các câu lệnh PUSH và POP. Trong Ollydbg bạn có thể quan sát thấy cửa sổ Stack rất trực quan :




Okie vậy là phần hai trong loạt bài viết về Olly đến đây là hết, trong phần tiếp theo tôi sẽ giới thiệu tới các bạn về các thanh ghi cũng như những tính năng của từng thanh ghi. Tôi sẽ cố gắng viết xong trong thời gian sớm nhất! ☺

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:
**kienmanowar[at]reaonline.net**