Trường Đại Học Khoa Học Tự Nhiên

Đại Học Quốc Gia Hồ Chí Minh

CRACK PHẦN MỀM

**BÀI BÁO CÁO**

**ĐỒ ÁN III– KTMT&HN**

**CRACK PHẦN MỀM**

**THÔNG TIN NHÓM**

1. Trần Duy Phương – 1512418
2. Dương Vinh Quang – 1512428
3. Nguyễn Đăng Quang – 1512432

NỘI DUNG

[I. BẢNG PHÂN CÔNG CÔNG VIỆC VÀ Ý TƯỞNG THIẾT KẾ 2](#_Toc478753735)

[A. Ý TƯỞNG THIẾT KẾ CÔNG VIỆC 2](#_Toc478753736)

[B. BẢNG PHÂN CÔNG CÔNG VIỆC 15](#_Toc478753737)

[II. CHẠY KIỂM TRA VÀ CHỤP HÌNH 16](#_Toc478753738)

[III. CÁC CHỨC NĂNG LÀM ĐƯỢC VÀ CHƯA LÀM ĐƯỢC 16](#_Toc478753739)

[A. Đã làm được 16](#_Toc478753740)

[B. Chưa làm được 16](#_Toc478753741)

[IV. ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH ĐỒ ÁN: 17](#_Toc478753742)

[V. CÁC TÀI LIỆU THAM KHẢO 17](#_Toc478753743)

# BẢNG PHÂN CÔNG CÔNG VIỆC VÀ Ý TƯỞNG THIẾT KẾ

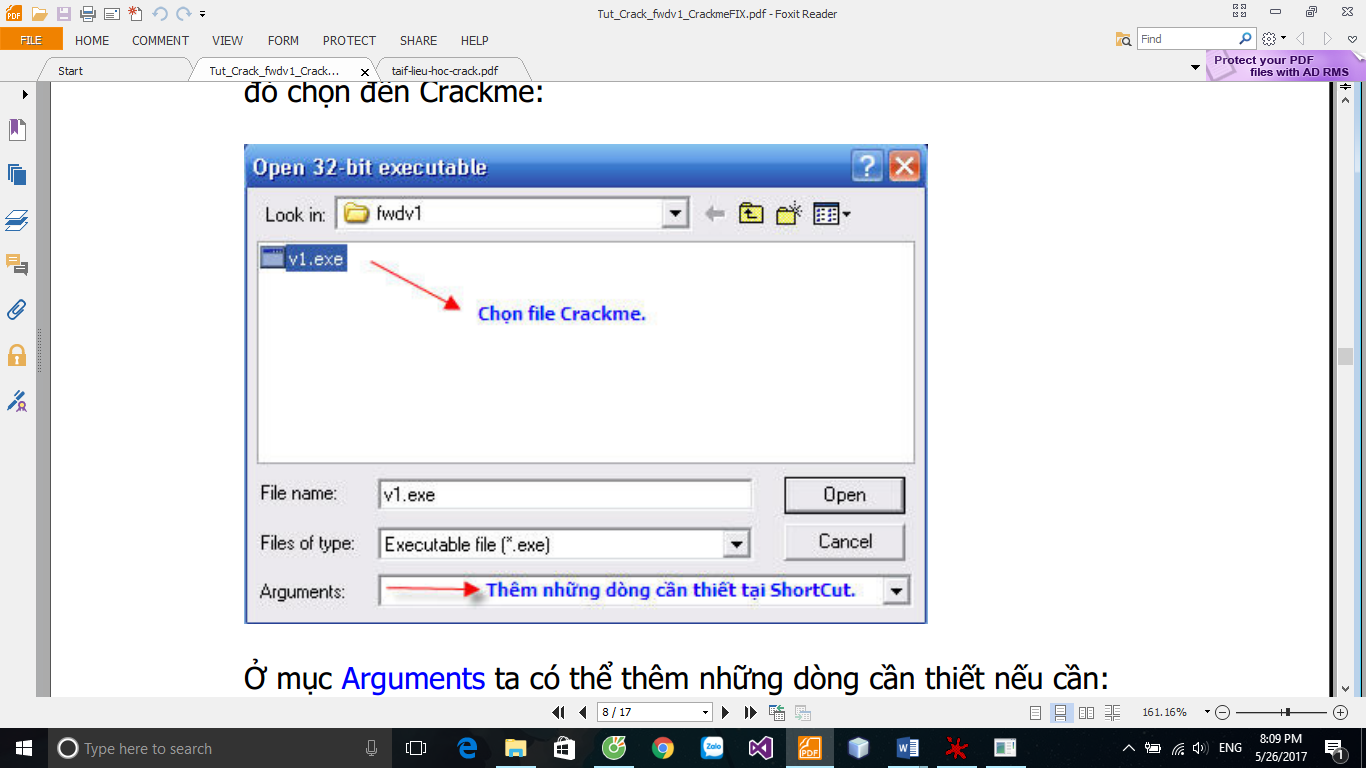
***LƯU Ý:***

* Dùng OllyDBG phân tích chương trình dưới dạng assembler rồi từ đó đọc hiểu code tìm key và viết keygen phù hợp cho chương trình
* Nhóm gồm 3 sinh viên có MSSV: 1512418, 1512428, 1512432; ta sẽ lấy: [(8 + 8 + 2) % 3] + 1 = 1. Nhóm sẽ làm đề 1

## Ý TƯỞNG THIẾT KẾ CÔNG VIỆC VÀ CÁCH CRACK Ở MỖI BÀI

1. **Bài 1**

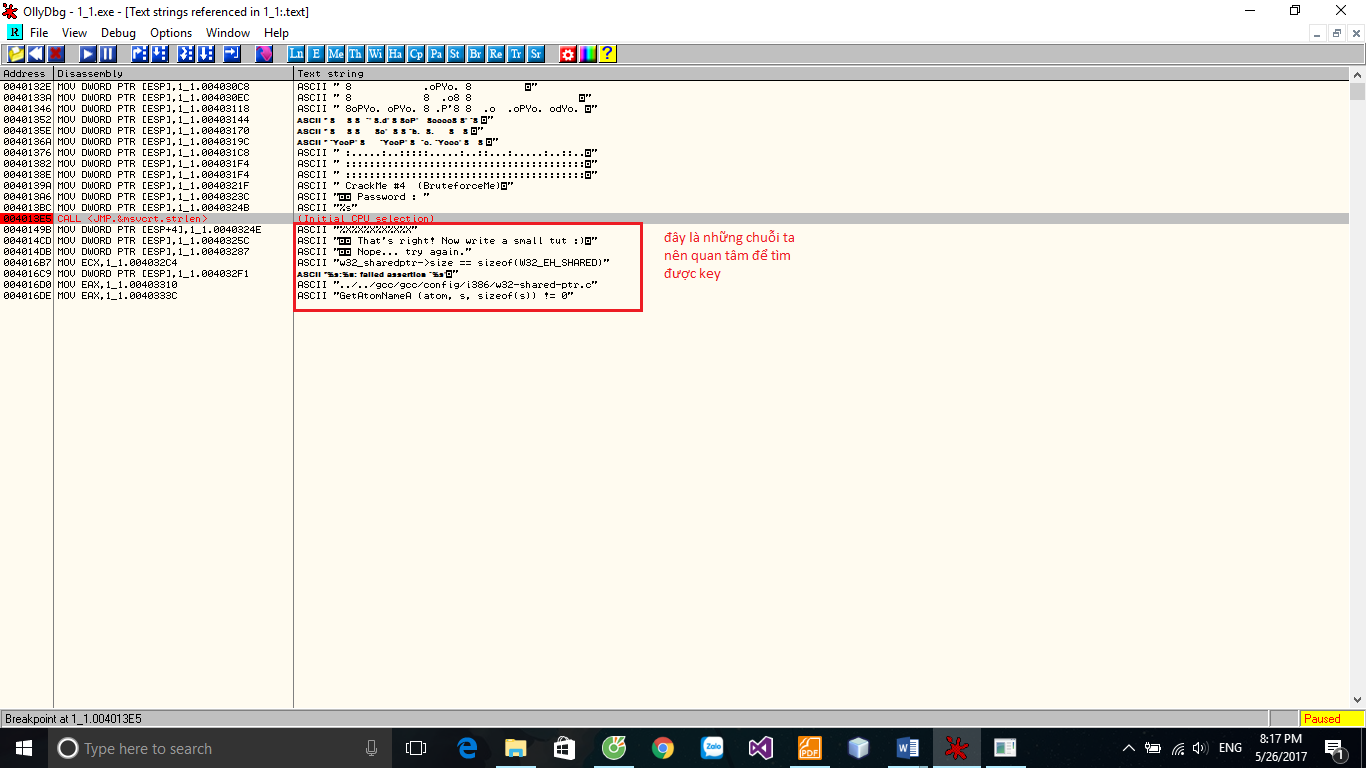
* Mở Olly lên, nhấn F3 (Open) (hoặc chọn qua Menu, nút bấm), sau đó chọn đến Crackme:



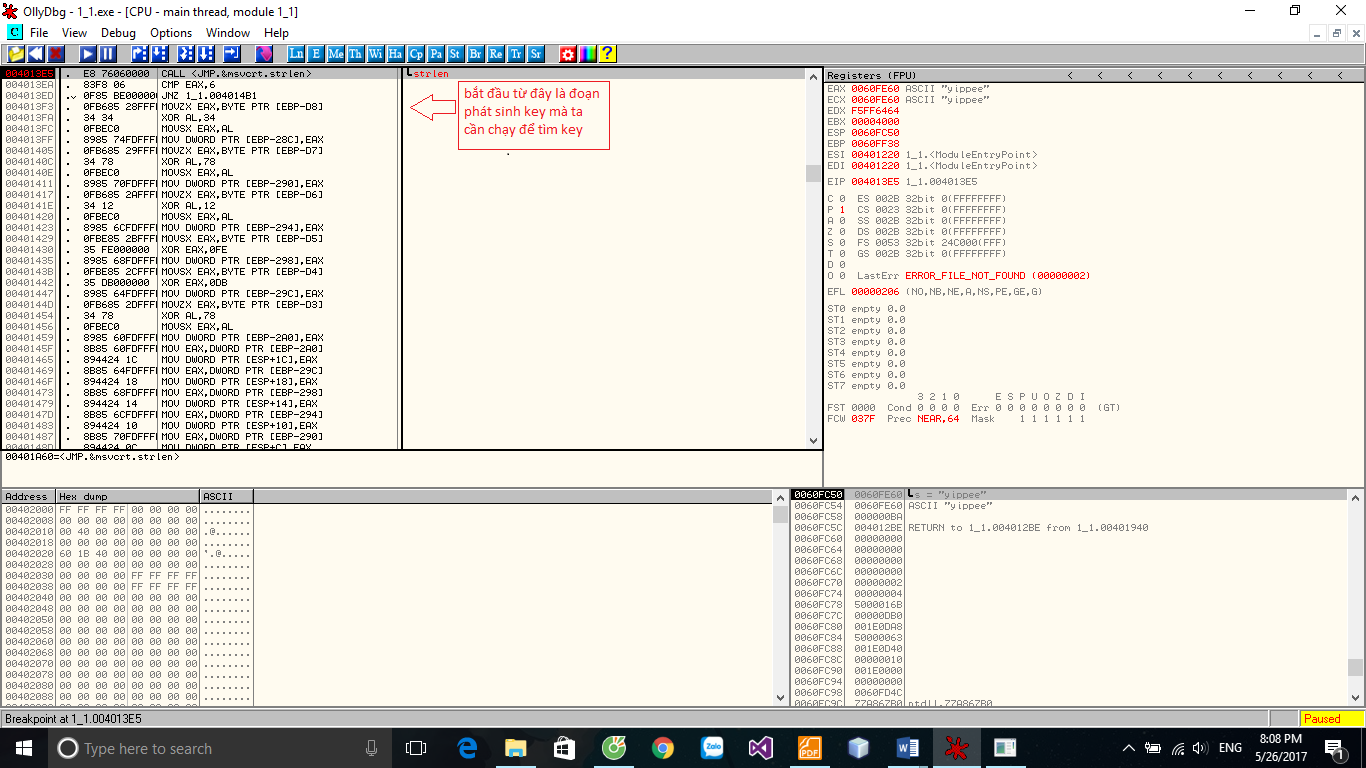
* Sau đó ta nhấp F9 để chạy, ta nhắp chuột phải chọn chức năng tìm kiếm string trong Olly như sau:



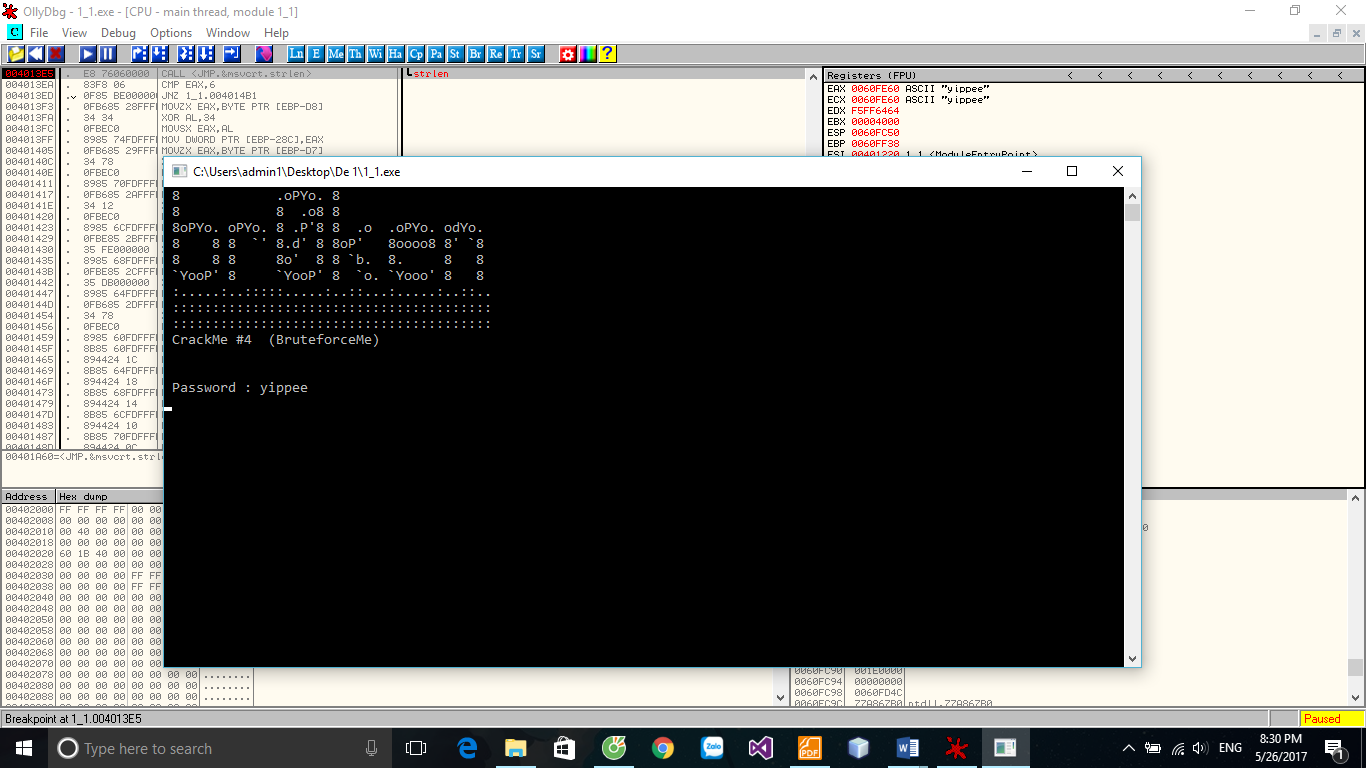
* Sau đó cửa sổ sẽ hiện lên các chuỗi:



* Khi nhắp đúp vào chuỗi “that’s right! Now write a small tut :)” bạn sẽ đi tới dòng lệnh đó kéo lên trên là bạn đã tìm được đoạn phát sinh key:



* Sau khi đặt break point ở đó bằng cách nhắp đúp chuột vào bên phải ô địa chỉ, ta nhấp ctrl F7 để nhập vào key ban đầu (có thể sai) để thử và tìm key chính xác của nó
* Ở ví dụ này tôi sẽ nhập key ban đầu vào là “yippee”:



* Ta tiếp tục nhấn f7 để nhảy tới từng dòng lệnh của chương trình:

004013E5 |. E8 76060000 CALL <JMP.&msvcrt.strlen> <= tính chiều dài chuỗi nhập vào

004013EA |. 83F8 06 CMP EAX,6 <= so sánh chiều dài chuỗi key với 6

004013ED |. /0F85 BE000000 JNZ 1\_1.004014B1 <= nếu không bằng nhảy tới câu lệnh có địa chỉ 004014B1 nếu cờ Z = 0

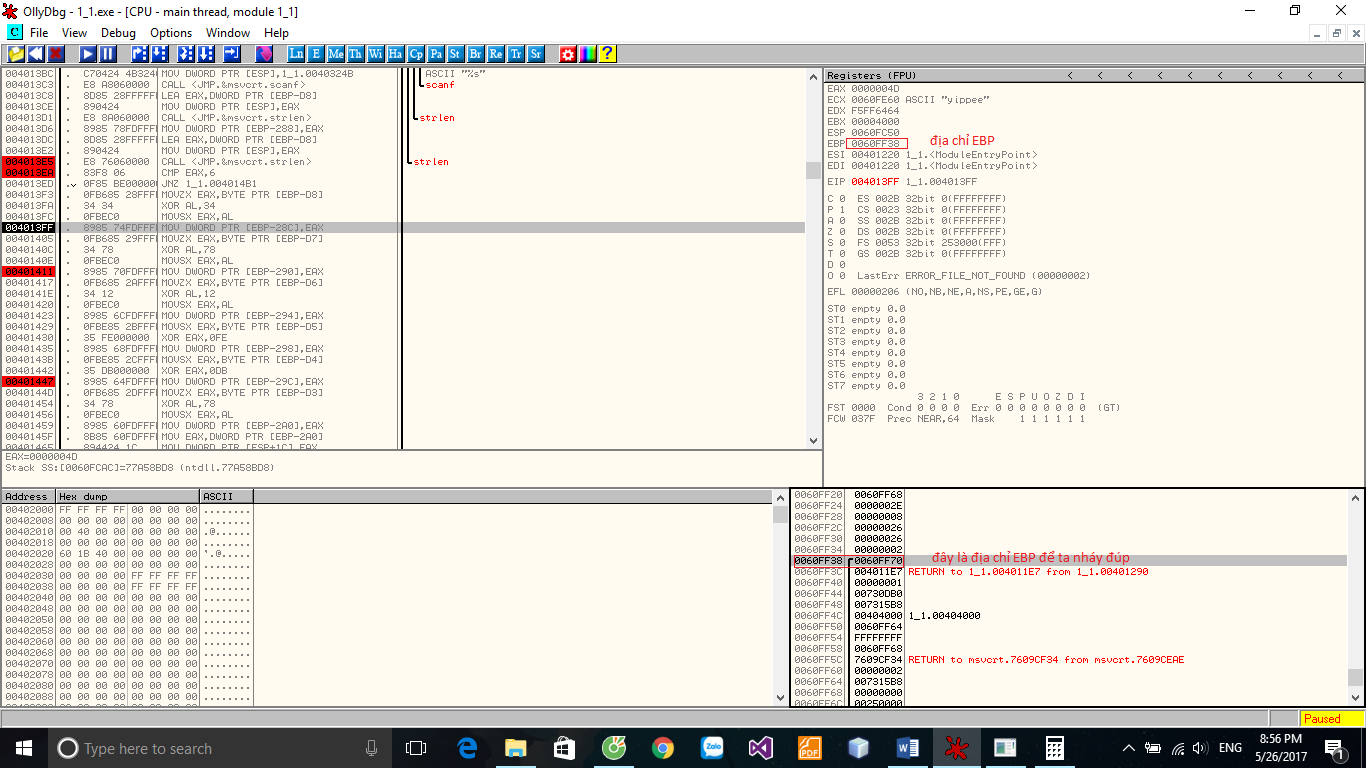
004013F3 |. 0FB685 28FFFF>MOVZX EAX,BYTE PTR [EBP-D8] <= dùng để copy 1 thanh ghi ô nhớ có chiều dài nhỏ hơn sang 1 thanh ghi ô nhớ có chiều dài lớn hơn, phần còn lại tự động sẽ gán bằng 0, trong đó EBP là địa chỉ dữ liệu trong stack – D8 (đang chứa kí tự đầu của chuỗi là ‘y’) vào EAX

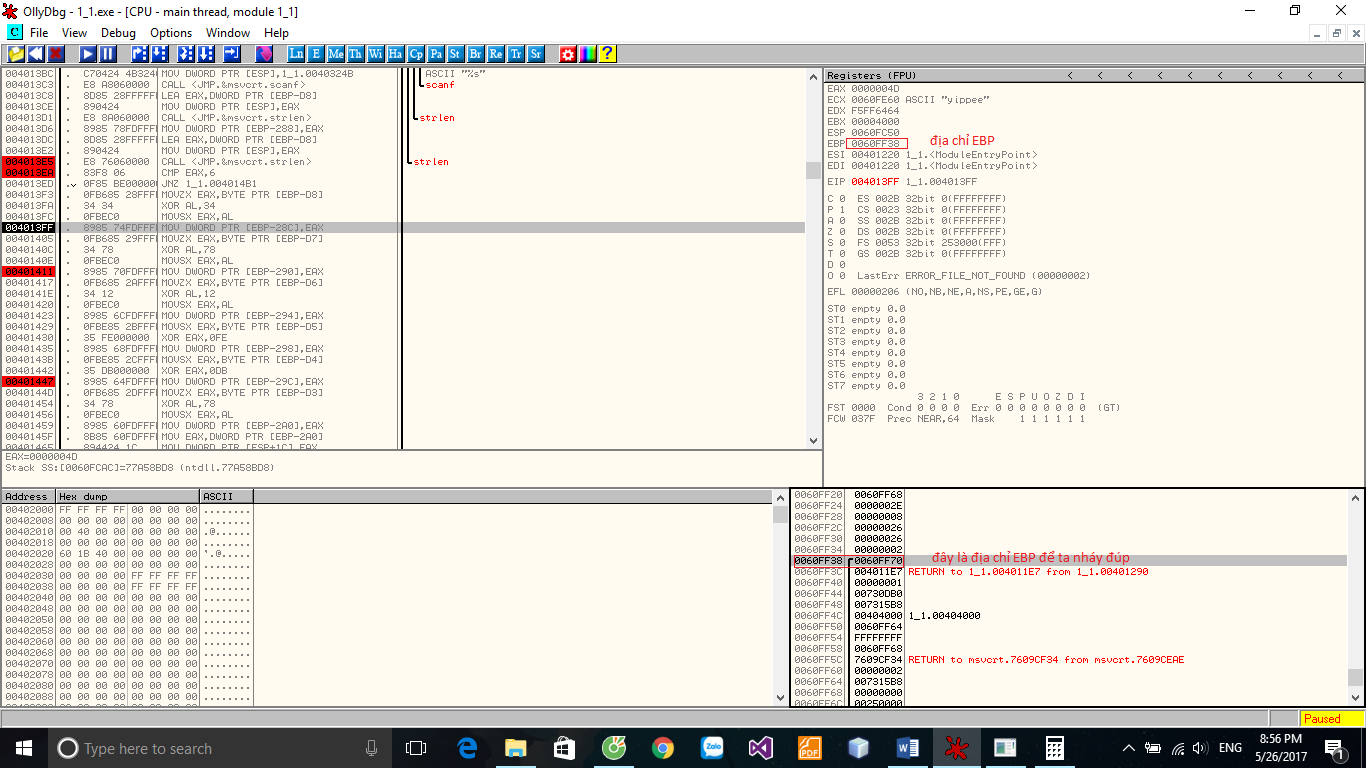
004013FA |. 34 34 XOR AL,34 <= 34 xor thanh ghi AL bây giờ là 79 hex tương đương với ‘y’ trong mã ascii ra kết quả là 4D

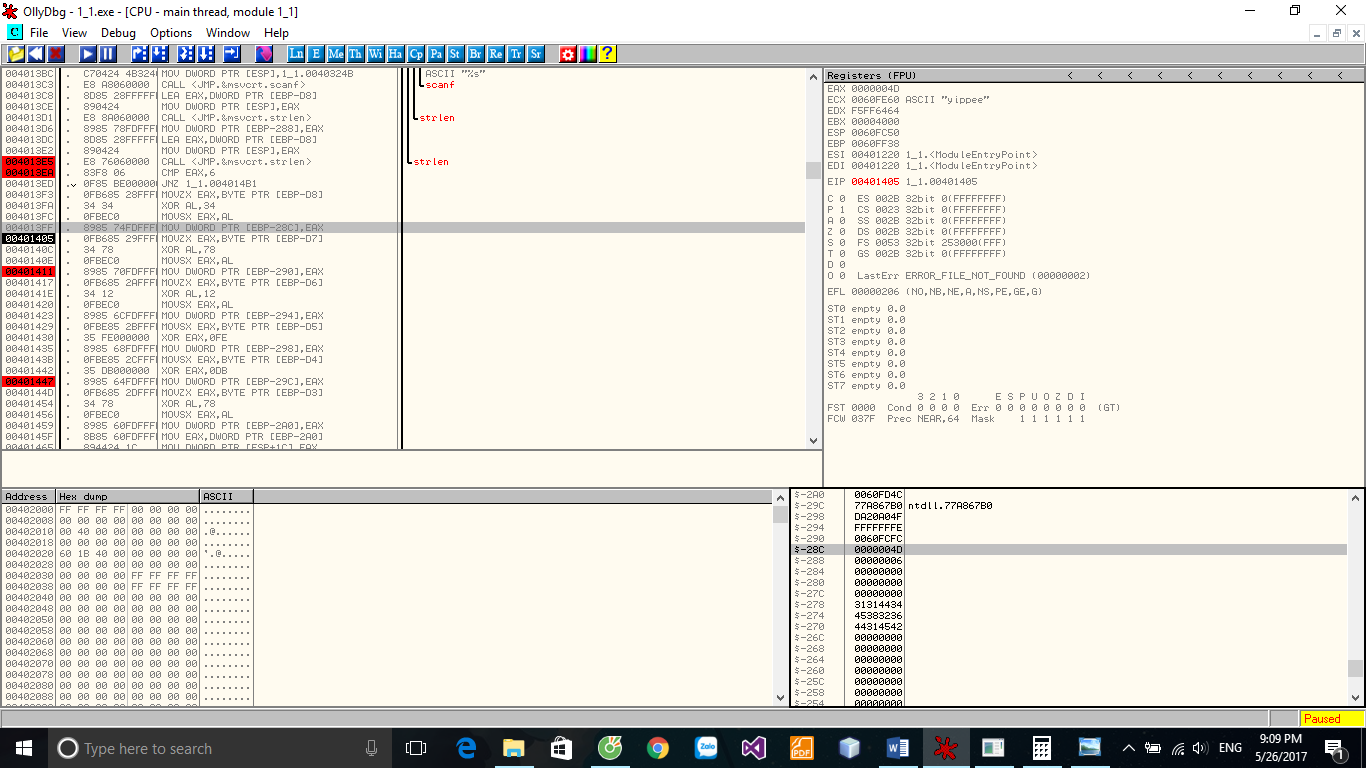
004013FC |. 0FBEC0 MOVSX EAX,AL <= sau đó gán giá trị vào thanh ghi EAX

004013FF |. 8985 74FDFFFF MOV DWORD PTR [EBP-28C],EAX <= lưu EAX vào ô có địa chỉ EBP-28C

\*lưu ý: ta có thể thấy địa chỉ của EBP dò trong bảng stack ta tìm được nó nhắp đúp chuột vào và kéo lên ta có thể đi tới ô có địa chỉ EBP-28C







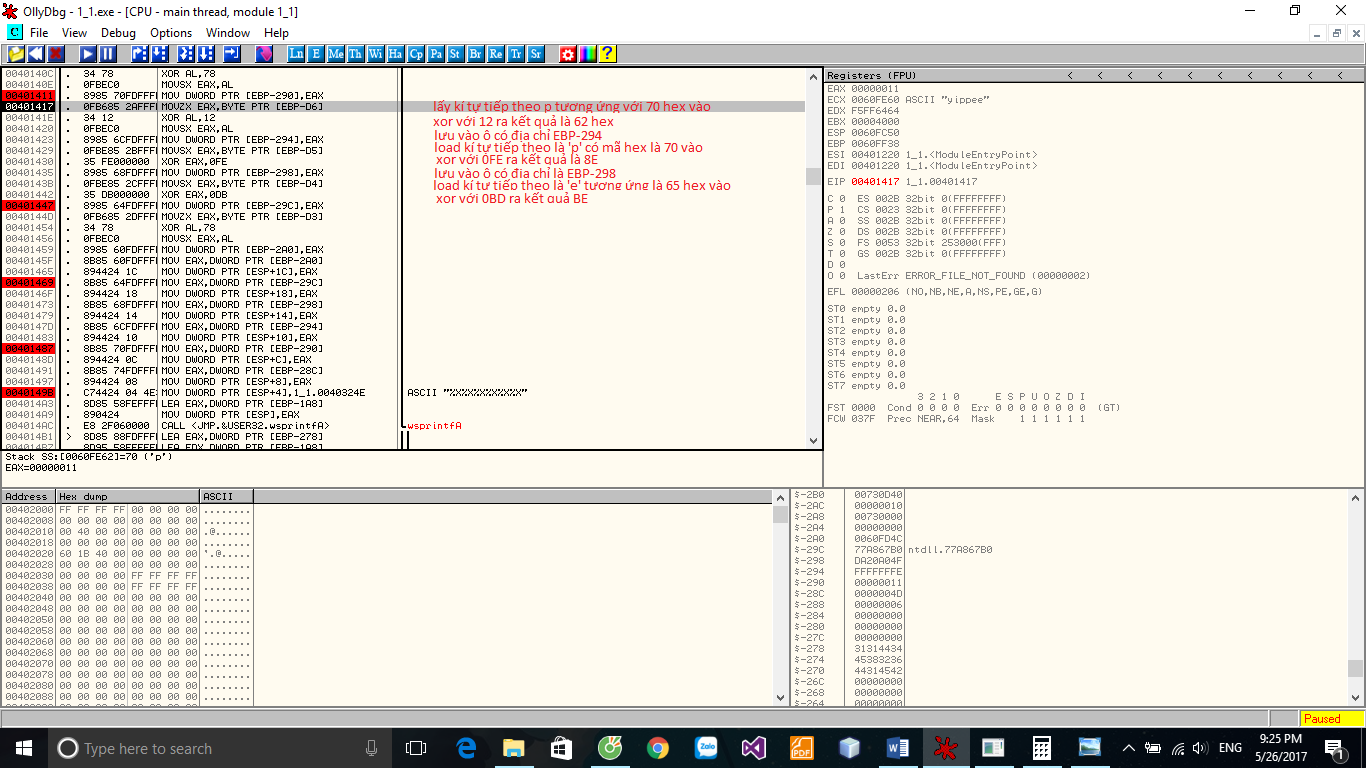
00401405 |. 0FB685 29FFFF>MOVZX EAX,BYTE PTR [EBP-D7] <= dùng để copy 1 thanh ghi ô nhớ mà EBP – D8 (đang chứa kí tự ‘i’) vào EAX là 69 hex

0040140C |. 34 78 XOR AL,78 <= 78 xor thanh ghi AL bây giờ là 69 hex ra kết quả là 11

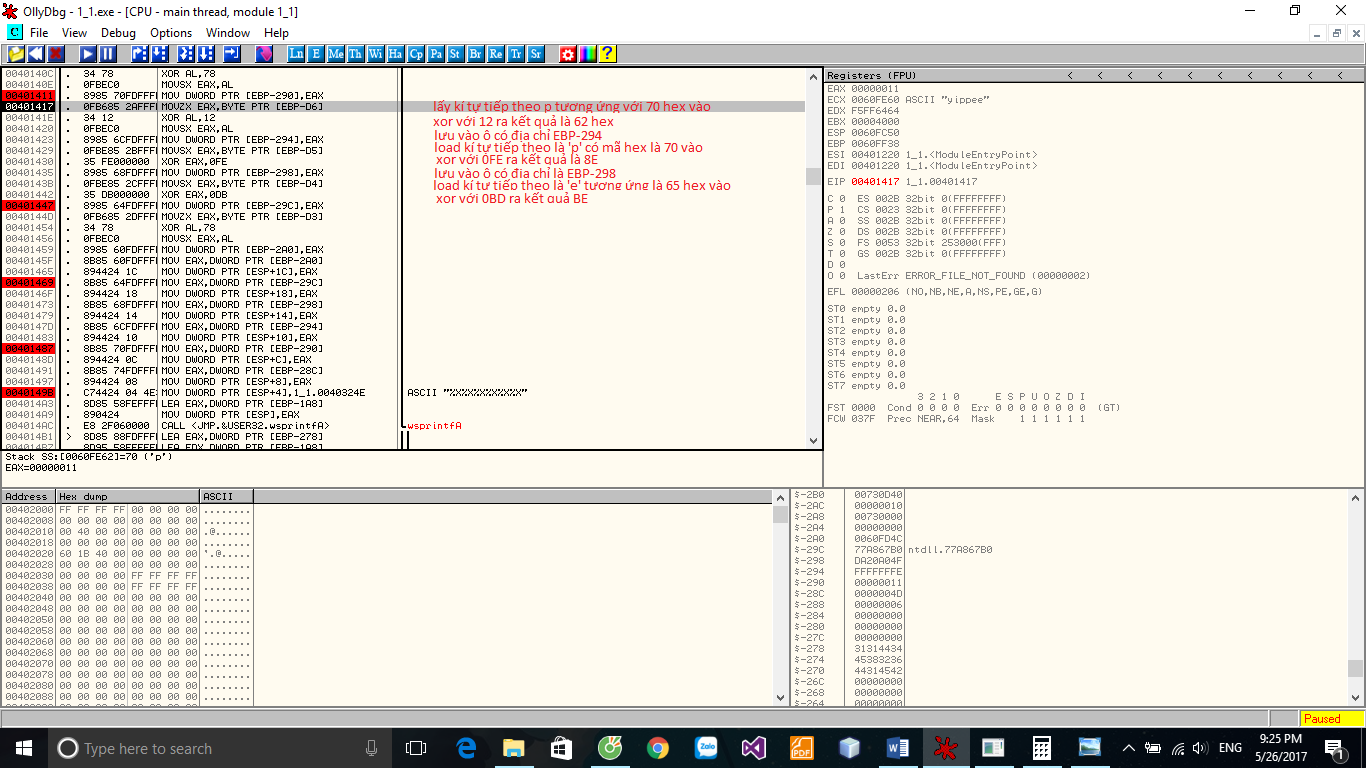
0040140E |. 0FBEC0 MOVSX EAX,AL <= copy 11 vào EAX

00401411 |. 8985 70FDFFFF MOV DWORD PTR [EBP-290],EAX <= lưu EAX vào địa chỉ EBP-290

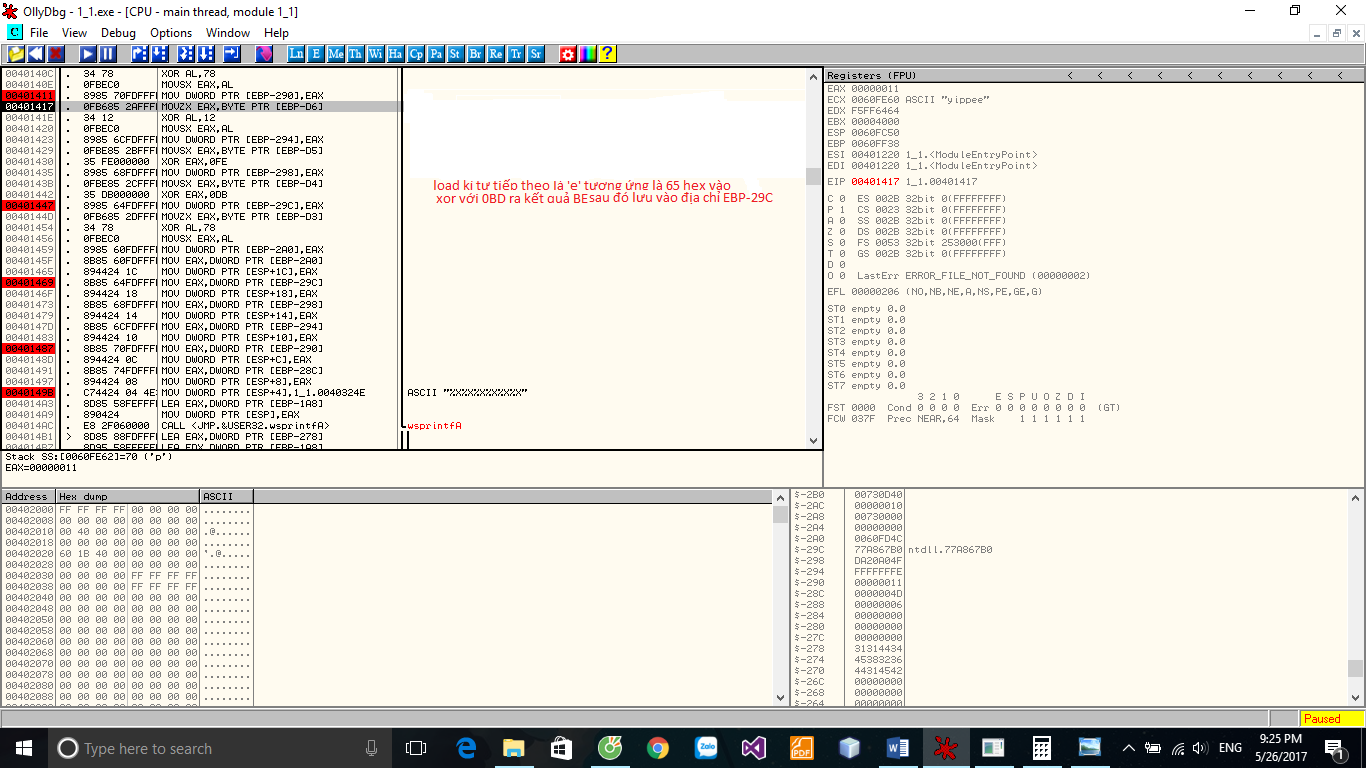
* Tiếp theo ta làm tương tự lấy kí tự tiếp theo XOR 12 và lưu vào EBP-294:



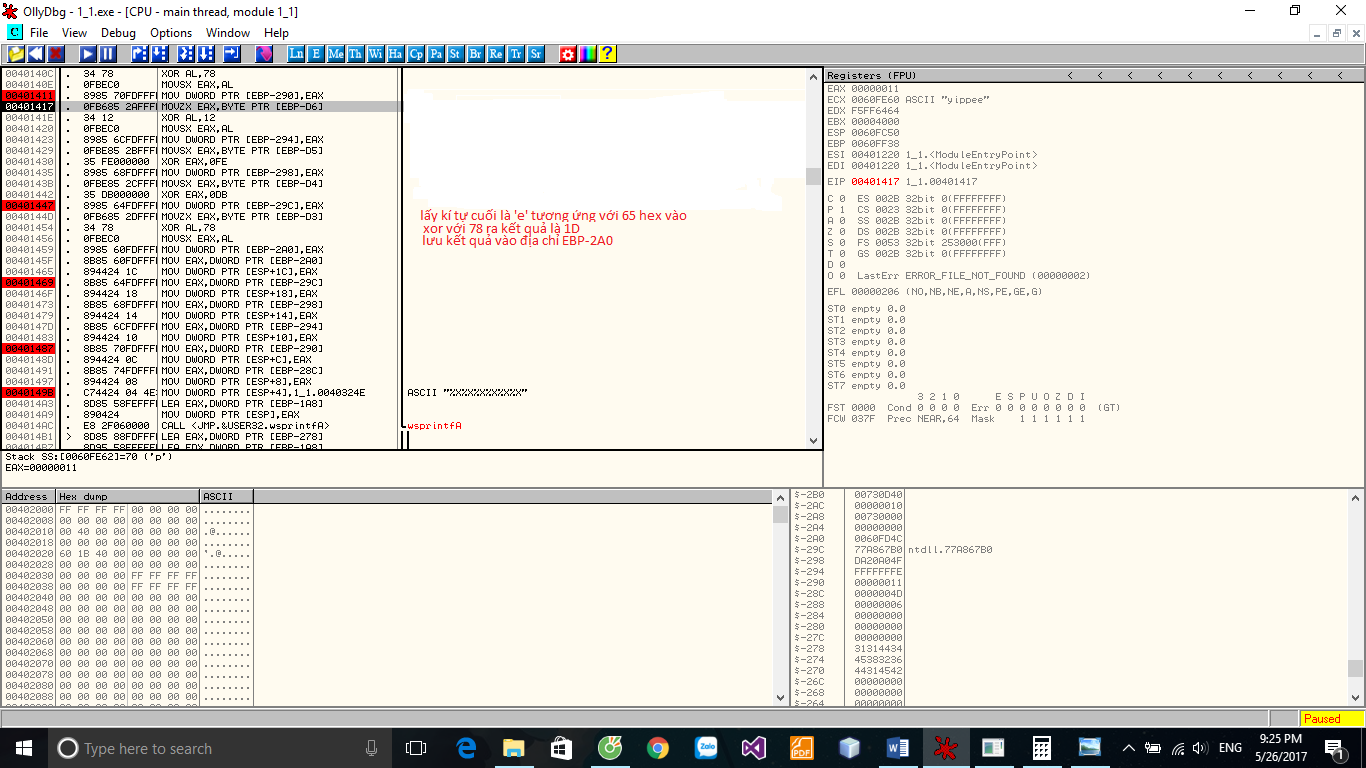
* Ta làm tương tự lấy kí tự tiếp theo XOR 0FE và lưu vào EBP-298:



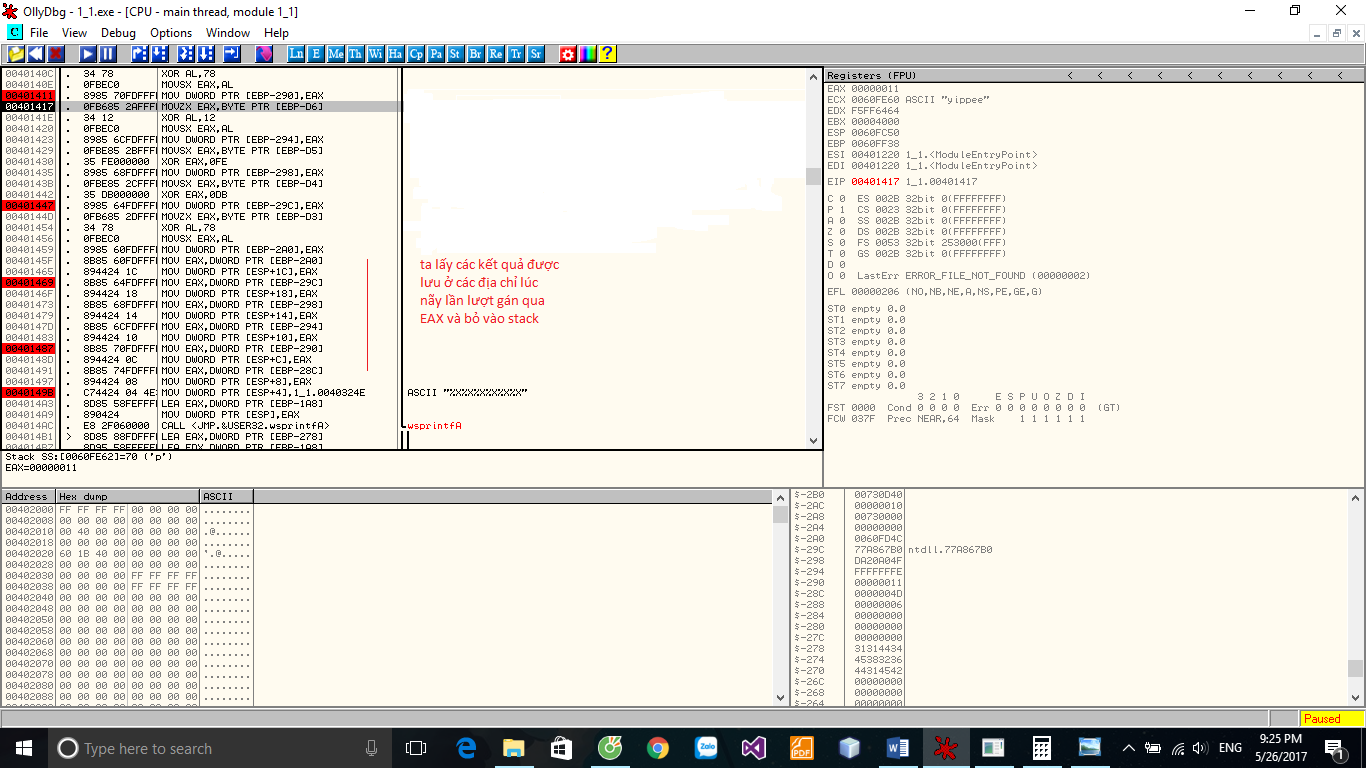
* Ta làm lấy kí tự tiếp theo XOR 0DB và lưu vào EBP-29C:



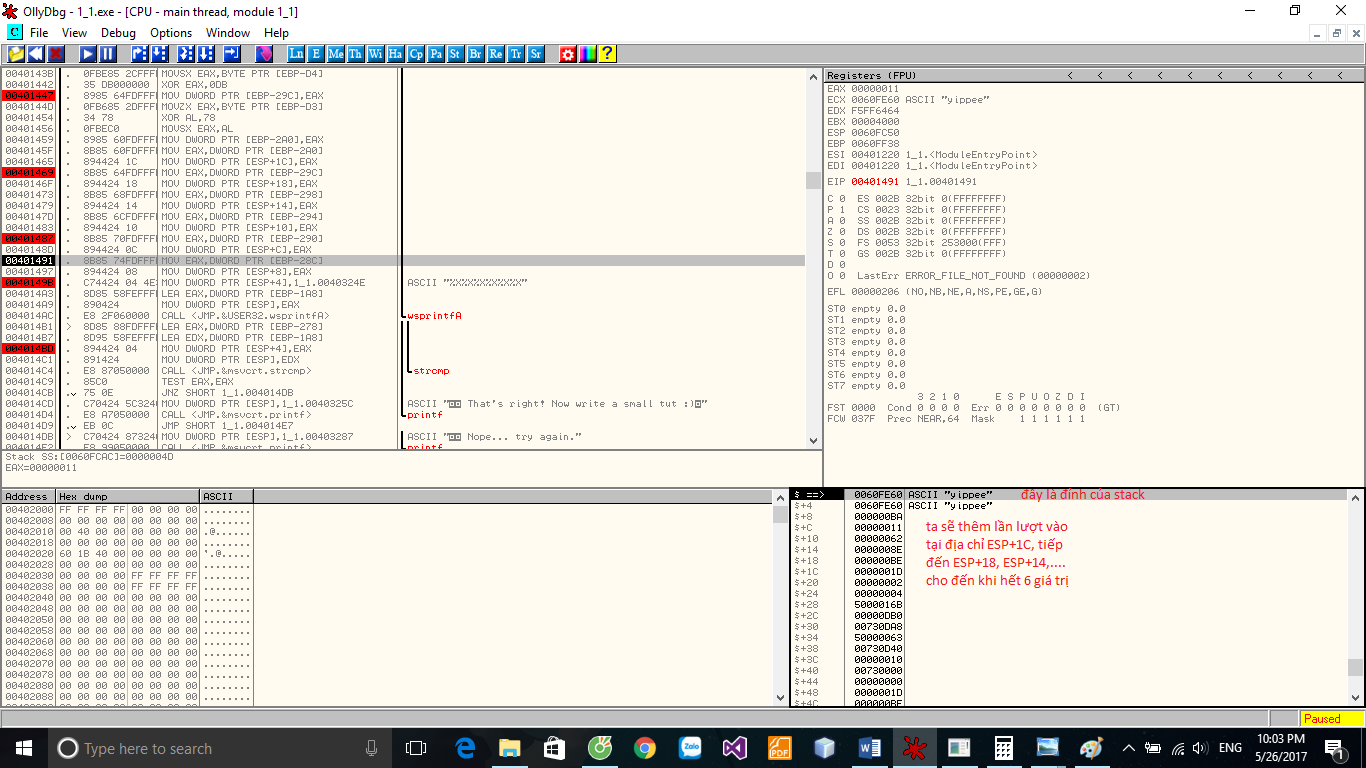
* Ta lấy kí tự cuối cùng vào XOR 78 và lưu vào EBP-2A0:



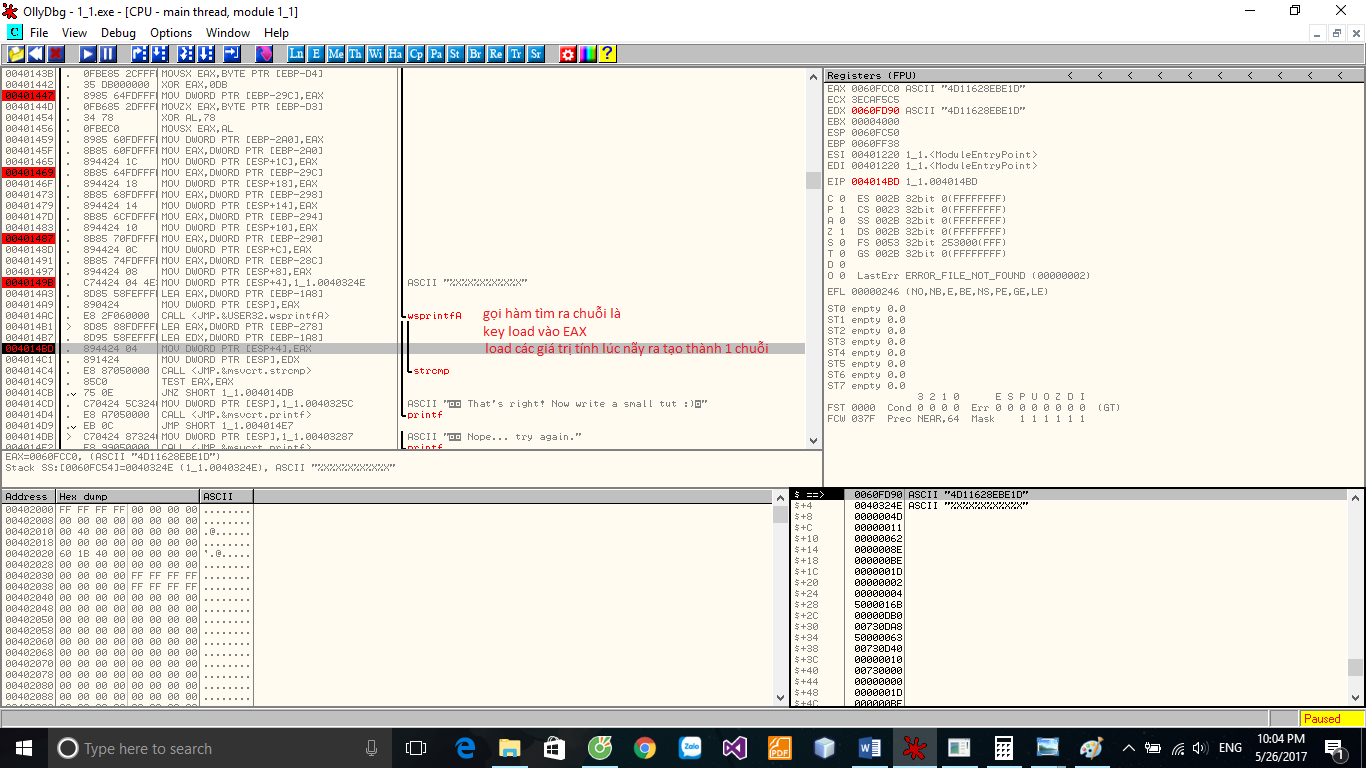
* Sau đó ta lấy các kết quả vừa tính đem vào stack:



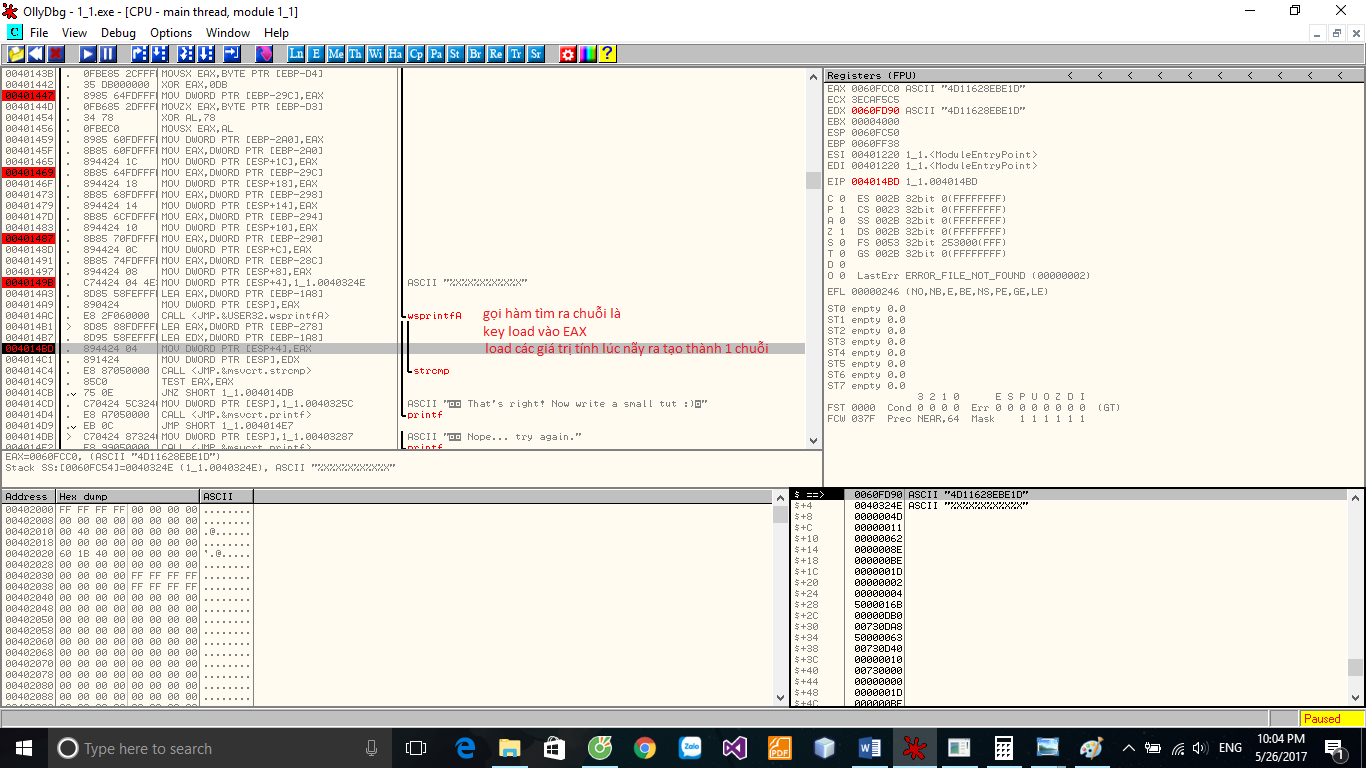
* Tại đây giá trị cụ thể lần lượt là:



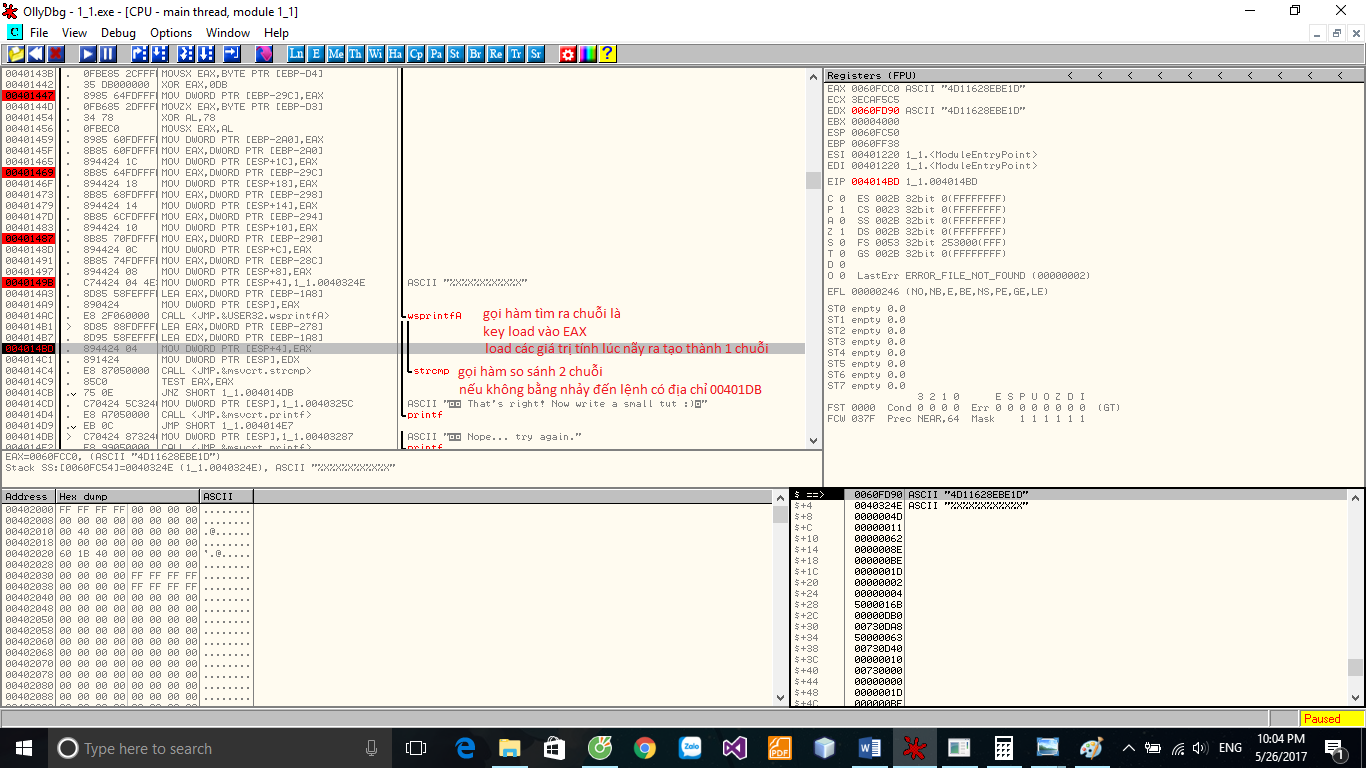
* Tiếp đến:

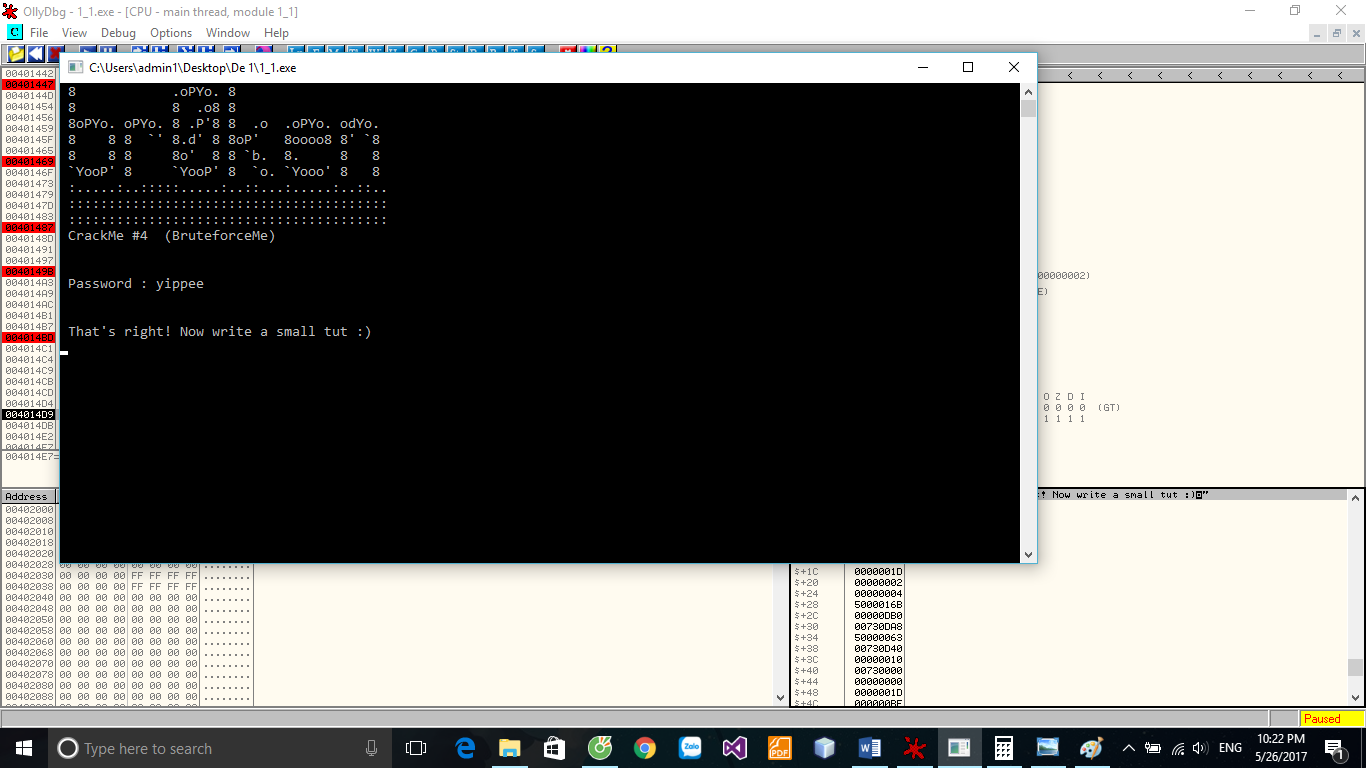


* Các giá trị trên thanh ghi hiện thời:

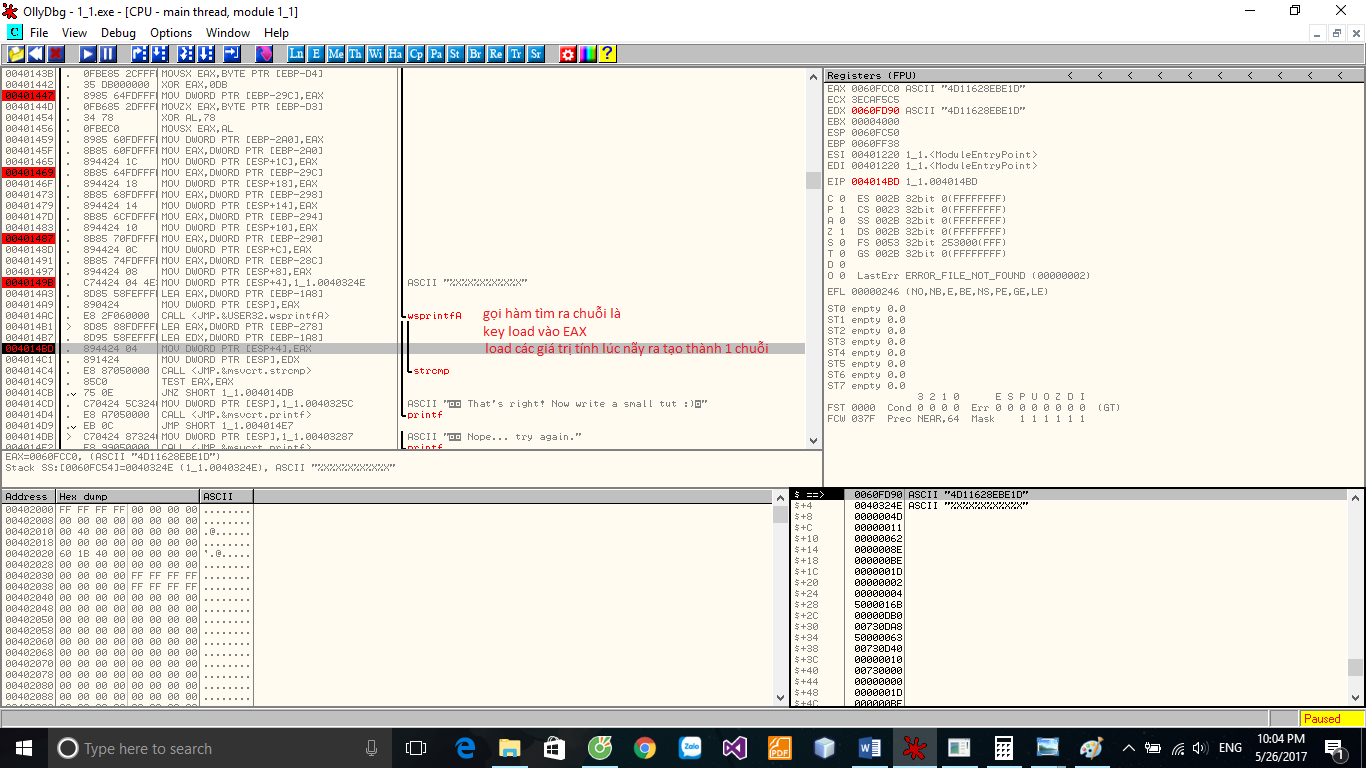


* Nếu ta so sánh 2 giá trị này khớp nhau nếu sai thì sẽ thực hiện lệnh jump tới địa chỉ 004014DB xuất ra thông báo “Nope….try again.” Còn không sẽ thông báo đúng và kết thúc chương trình:

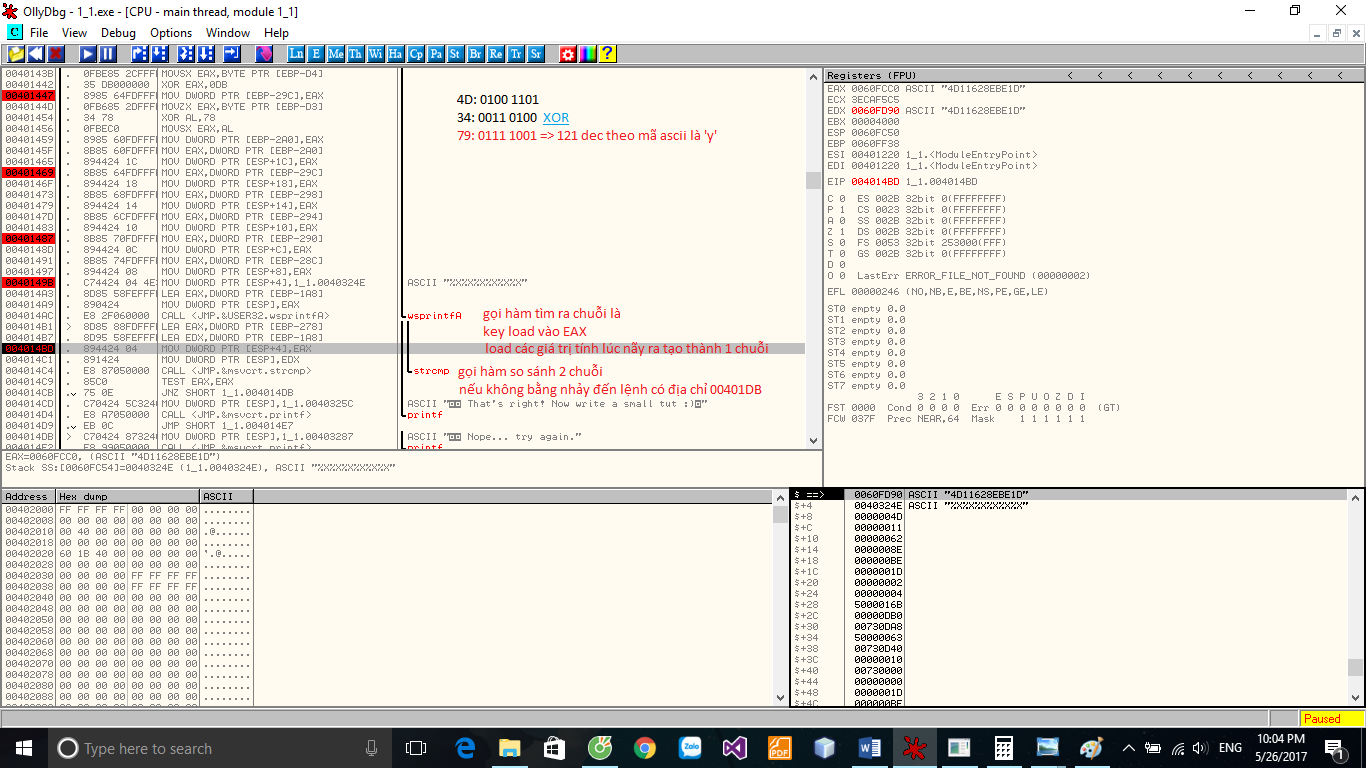




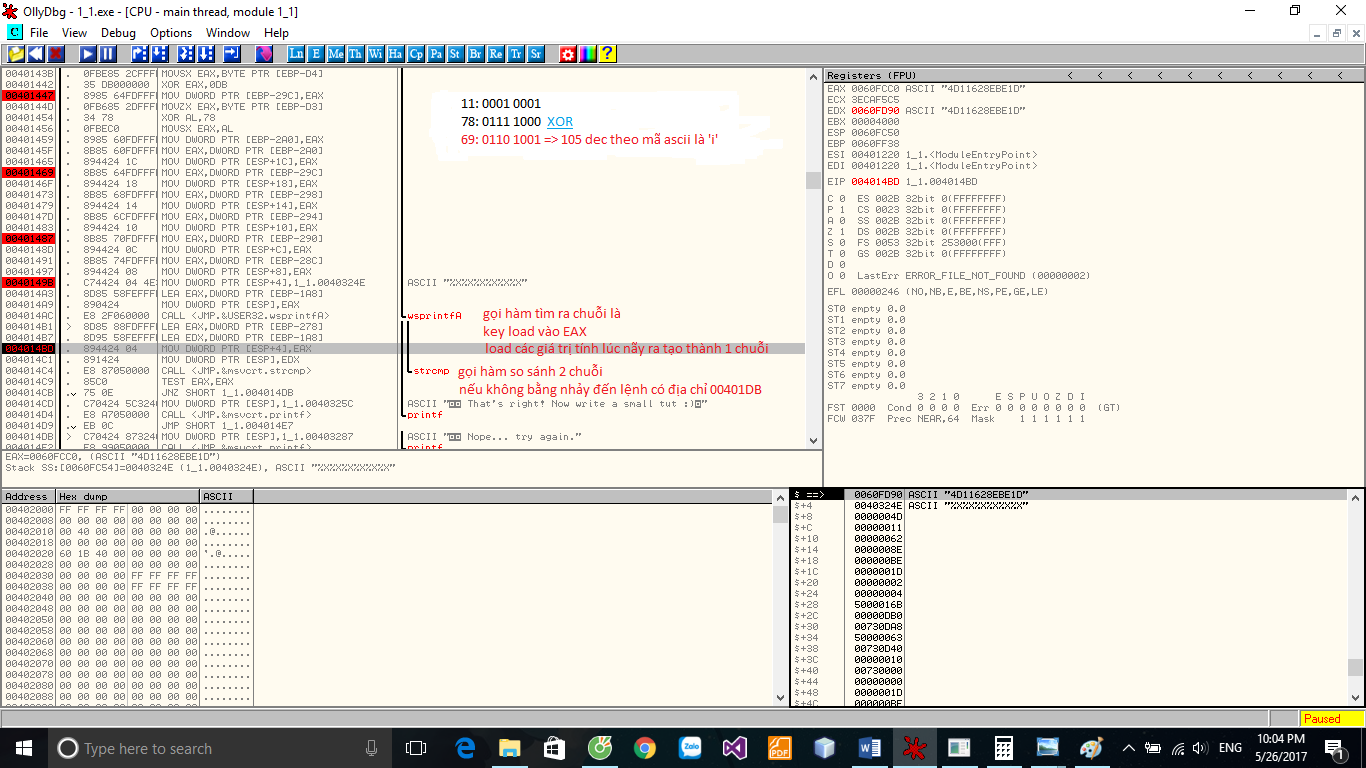
* Thuật toán phát sinh key:
* Ở chương trình này ta nhận thấy khi chạy tới cuối có chuỗi “4D11628EBE1D” để so sánh nên ta chỉ cần tìm ra quy luật sau khi XOR để có key:



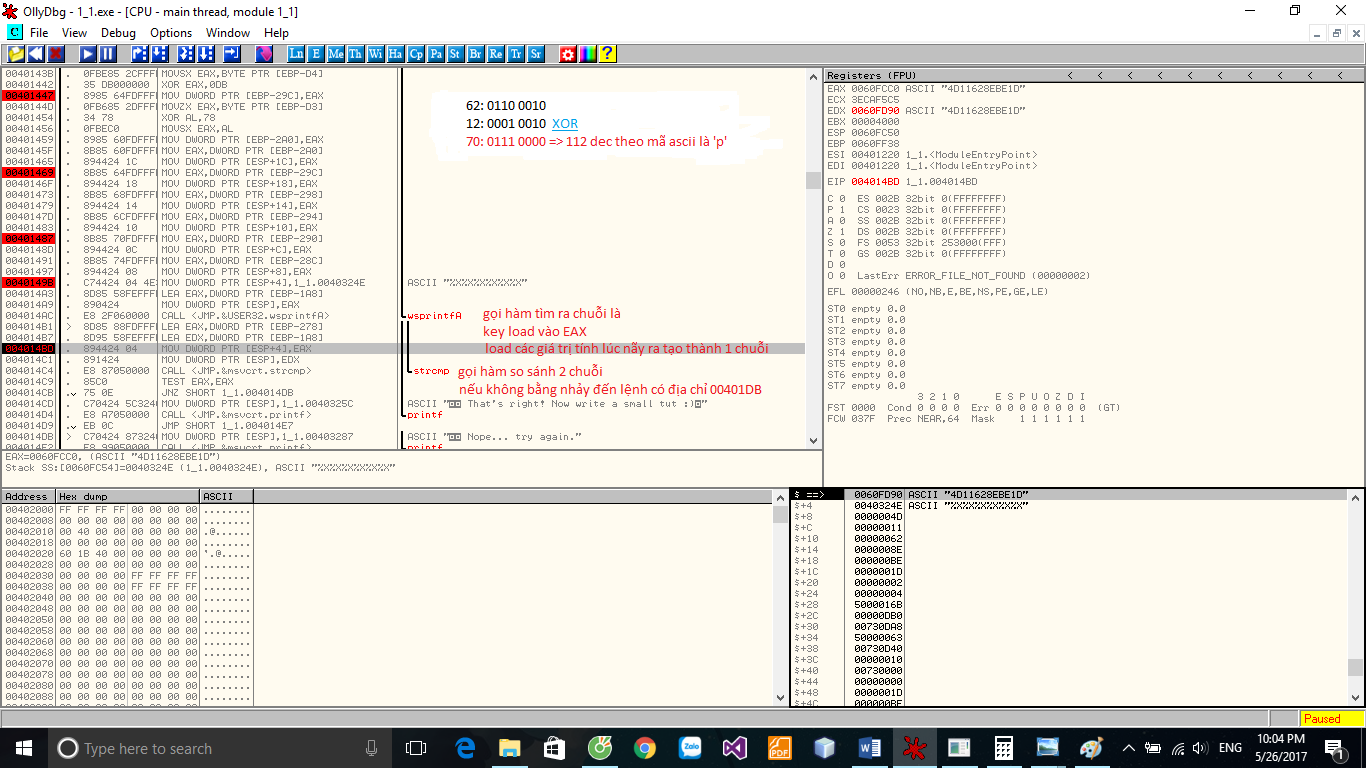
* Vậy đầu tiên để ra 4D trong hex ta phải tìm x khi XOR 34:



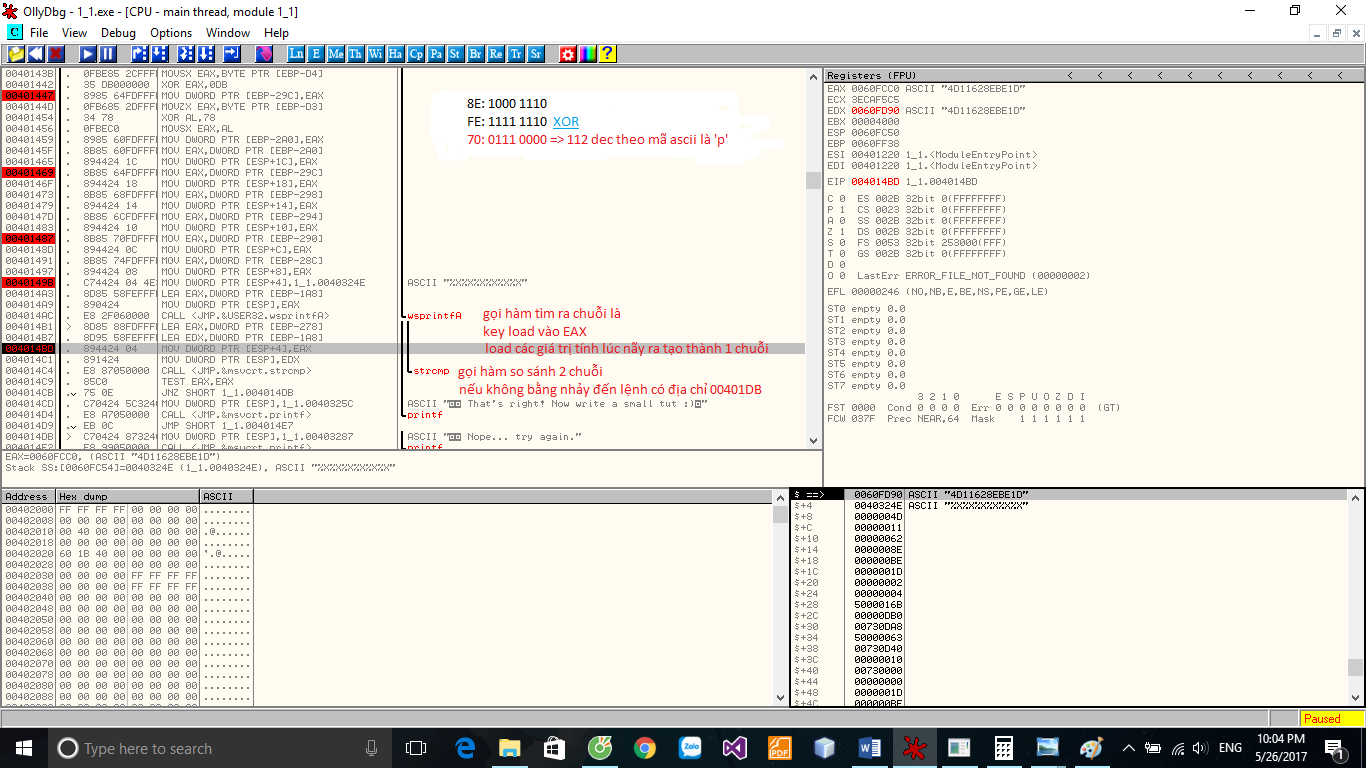
* Tương tự ta tìm 11 hex ta phải tìm x XOR 78:



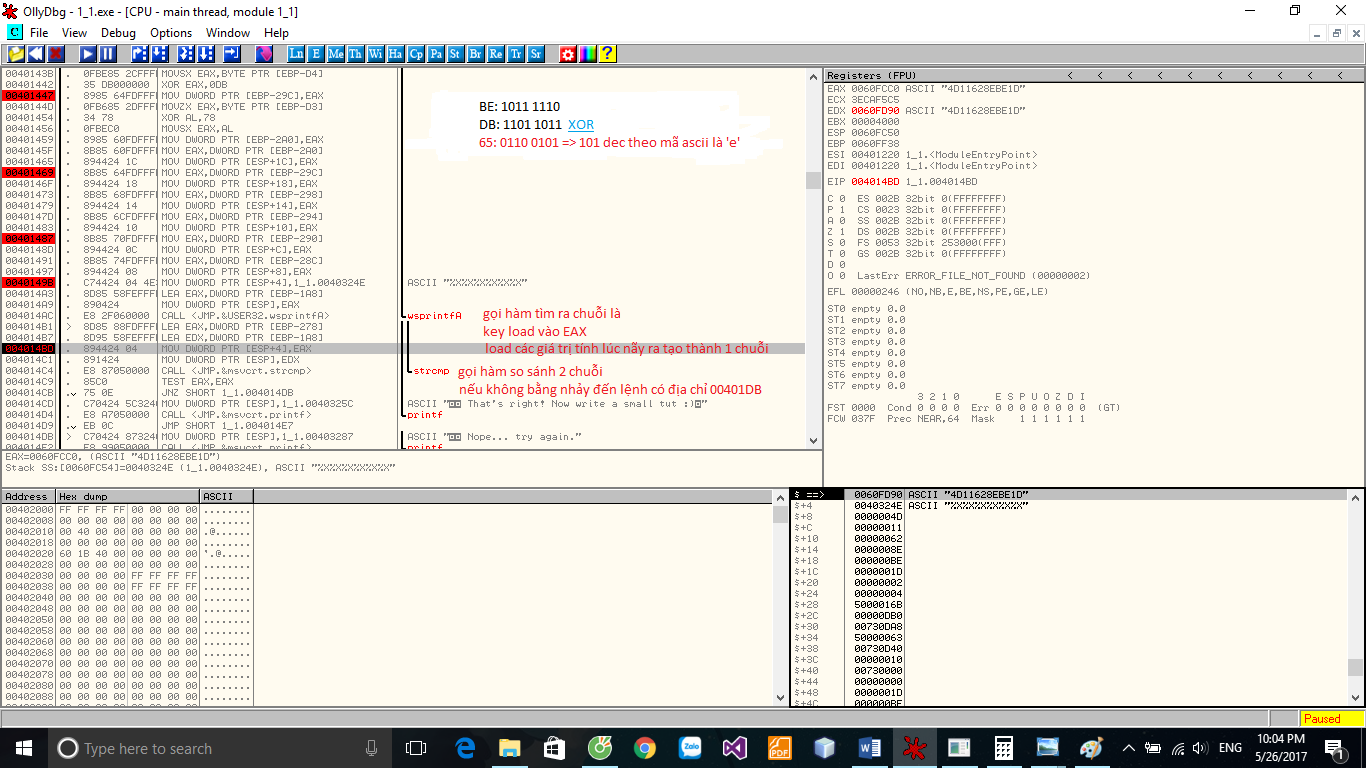
* Tiếp theo ta tìm 62 hex ta phải tìm x XOR 12:



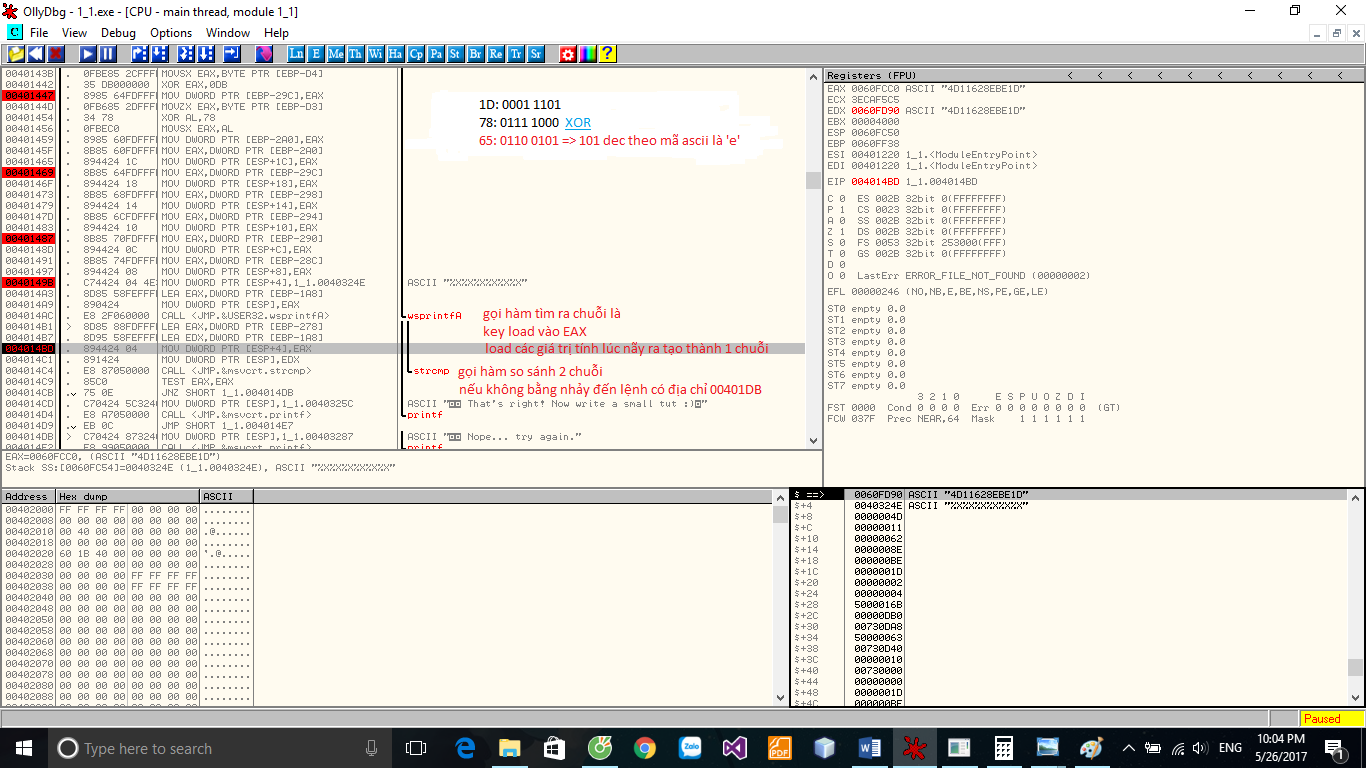
* Tiếp theo ta tìm 8E hex ta phải tìm x XOR FE:



* Tiếp theo ta tìm BE hex ta phải tìm x XOR DB:



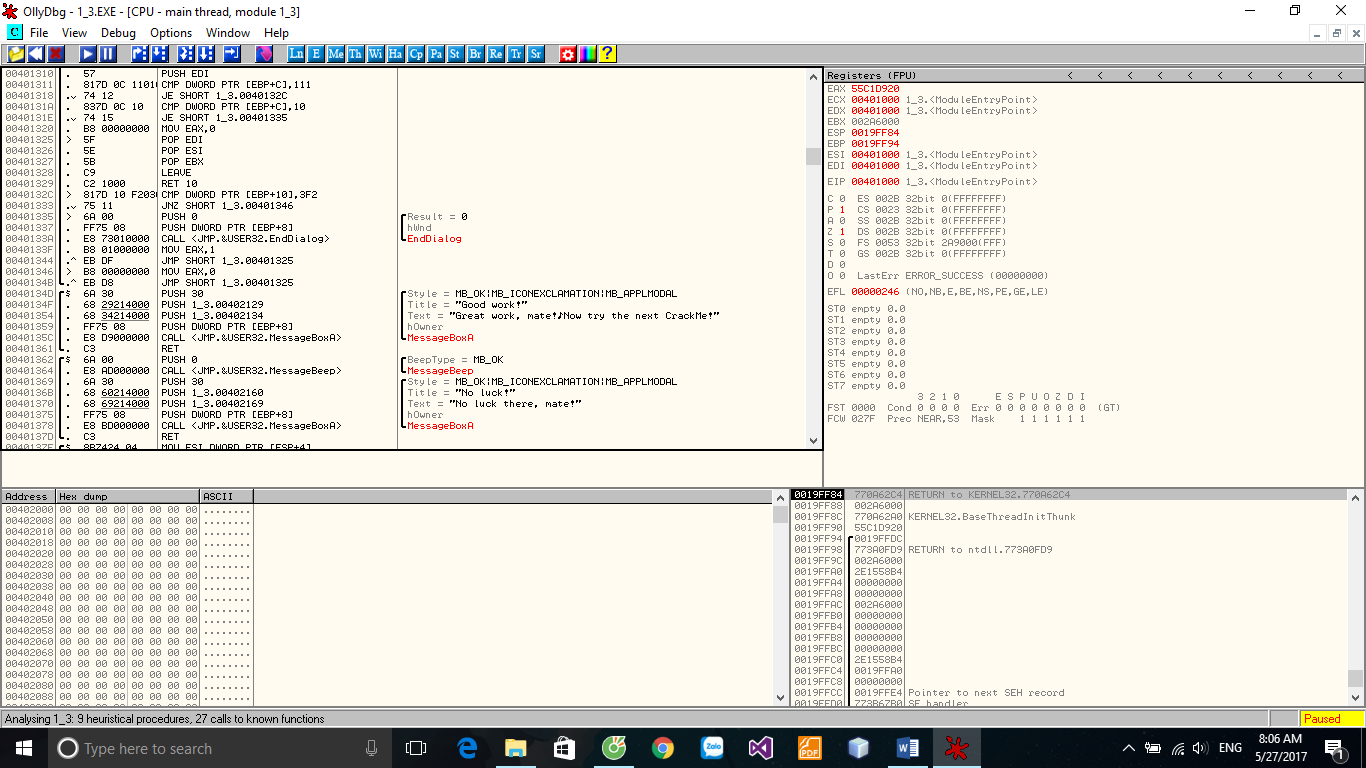
* Tiếp theo ta tìm 1D hex ta phải tìm x XOR 78:



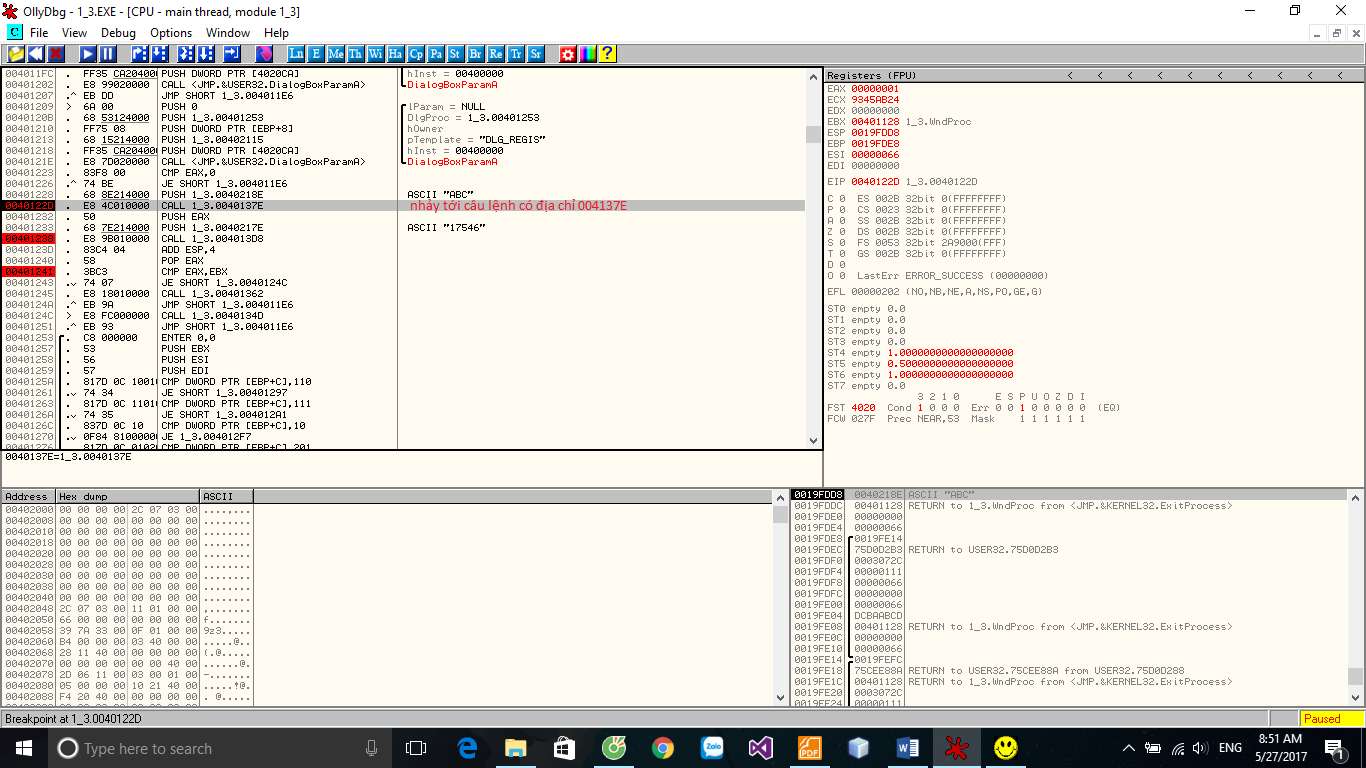
* Ráp lại ta sẽ tìm được key là: “yippee”

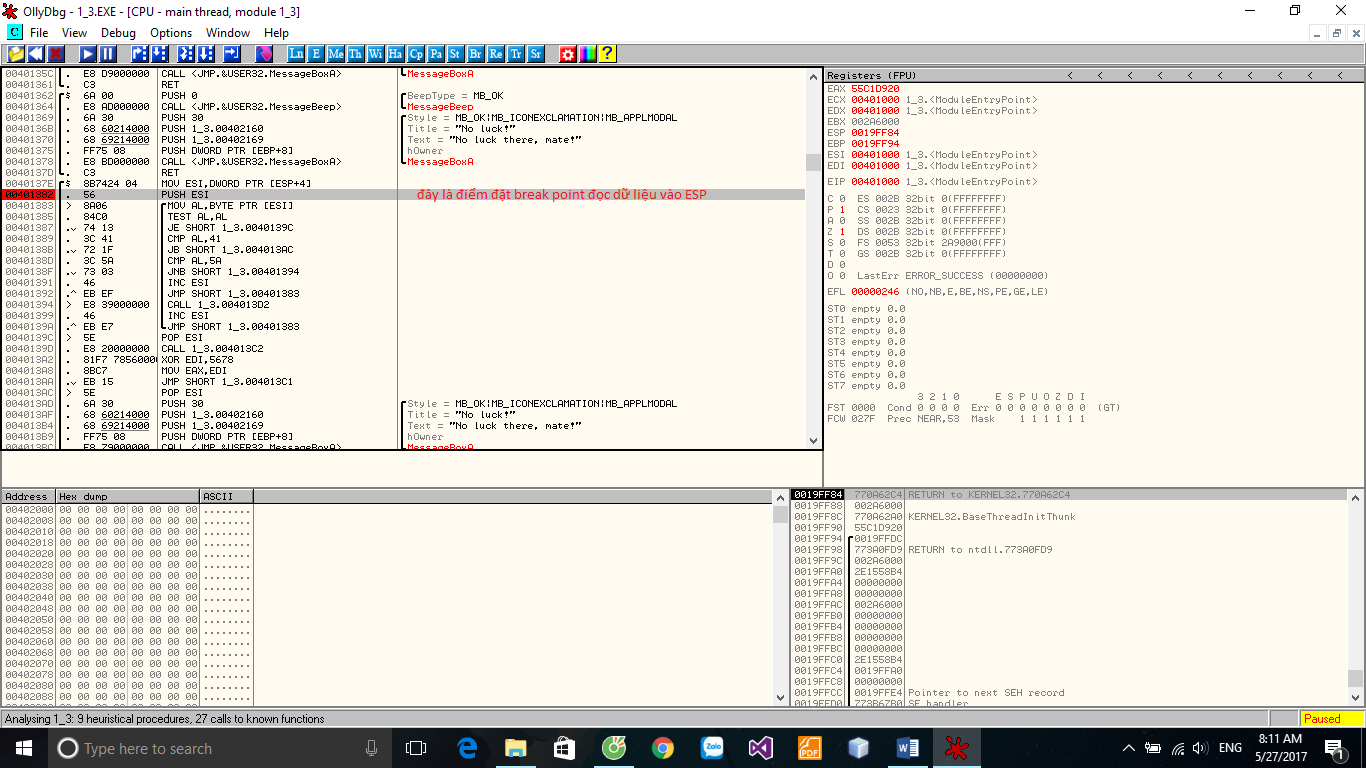
1. **Bài 2**
2. **Bài 3**

* Mở Olly lên, nhấn F3 (Open) (hoặc chọn qua Menu, nút bấm), sau đó chọn đến file cần crack và ấn f9 để chạy chương trình ta sẽ được kết quả như sau:

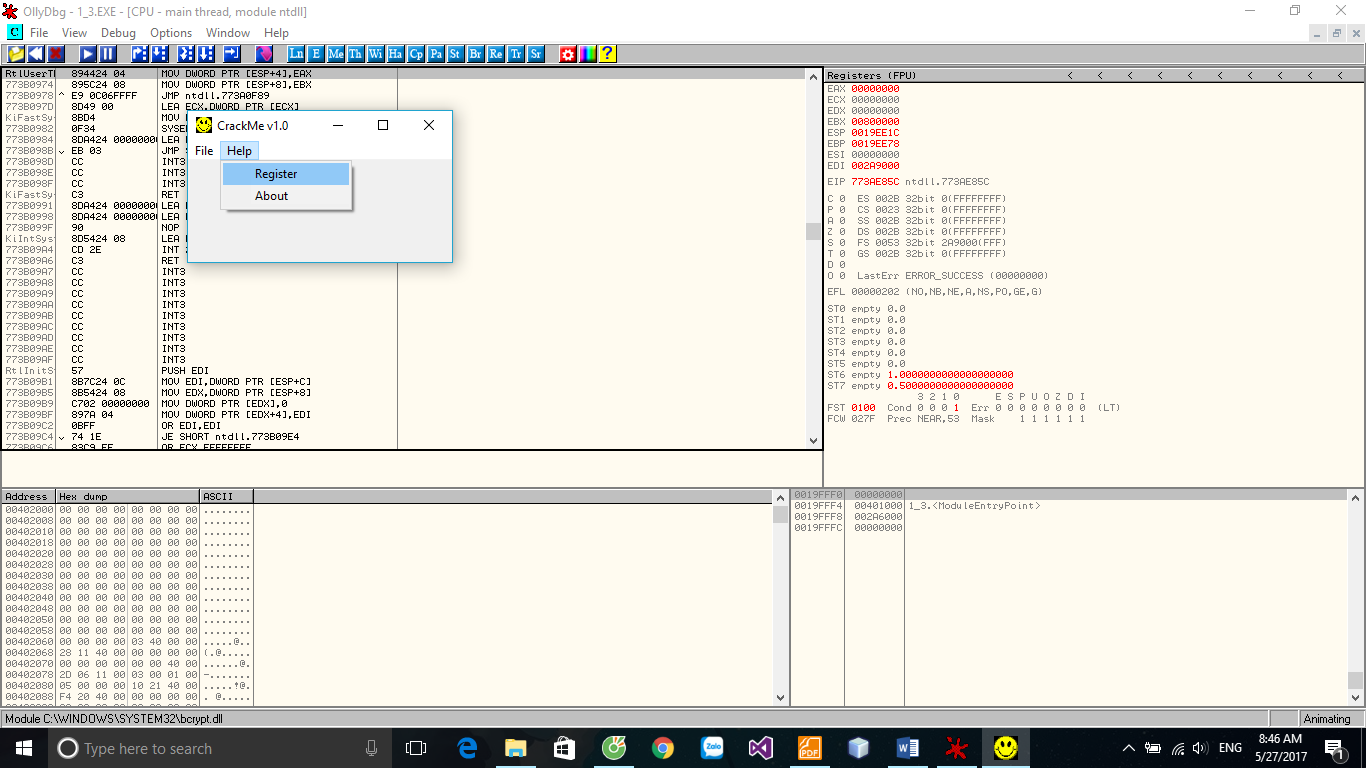
****

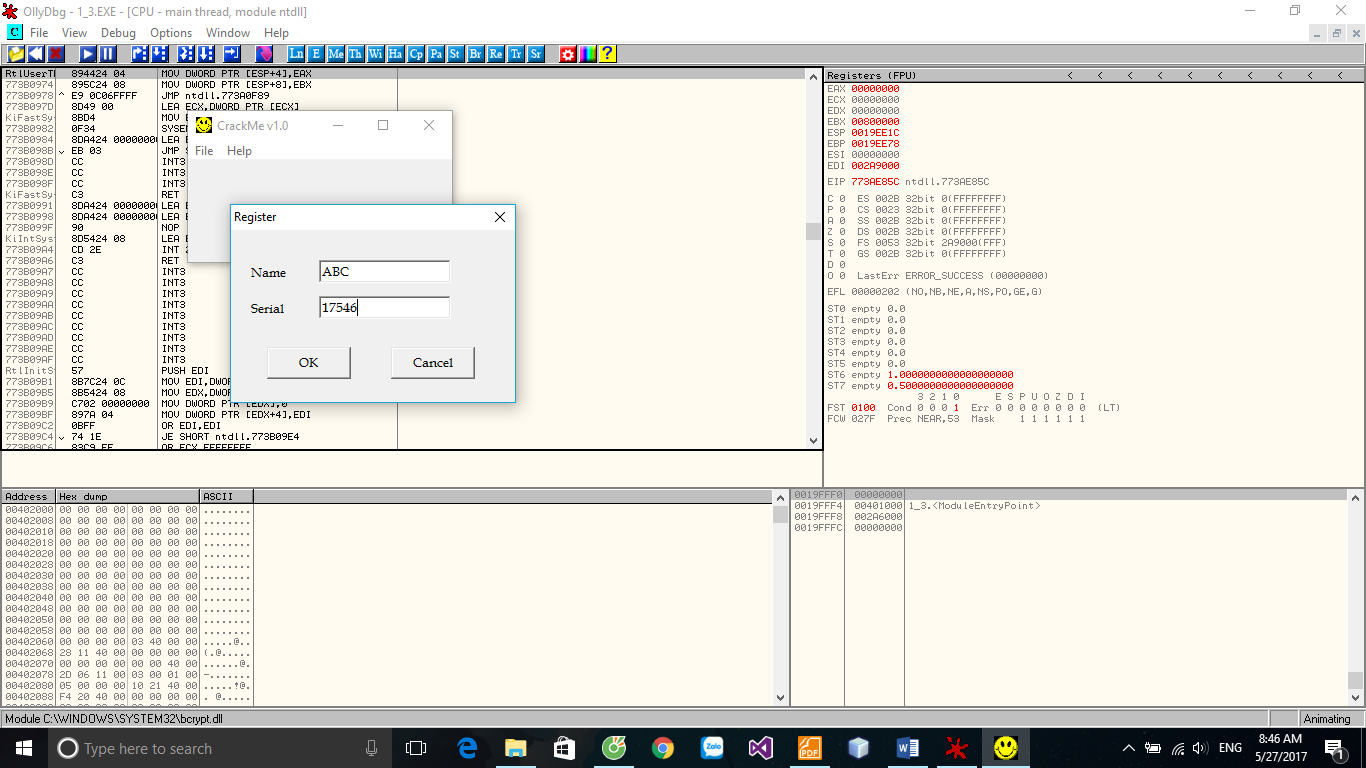
* Tiếp đến ta đặt break point tại đây để debug khi nhập dữ liệu vào và ấn Ctrl F7 để chạy đến bước nhập vào ESP là địa chỉ dữ liệu nguồn trong DS:

****

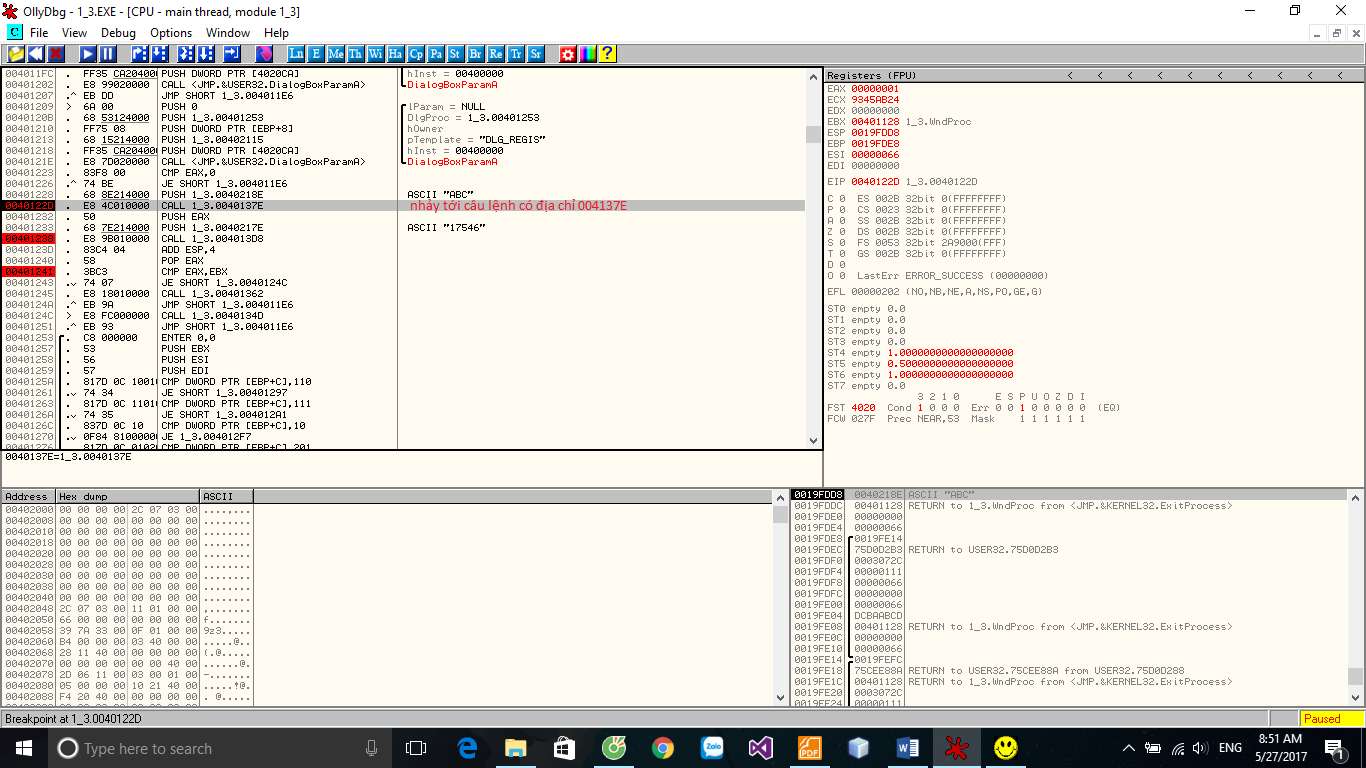
****

* Lúc này chương trình sẽ hiện lên bạn vào help\ register để nhập key với user name:

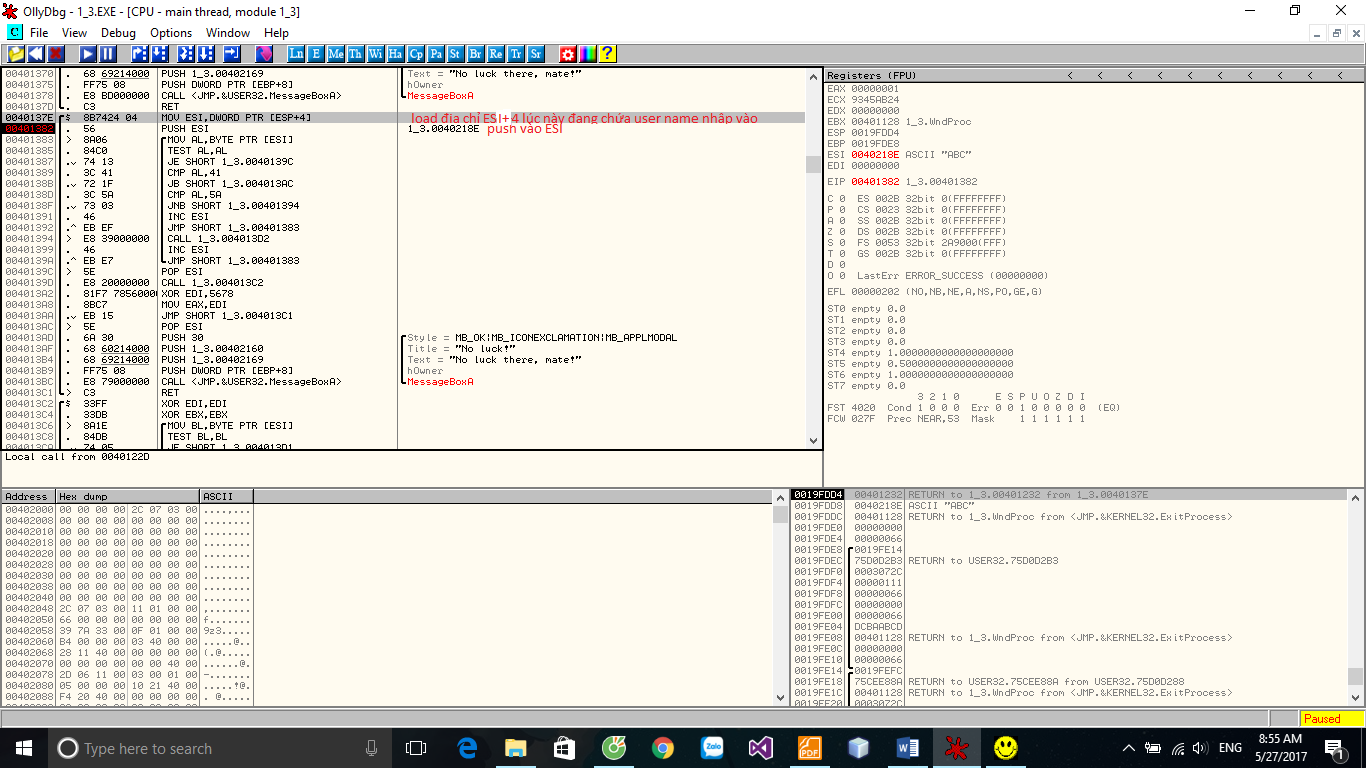


****

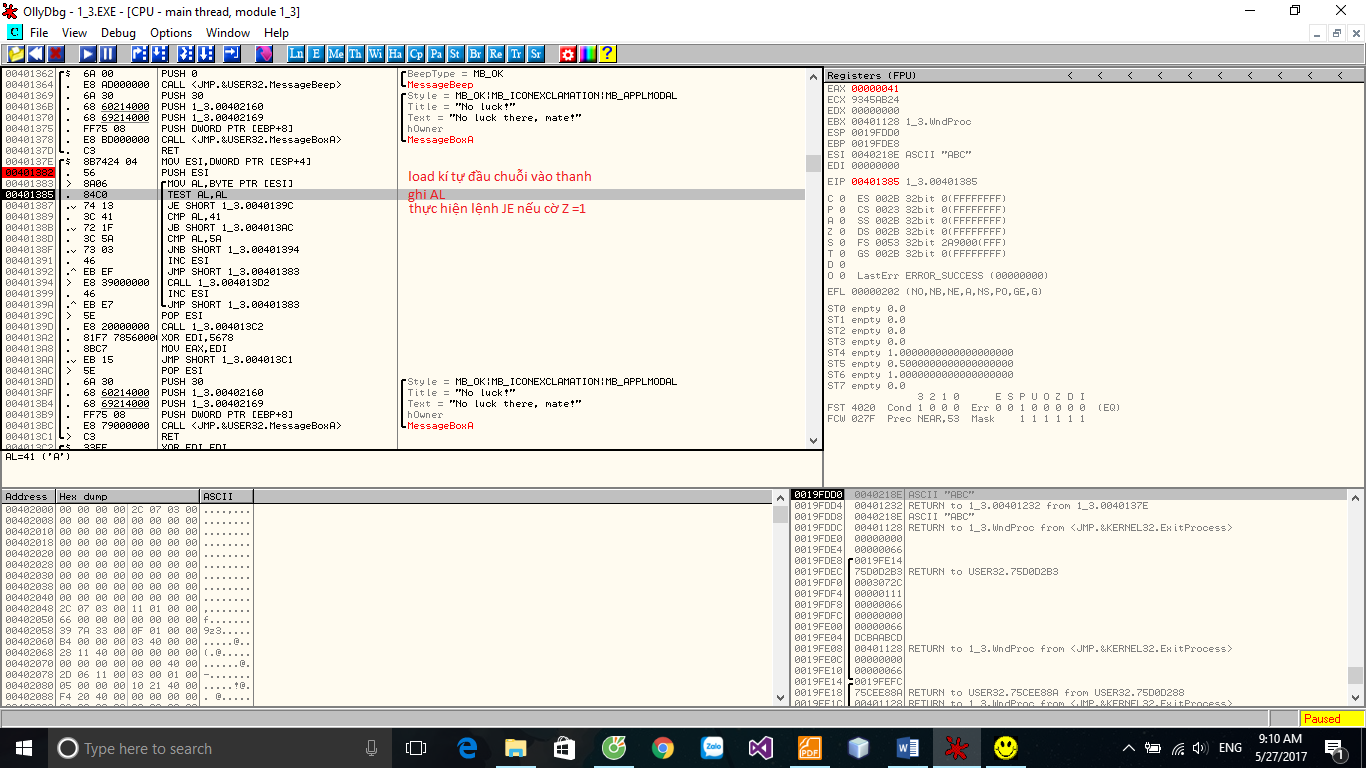
* Ở ví dụ này ta sẽ nhập user name là ABC và pass là 17546 để chạy thử tìm key
* Sau khi nhập tên vào lúc này Olly sẽ nhảy tới dòng lệnh mà ta đặt break point (phải đặt trước ở địa chỉ này trước khi debug)ở đây:

****

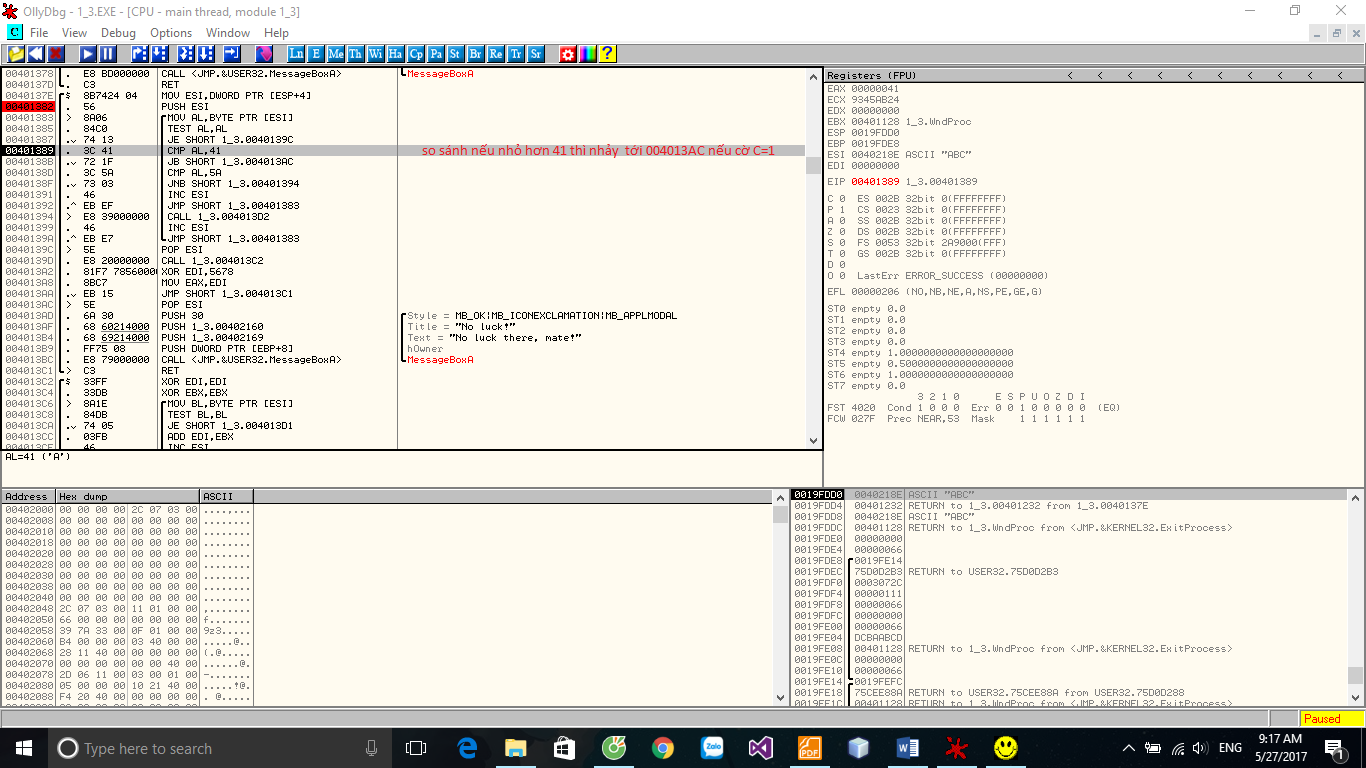
* Ta nhấn f7 để nhảy vào bên trong hàm để nhảy đến câu lệnh 004137E debug xem cách biến đổi của user name:

****

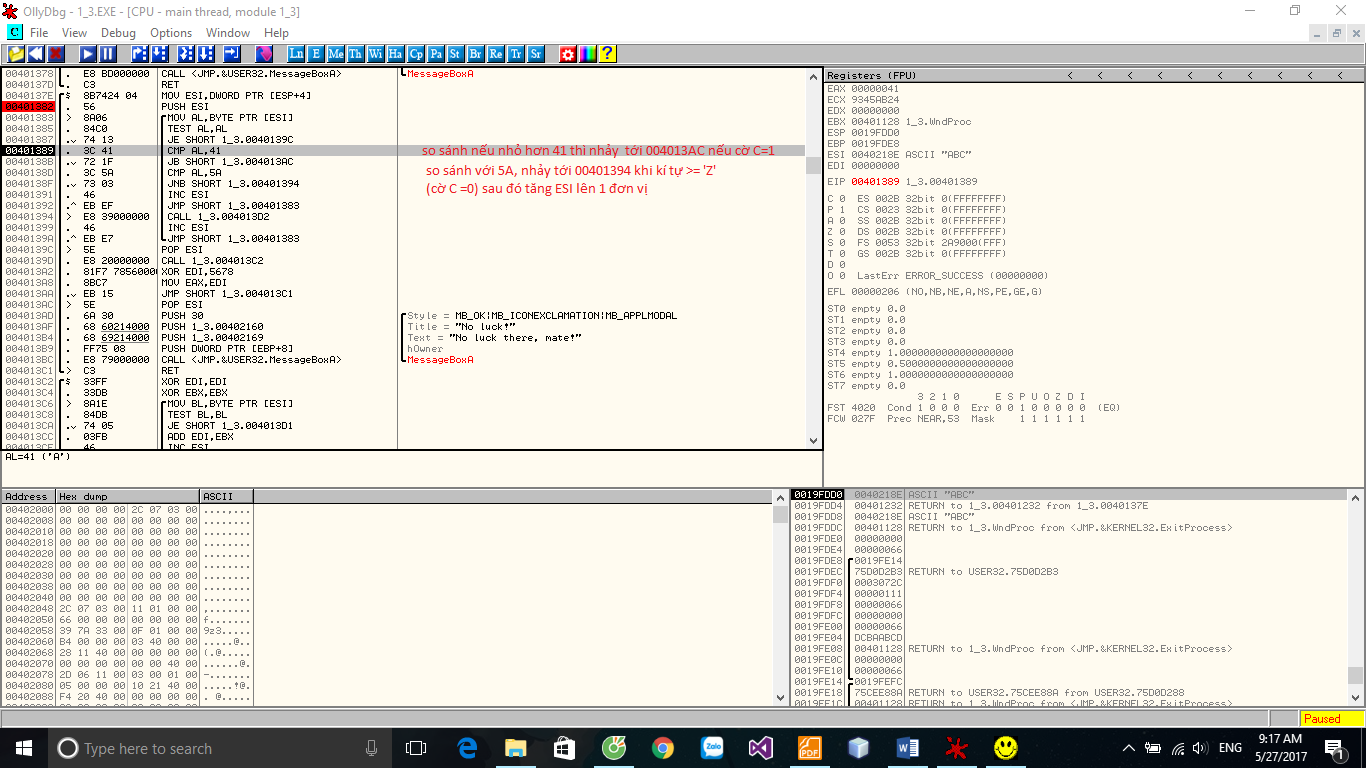
* Ta tiếp tục ấn F7, lúc này ta sẽ load kí tự đầu 1 byte(8 bit) của chuỗi vào thanh ghi AL và thực hiện lệnh JE nếu cờ Z = 1:



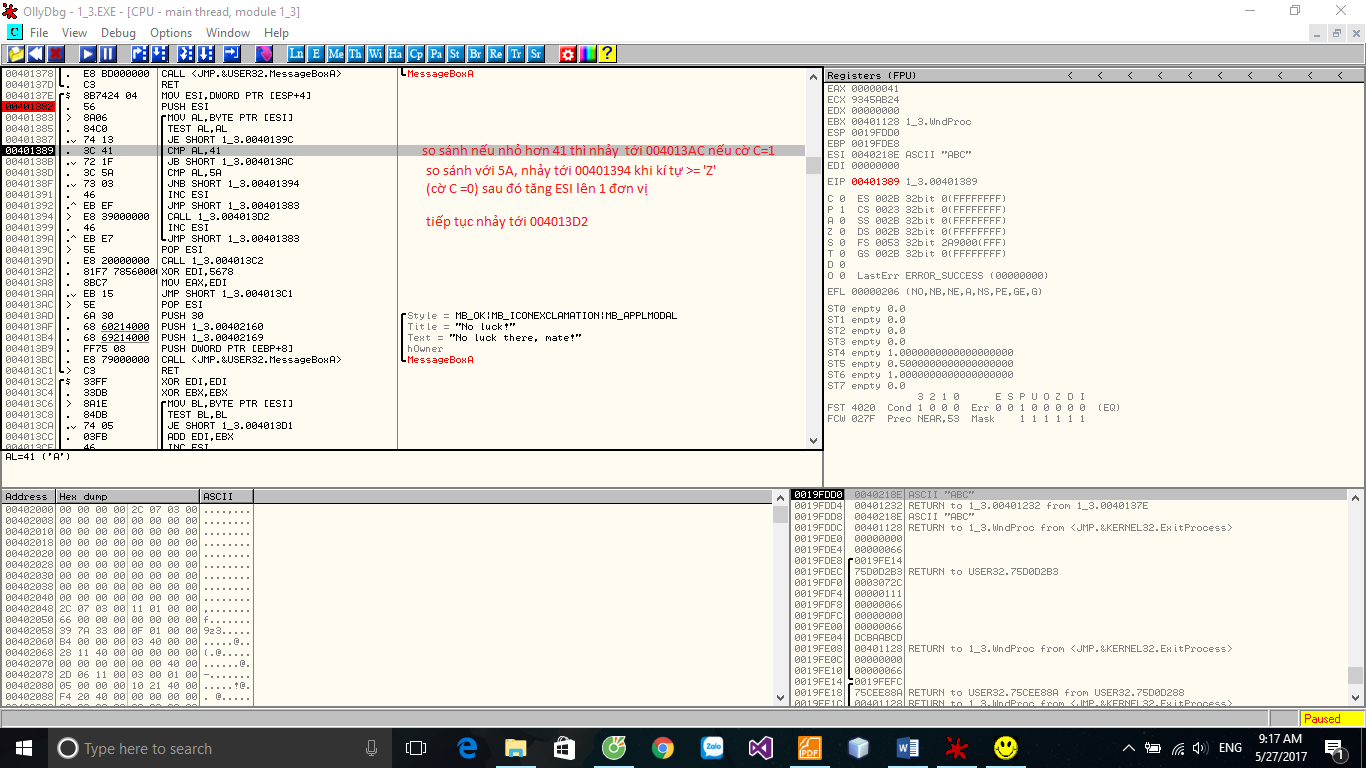
* Tiếp tục nó sẽ so sánh với 41 hex tương đương kí tự ‘A’ nếu nhỏ hơn sẽ nhảy tới câu lệnh có địa chỉ 004013AC:

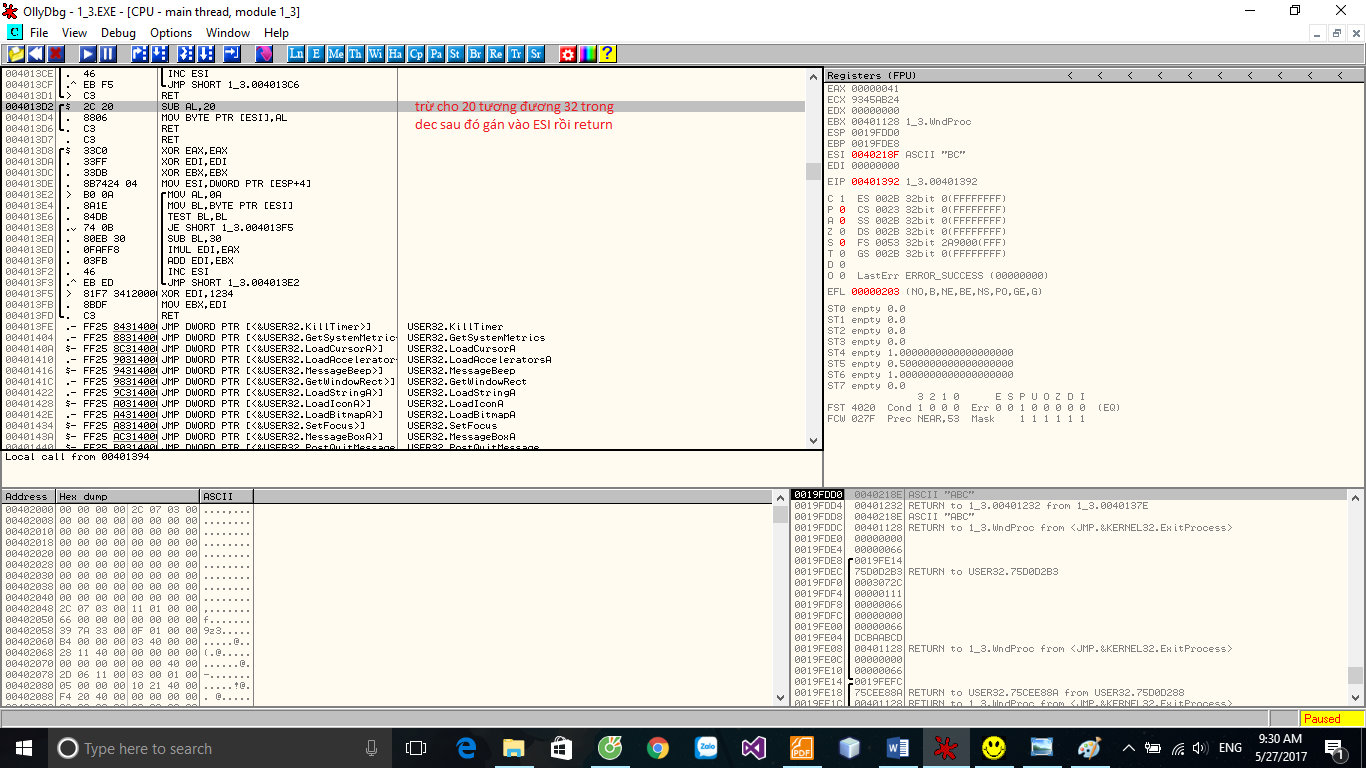


* So sánh tiếp với 5A hex tương đương kí tự ‘Z’ nếu lớn hơn sẽ nhảy tới câu lệnh có địa chỉ 00401394:

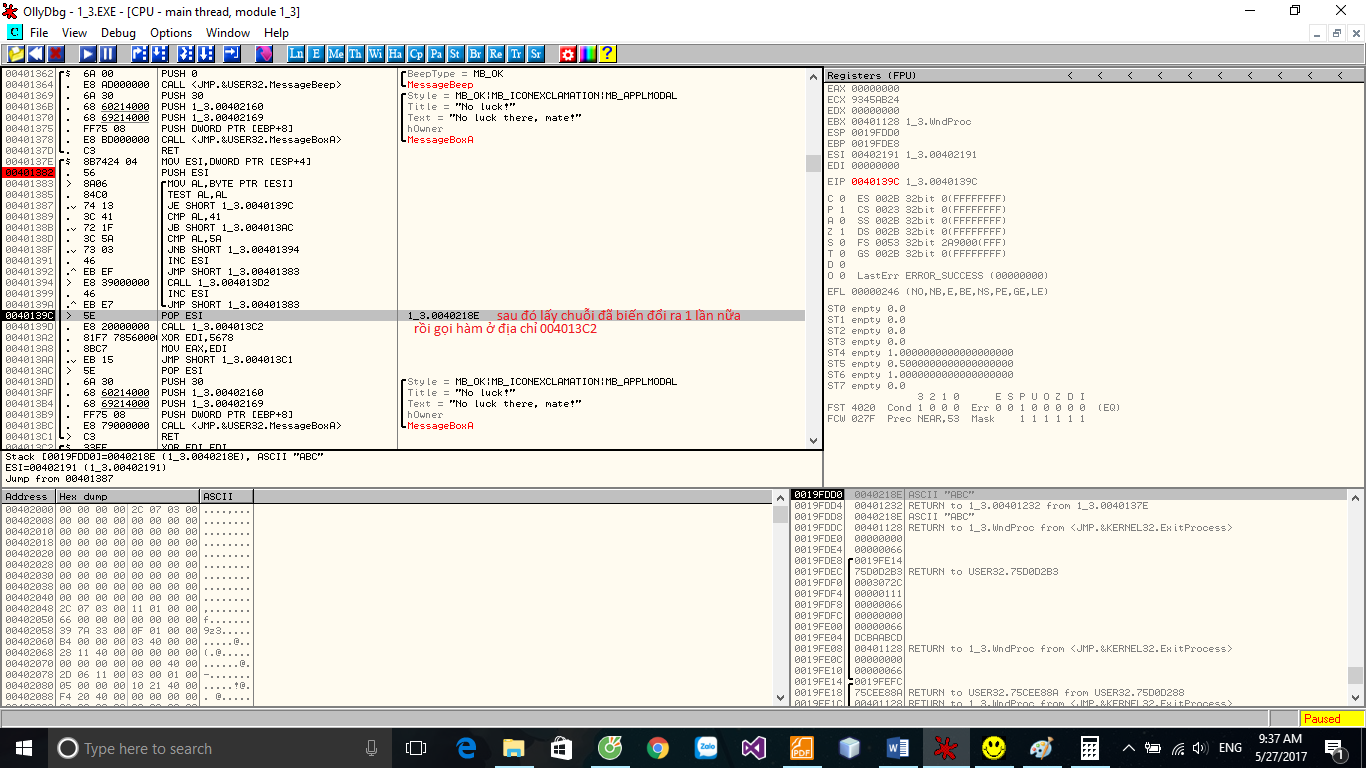
****

* Khi kí tự lớn hơn ‘Z’ ở câu lệnh 00401394 nó sẽ trừ cho 32 dec để chuyển kí tự thường thành hoa ngược lại sẽ jump về 00401383 để lặp lại tính các kí tự tiếp theo như vậy:

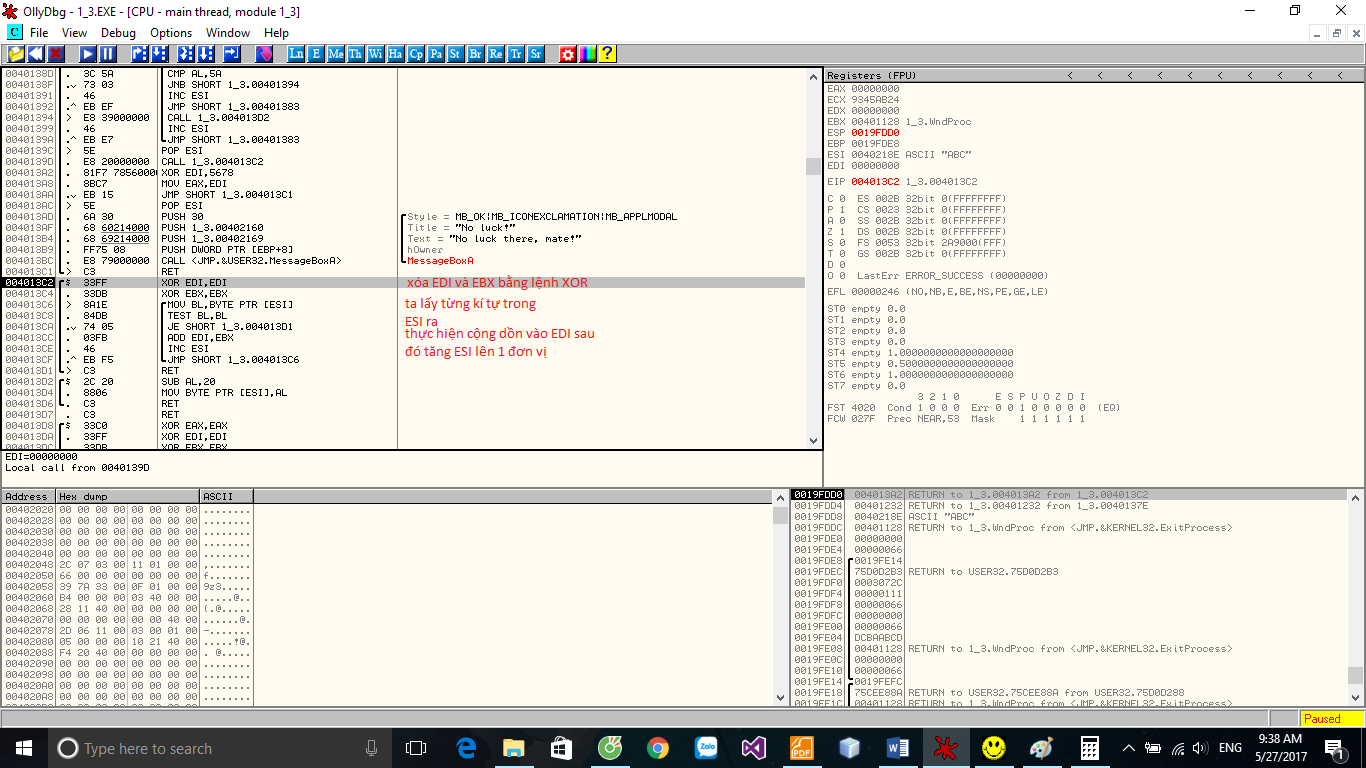
****

****

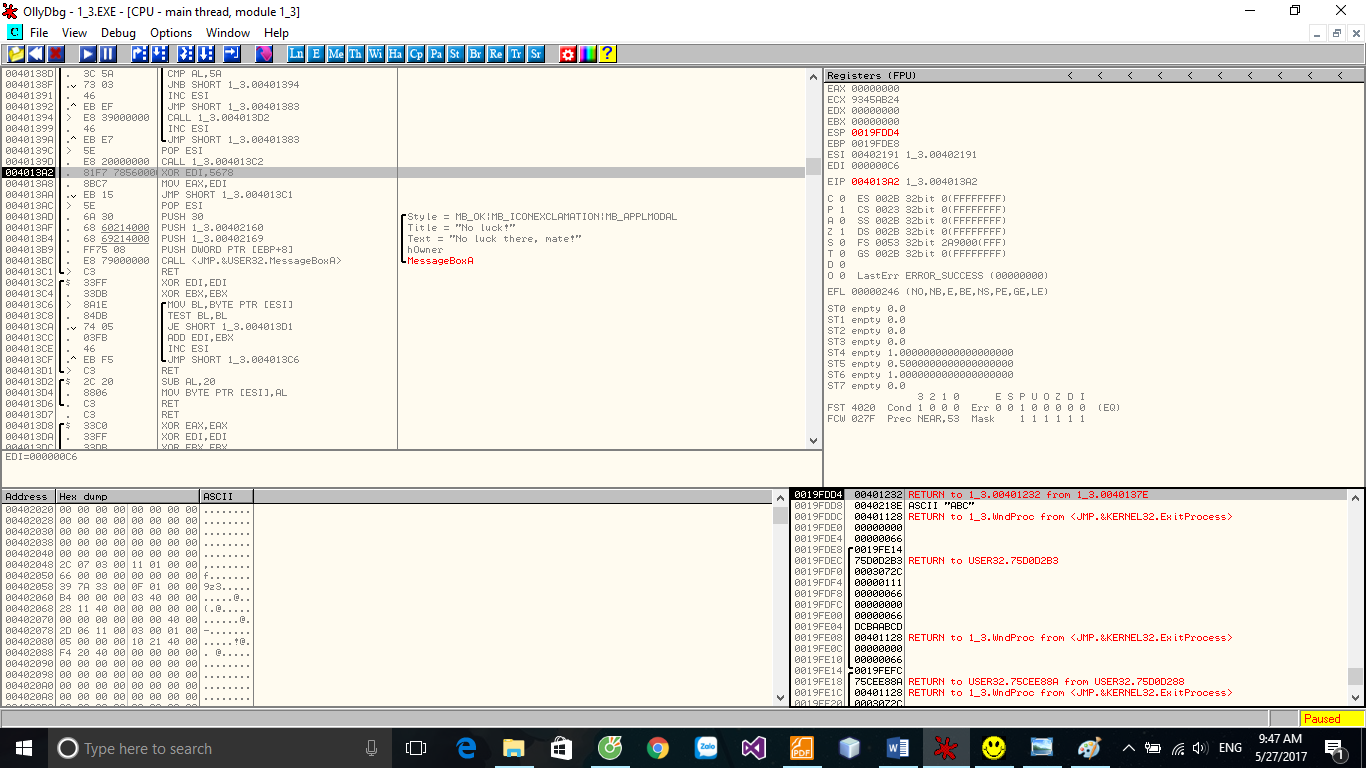
* Sau khi chuyển hết thành kí tự in hoa ta tiếp tục nhấn F7 tại đây:

****

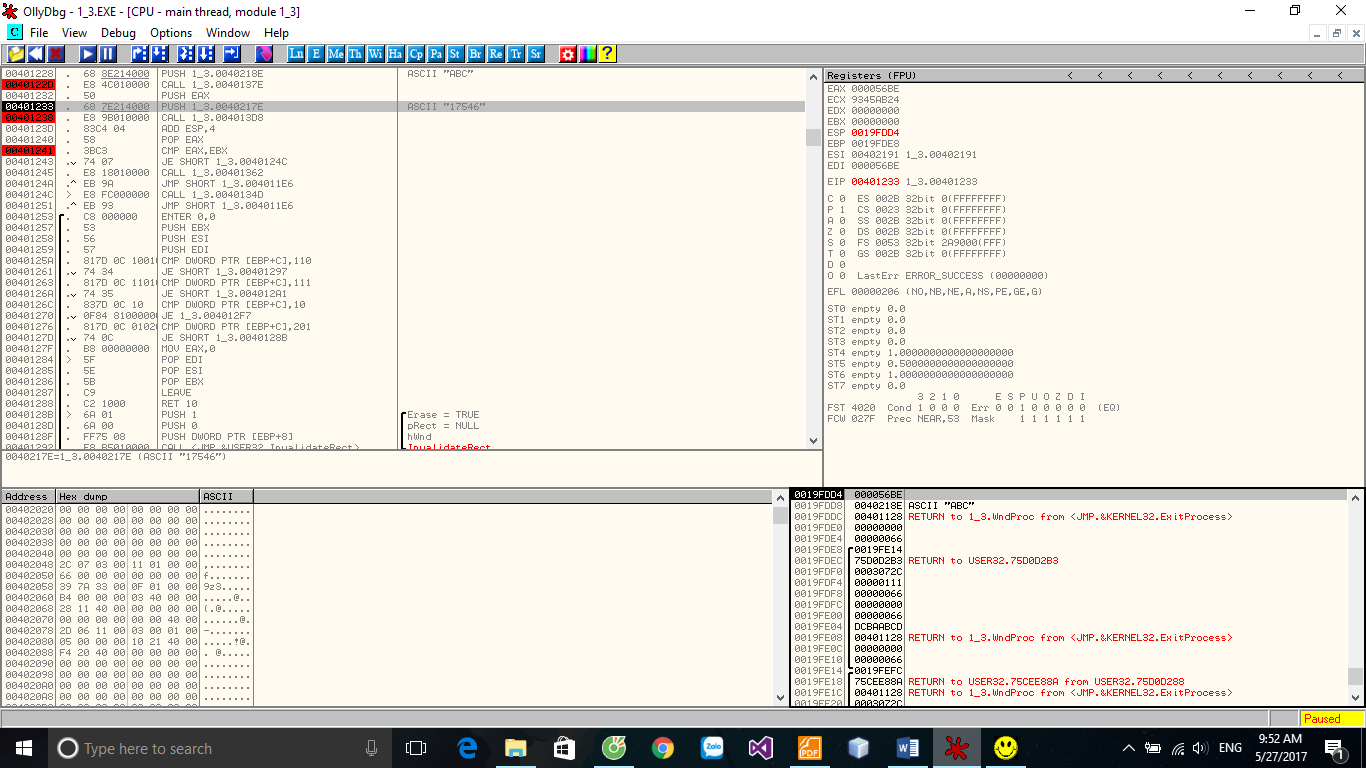
* Ở hàm lúc này ta ấn f7 để debug:

****

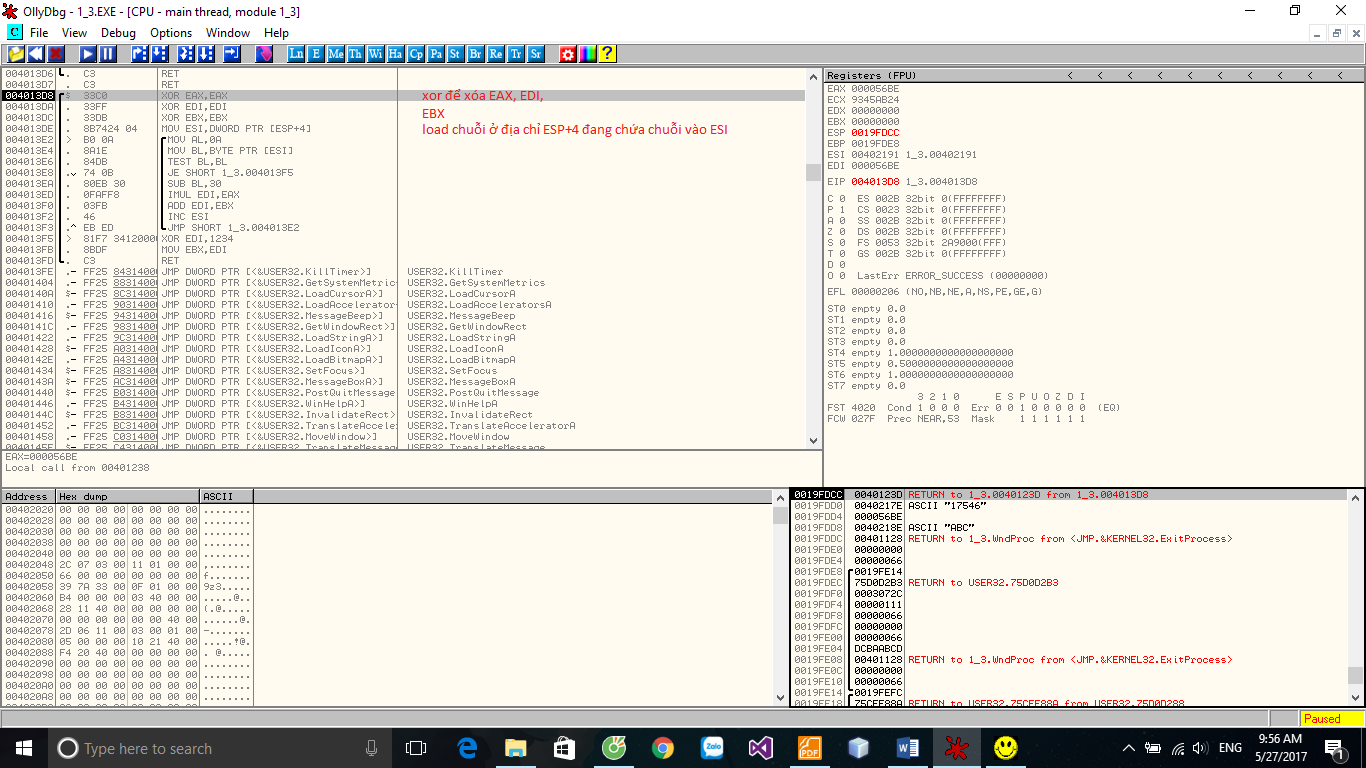
* Sau khi cộng dồn vào EDI hết các kí tự trong chuỗi nó sẽ return và nhảy đến lệnh XOR 5678 và cho ra kết quả là 56BE và gán vào EAX rồi jump đến lệnh ở địa chỉ 004013C1là RET:

****

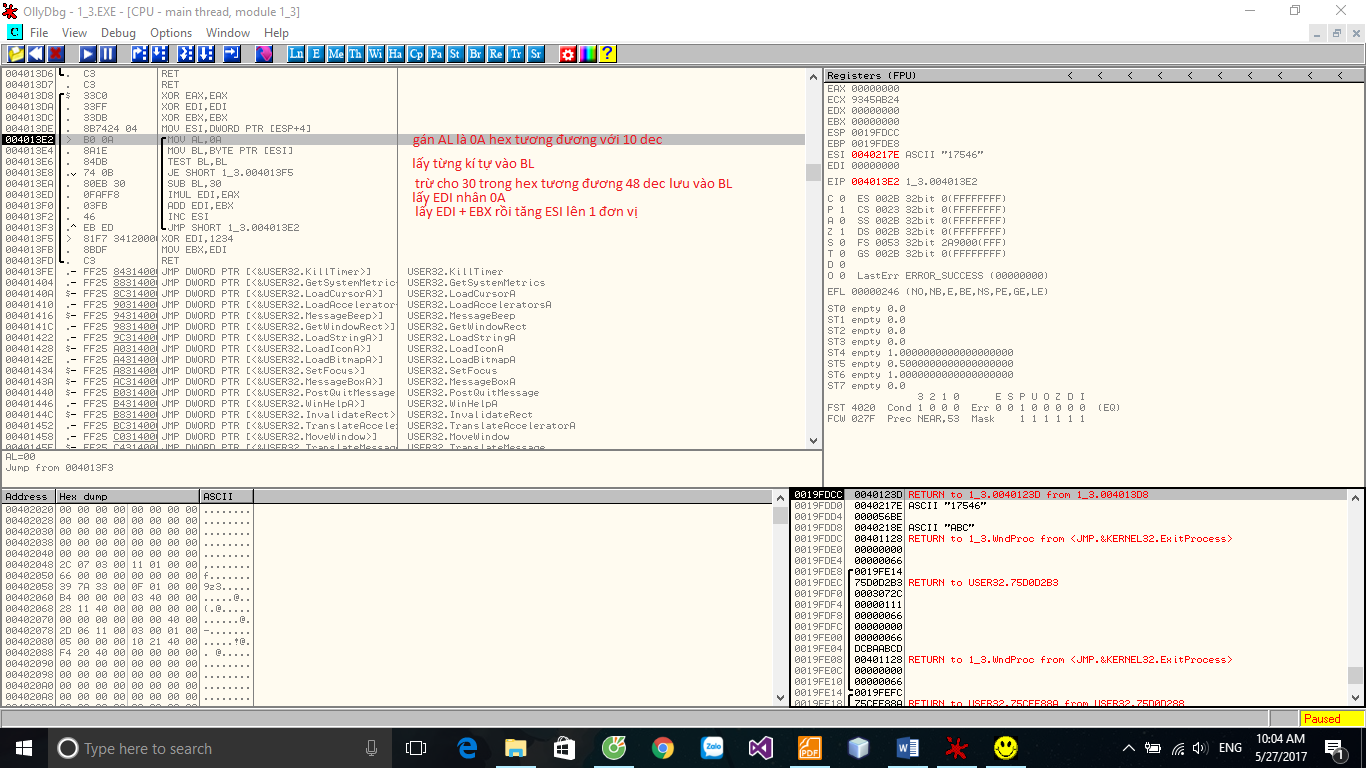
* Lúc này ta push EAX vào và ta tiếp tục nhấp F7 để vào lệnh call tiếp theo:

****

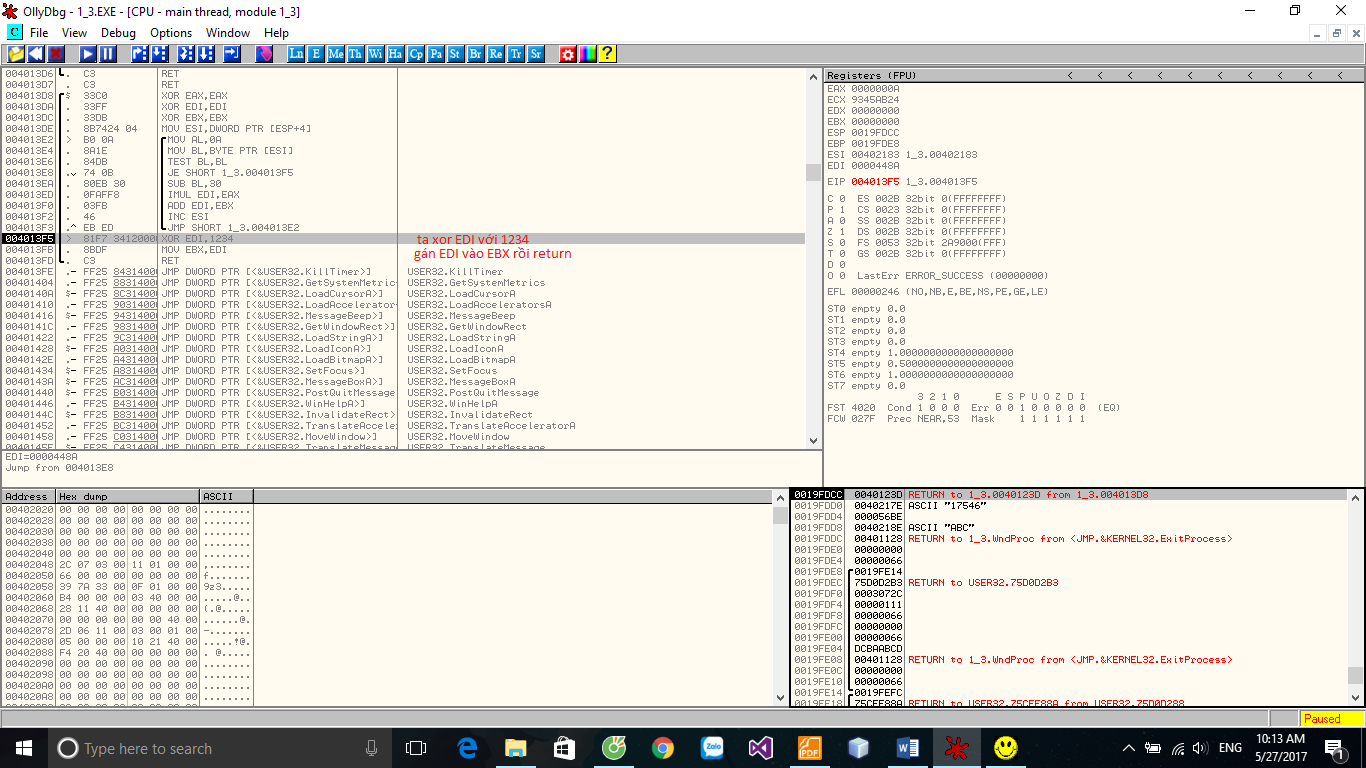
* Sau khi nhấp nó sẽ nhảy đến câu lênh 00413D8:



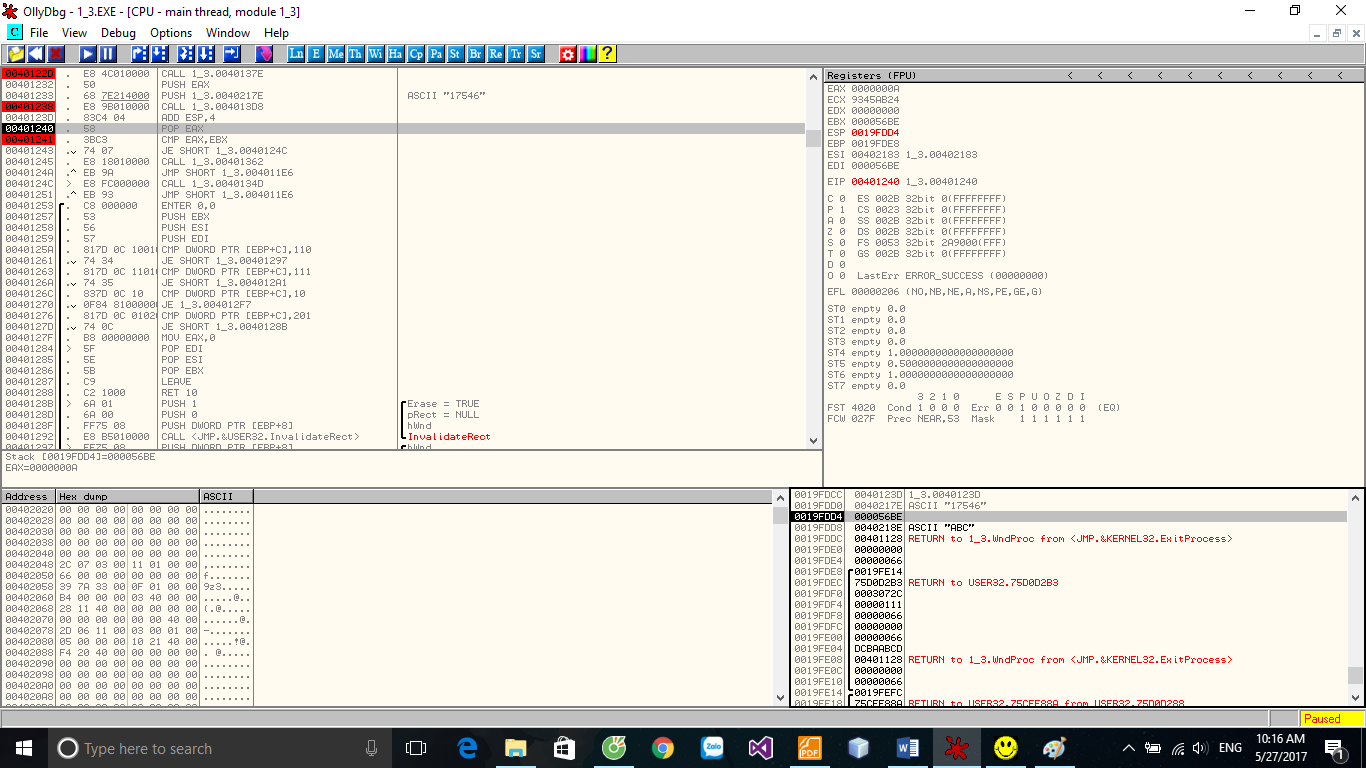
* Ta tiếp tục ấn F7, lúc này ta sẽ load kí tự đầu 1 byte(8 bit) của chuỗi vào thanh ghi BL:

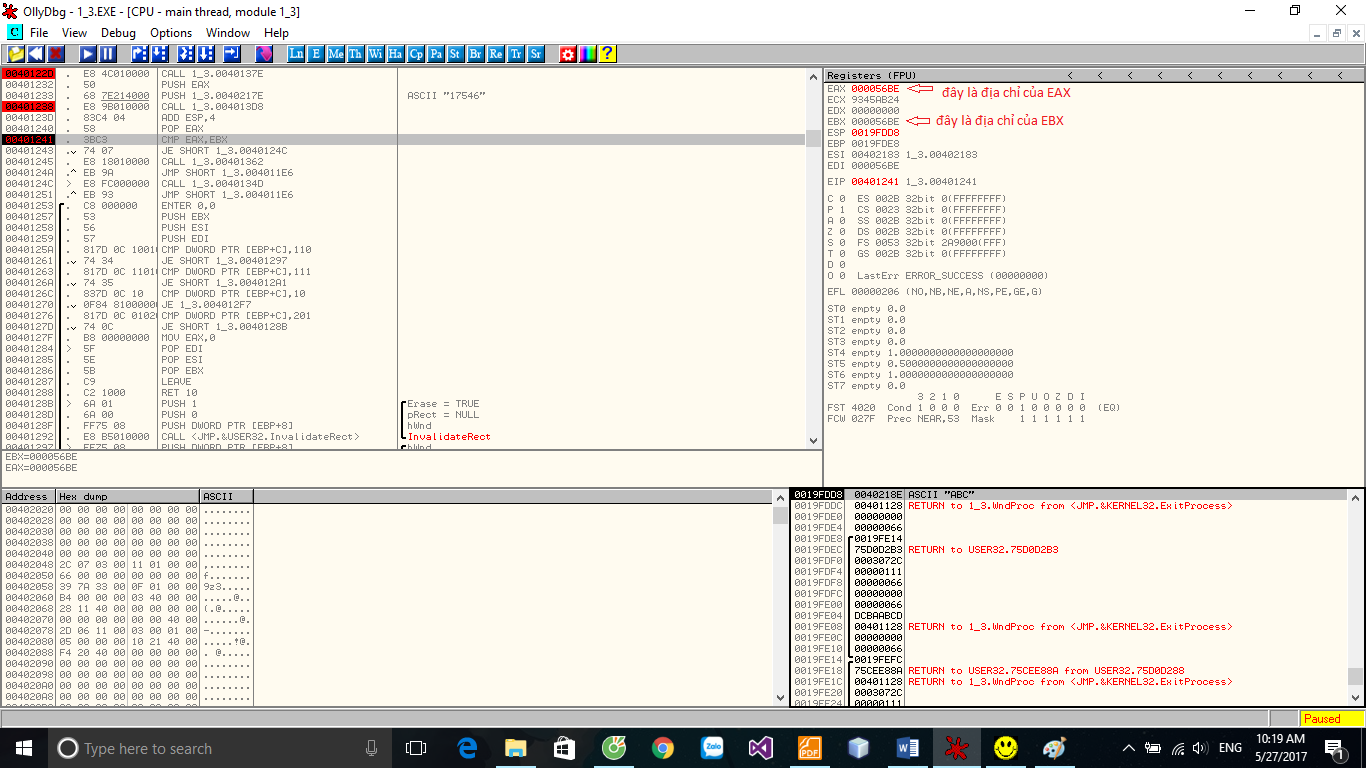


* Ta làm liên tục như vậy cho đến khi hết chuỗi và giá trị của EDI lúc này là 448A rồi ta XOR với 1234 lưu dô EBX rồi return:

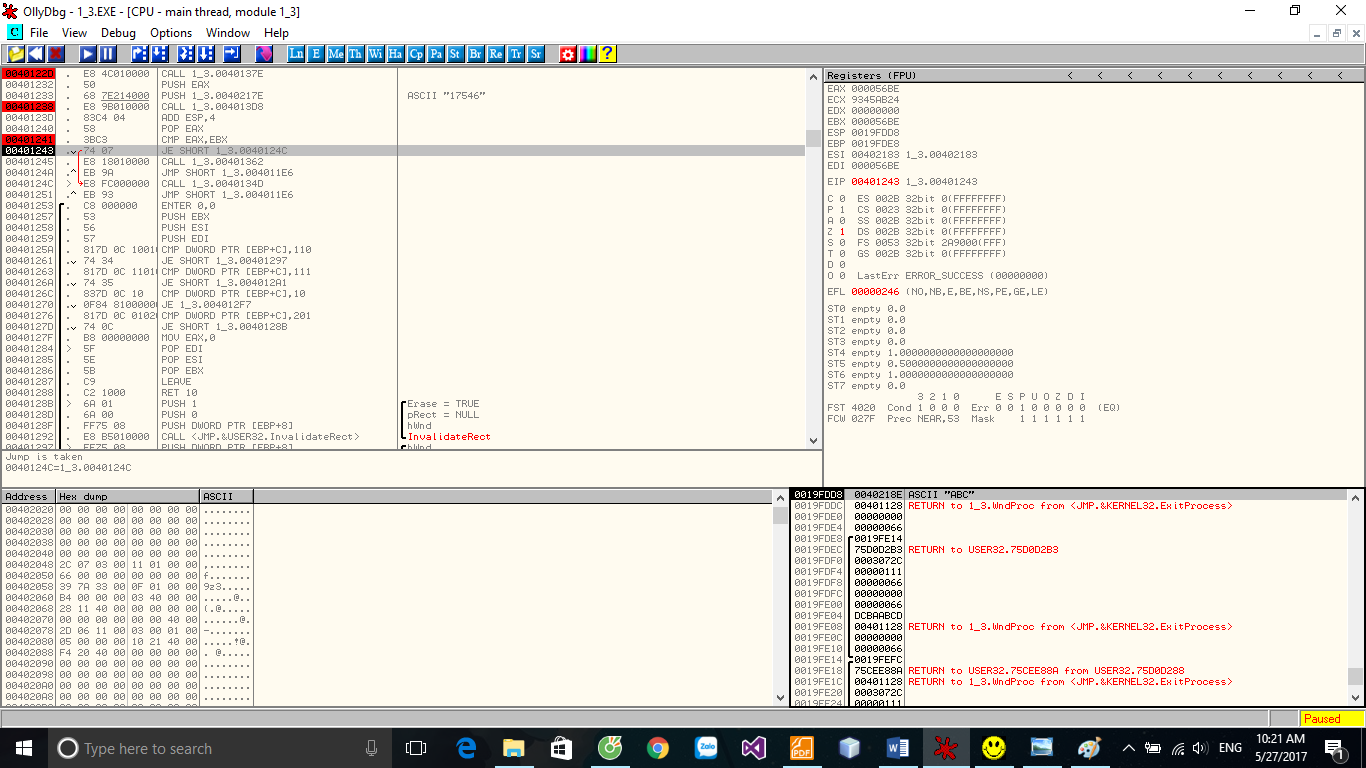


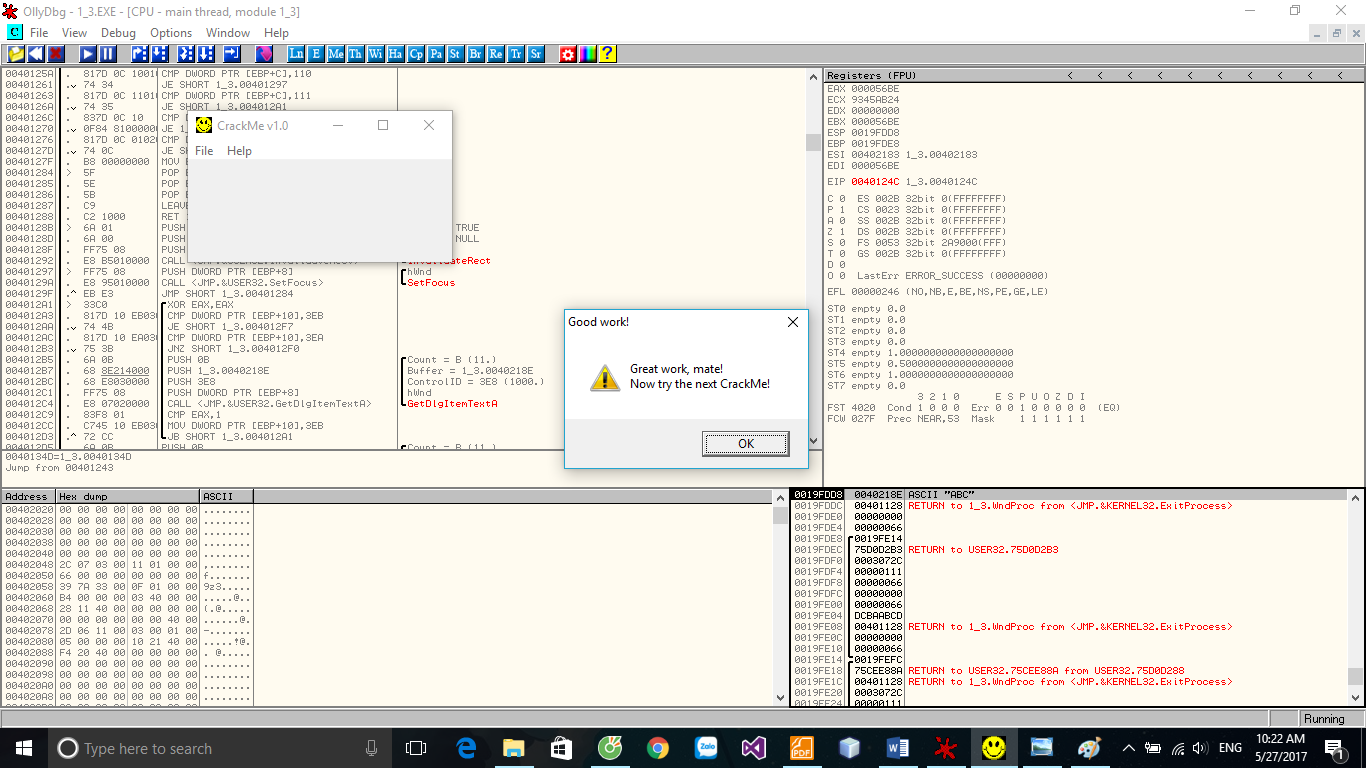
* Sau đó ta pop EAX từ đỉnh stack ra và so sánh với EBX:



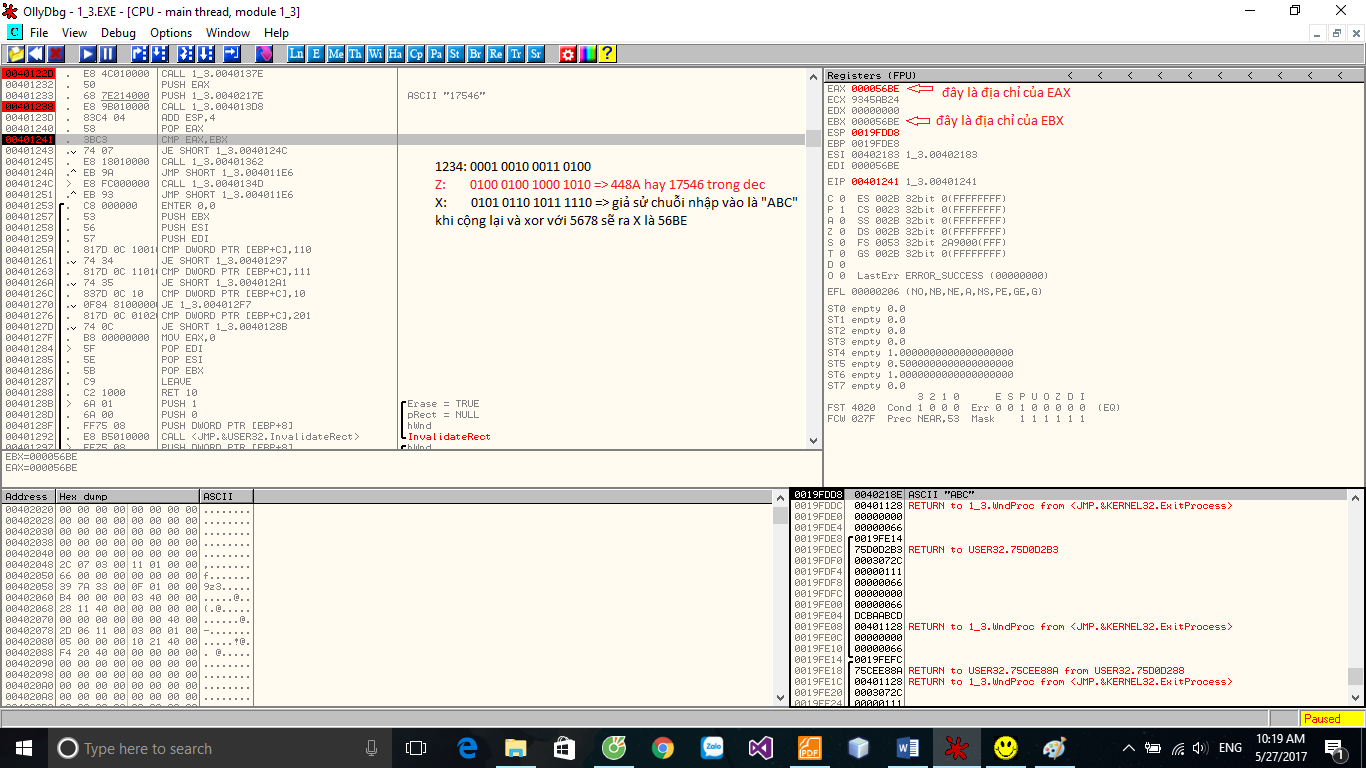


* Nếu 2 địa chỉ này trùng nhau nó sẽ nhảy đến 0040124C và nhảy đến hàm 0040134D và thông báo thành công





* Quy luật tìm key bằng cách viết keygen:
* Đầu tiên ta phải nhập vào chuỗi cộng hết các số theo hệ thập phân lại nếu là kí tự thường thì chuyển thành hoa theo công thức: s[i] = (s[i] >’Z’) ? s[i] – 32 : s[i];
* Theo quá trình debug ta nhận thấy phải cộng hết các kí tự trong chuỗi lại với nhau thành 1 số rồi xor với 5678( chuyển sang thập phân để tính đơn giản là 22136) sẽ tạo ra số X
* X so sánh với Z = Y XOR 1234( chuyển sang thập phân để tính đơn giản là 4660)
* Vậy để tìm ra Y là key ta tìm các bit khi xor để Z=X sau đây là 1 ví dụ:



* Ta nhận thấy lúc key nhập vào bị chuyển bằng các tính ra từng số nhân 10 và cộng dồn nên key chắc chắn là 17546 như ví dụ trên và từ đó ta viết keygen
* Lưu ý: trong quá trình chạy và test em nhận thấy số kí tự của user name  
  nhập vào tối đa là 10 vì chuỗi chỉ chứa được 10 kí tự

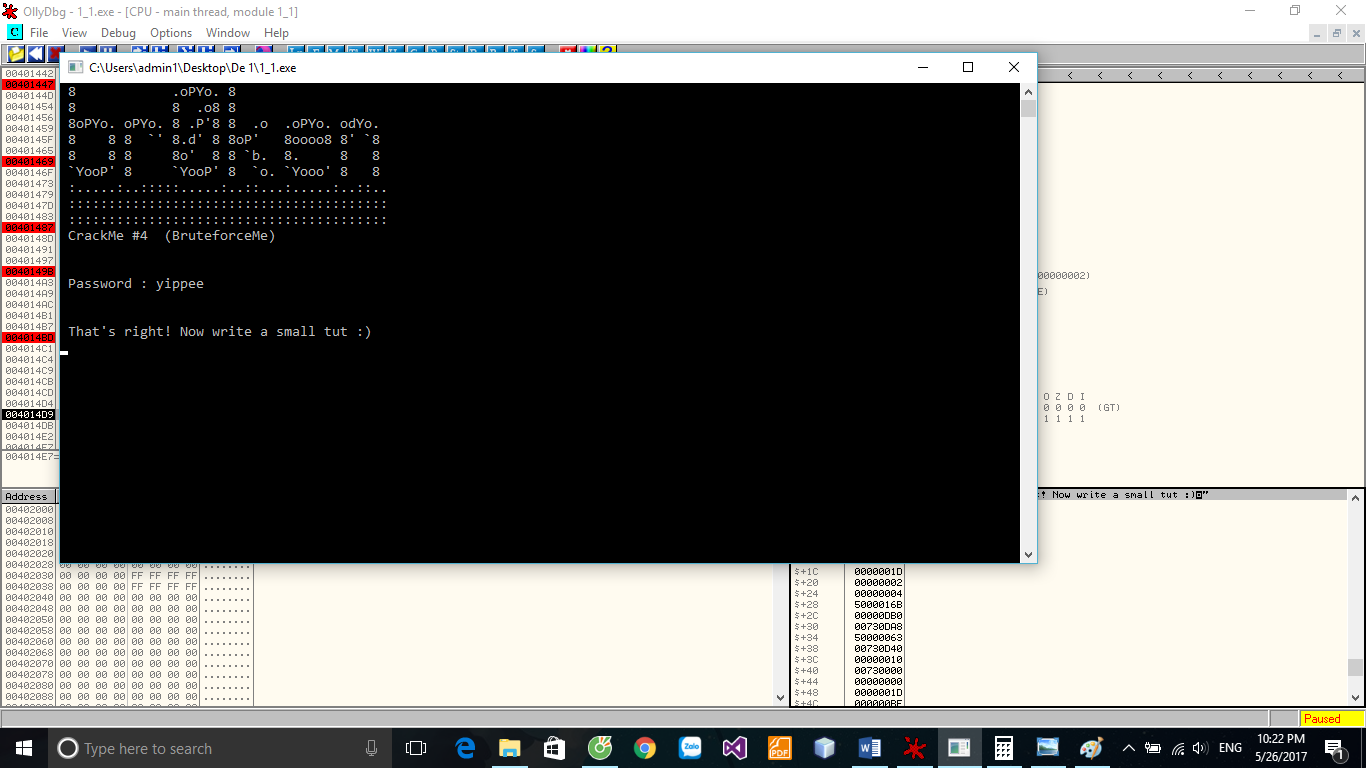
## BẢNG PHÂN CÔNG CÔNG VIỆC

|  |  |  |
| --- | --- | --- |
| **Trần Duy Phương** | **Dương Vinh Quang** | **Nguyễn Đăng Quang** |
| Crack bài 3 | Viết keygen cho bài 3 và làm bài 2 | Crack bài 1 |
|  |  |  |

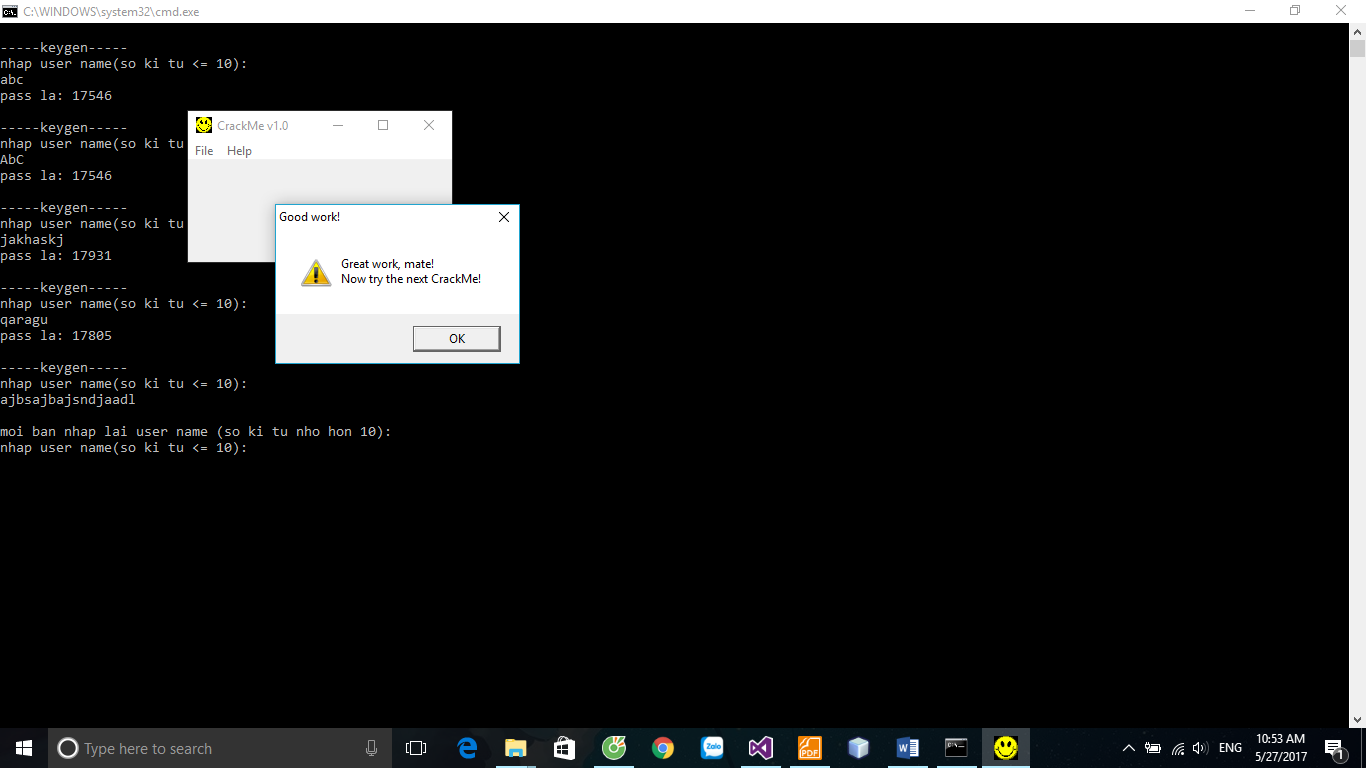
* Trong quá trình sẽ có sự trợ giúp lẫn nhau và họp nhóm cùng nhau viết báo cáo

# CHẠY KIỂM TRA VÀ CHỤP HÌNH

* Bài 1:



* Bài 3:



# CÁC CHỨC NĂNG LÀM ĐƯỢC VÀ CHƯA LÀM ĐƯỢC

## Đã làm được

* Hoàn thành bài 1
* Hoàn thành bài 3 và viết keygen

## Chưa làm được

* Bài 2 nhóm em chưa làm được

# IV. ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH ĐỒ ÁN:75%

# CÁC TÀI LIỆU THAM KHẢO

* Các file tài liệu của thầy cô được đăng trên moodle