**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN**

**KHOA CÔNG NGHỆ THÔNG TIN**

**BÁO CÁO ĐỒ ÁN 1**

**Bạch Tăng Thắng - 18120237**

**Trần Thanh Tùng - 18120258**

**Trần Hữu Chí Bảo - 18120288**

**Vòng Cảnh Chi - 18120293**

**Cao Tất Cường - 18120296**

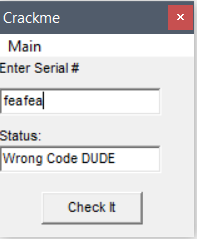
**Môn: Kiến trúc máy tính – Hợp ngữ**

**Thành phố Hồ Chí Minh - 2020**

1. Quá trình thực hiện:

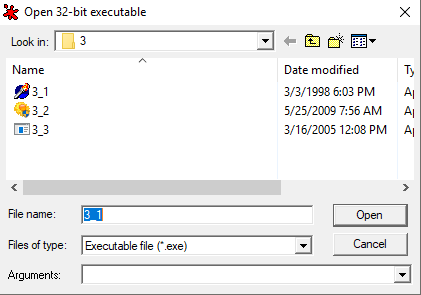
Bài 3.1:

* Đầu tiên, mở crackme lên sau đó nhập vào một serial (ngẫu nhiên). ta thấy nó xuất hiện dòng chữ “Wrong Code DUDE” 🡪 tức Serial đó không chính xác.

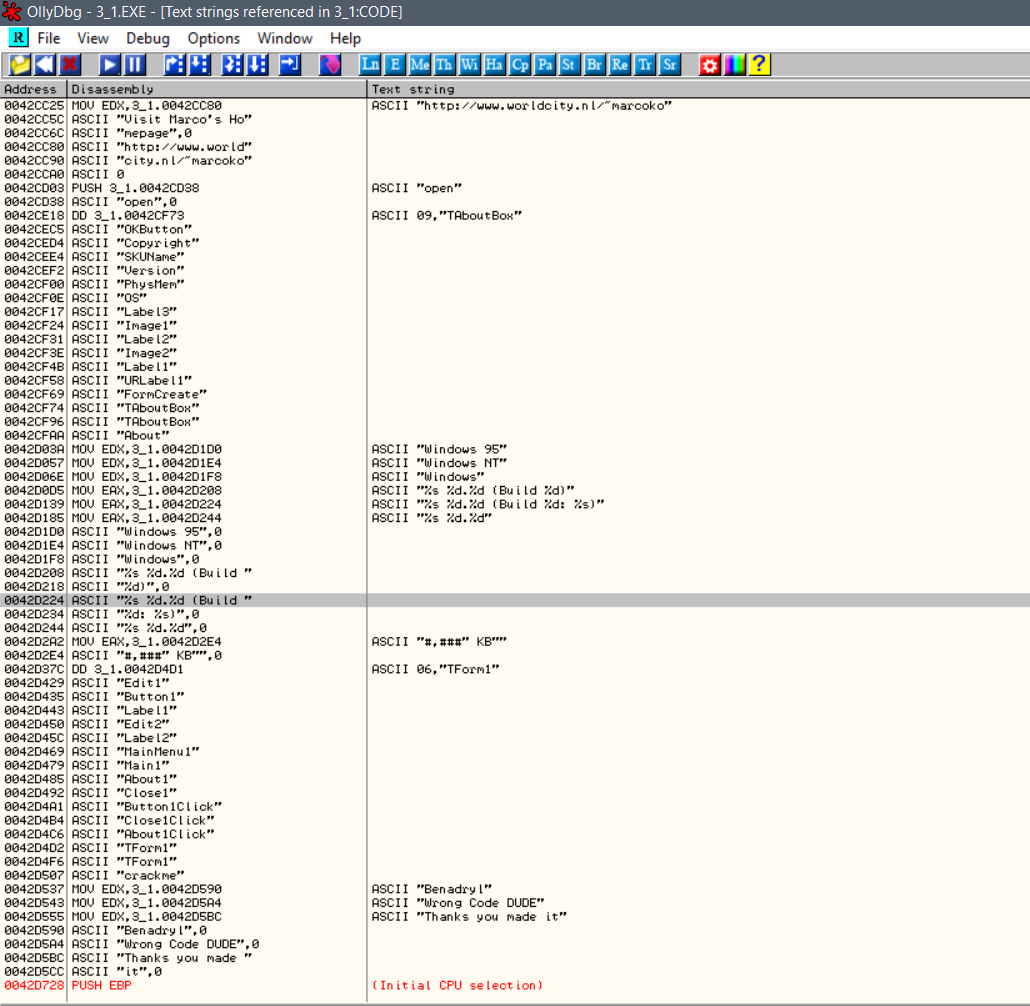


Hinh1. Đáp án điền ngẫu nhiên

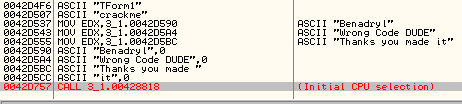
* Tiếp theo, mở Ollydbg lên, load file 3\_1 vào.

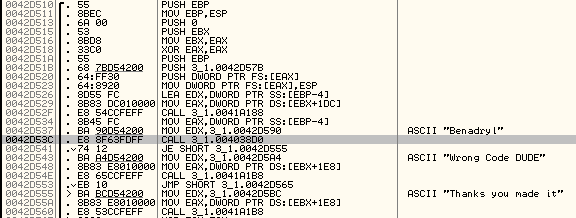


* Nhấn Open và đợi Ollydbg load xong.
* Chuột phải vào cửa sổ CPU -> Search for -> All referenced text strings.



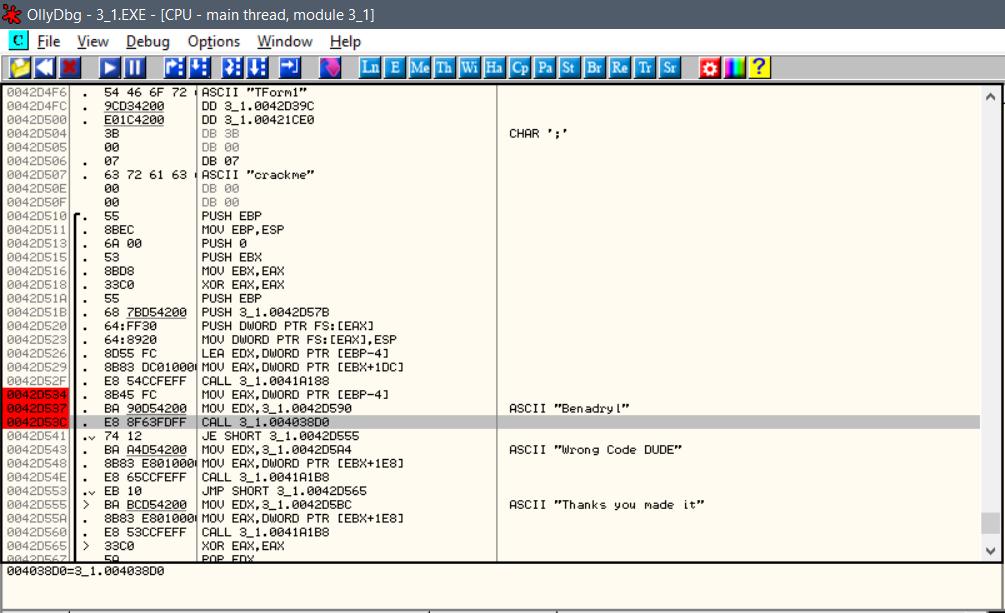
Hình 2. Giao diện khi mở All referenced text strings.

* Chú ý cửa sổ Text string, ta tìm được một số đoạn ASCII đáng nghi sau, trong đó có cả trạng thái status mà chúng ta khi nhập Serial ngẫu nhiên mà ta đã tìm được.
* Ta tìm được trạng thái status chính xác chính là đoạn “Thanks you made it”.
* Nhấn vào “Wrong Code DUDE” để tìm kiếm câu lệnh thực thi đến lệnh đó.
* Ta tìm được trạng thái status chính xác chính là đoạn “Thanks you made it”.
* Nhấn vào “Wrong Code DUDE” để tìm kiếm câu lệnh thực thi đến lệnh đó.

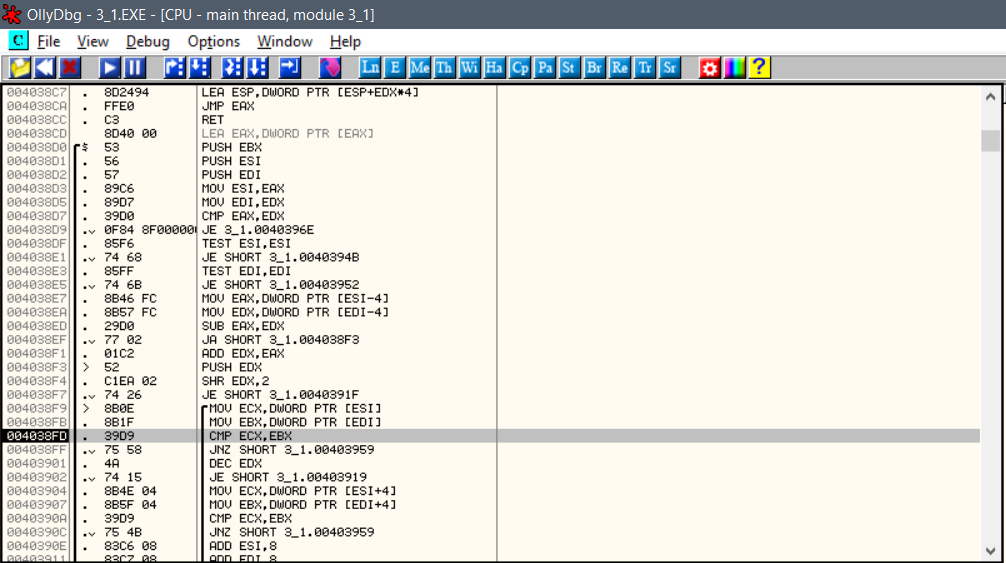


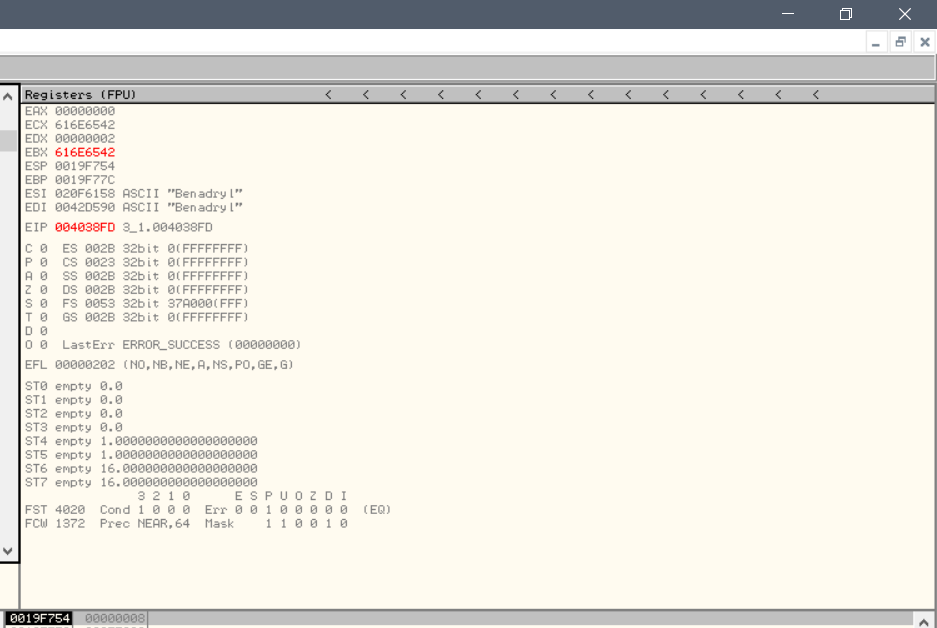
* Chú ý quan sát, ta thấy trước trạng thái status sai có một lệnh nhảy JE đến địa chỉ 0042D555, tìm xuống phía dưới ta nhận thấy 0042D555 chính là địa chỉ của trạng thái status đúng.
* Chú ý dòng lệnh ***0042D53C***, đây là dòng lệnh gán địa chỉ chuỗi ASCII “Benadryl” vào thanh ghi EDX.
* Ngoài ra, ta nhận lấy địa chỉ BA 90D54200 có chứa Từ khóa: “Benadryl”.

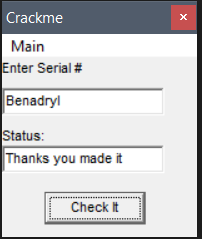
Ghi ngờ đây chính là keyword của chương trình.

* Dự đoán đây chính là lệnh gán địa chỉ chuỗi mà user nhập vào vào thanh ghi EAX. Tiến hành đặt Break Point tại đây.
* Sau đó, ta debug đến địa chỉ E8 8F63FDFF rồi bấm F7 để thực hiện hàm.
* Ta nhìn sang bảng Register thì thấy EDI đang chứa ASCII “Benadryl”,

ESI chứa “chuỗi ban đầu nhập vào”.

* Khi chạy sẽ thấy ECX chứa giá trị hệ hex của chuỗi vữa nhập, còn EBX chứa giá trị hệ hexa của EDI.
* Khi so sánh 2 chuỗi ECX và EBX thì thấy không chính xác và nhảy xuống hàm khác. Vì vậy, ta dự đoán Serial là “Benadryl”

* Quả nhiên ở cửa sổ Register thì giá trị thanh ghi EAX đã trùng khớp với những gì ta dự đoán.
* Vậy tức là lệnh CALL ở địa chỉ 0042D53C sẽ tiến hành so sánh giữa hai thanh ghi EAX và EDX sao cho lệnh JE SHORT 3\_1.0042D555 có thể thực thi (kết quả chính xác).
* Cũng có nghĩa là serial chính là “Benadryl”. Nhập serial tìm được ta được goodboy.



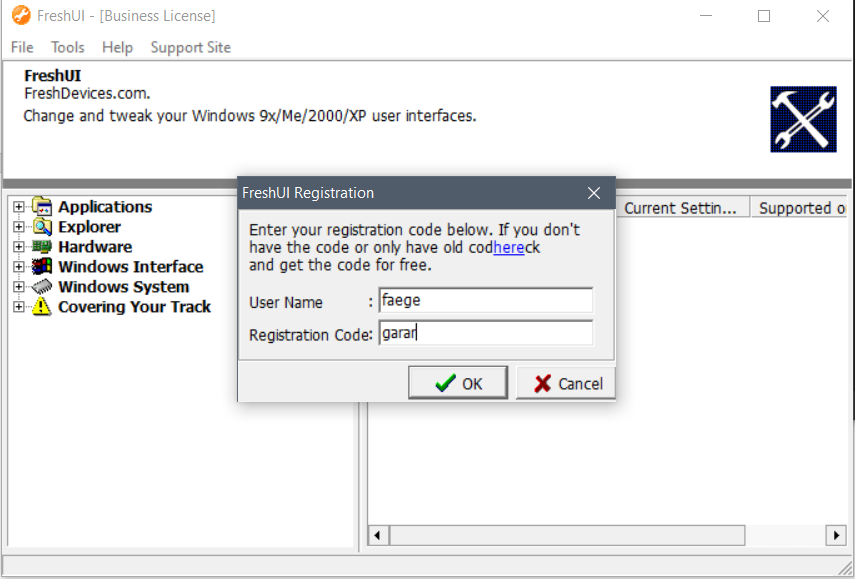
Bài 3.2:

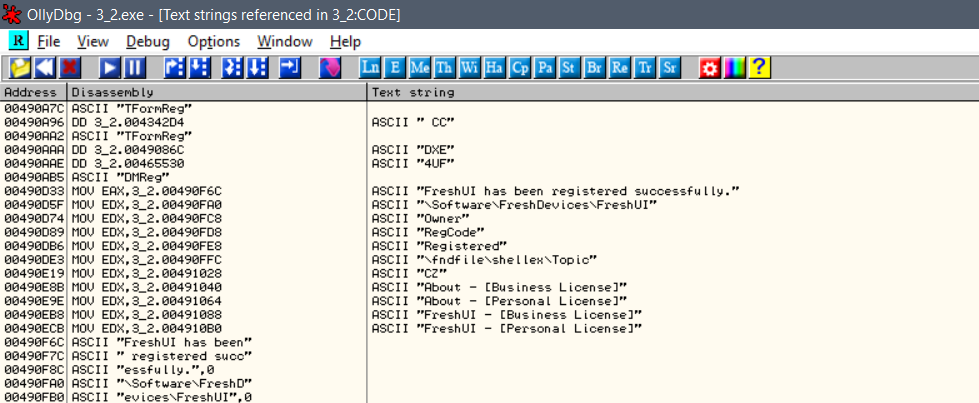
* Giao diện chương trình cần crack.

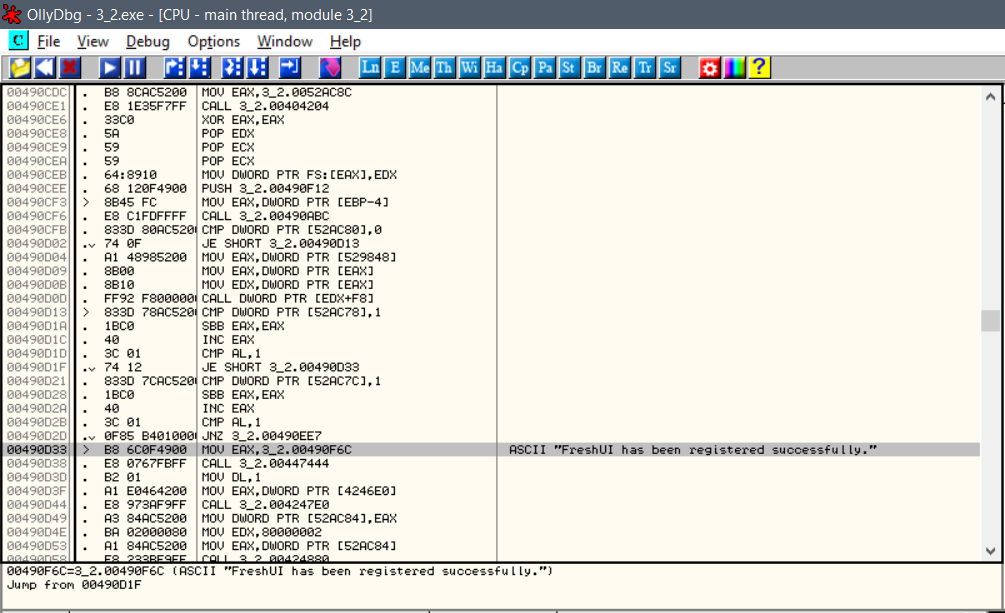
Nhập thử User Name : faege

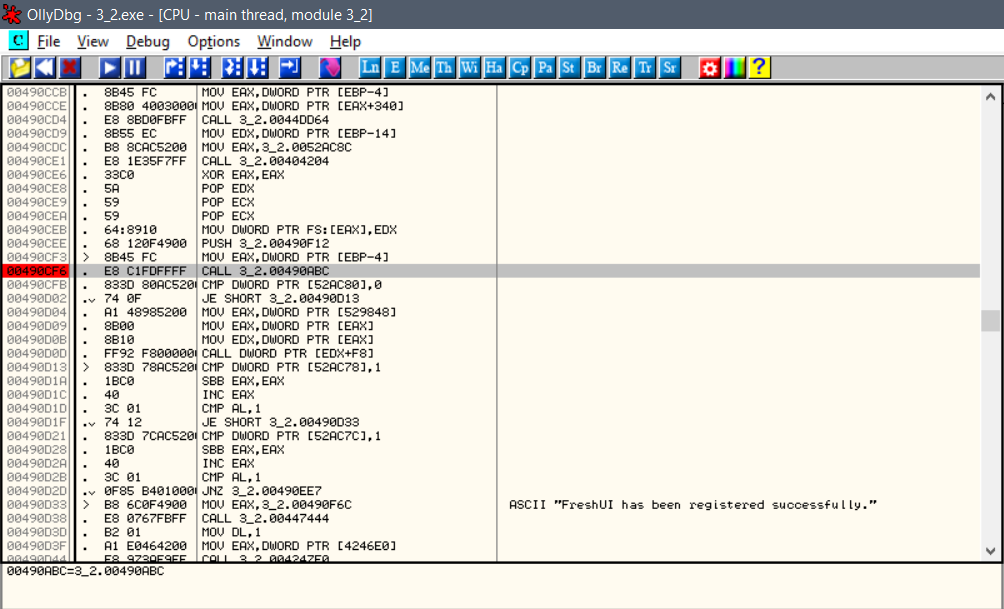
Registration Code : garar

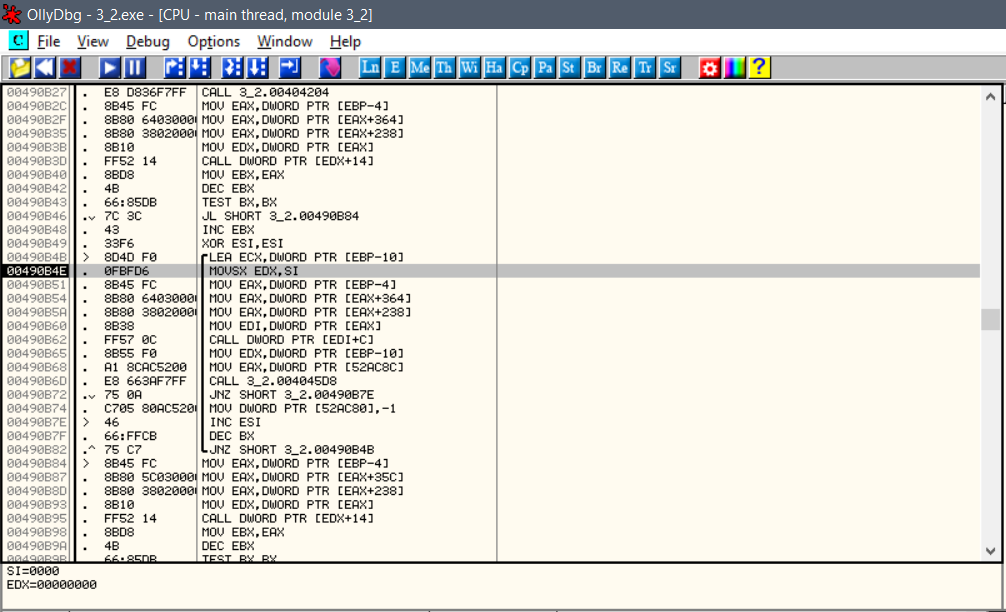
Khi nhập sai User Name và Registration Code, chương trình sẽ tự xóa dữ liệu nhập vào và không thông báo => điều này khiến việc tìm badboy trở nên khó khăn

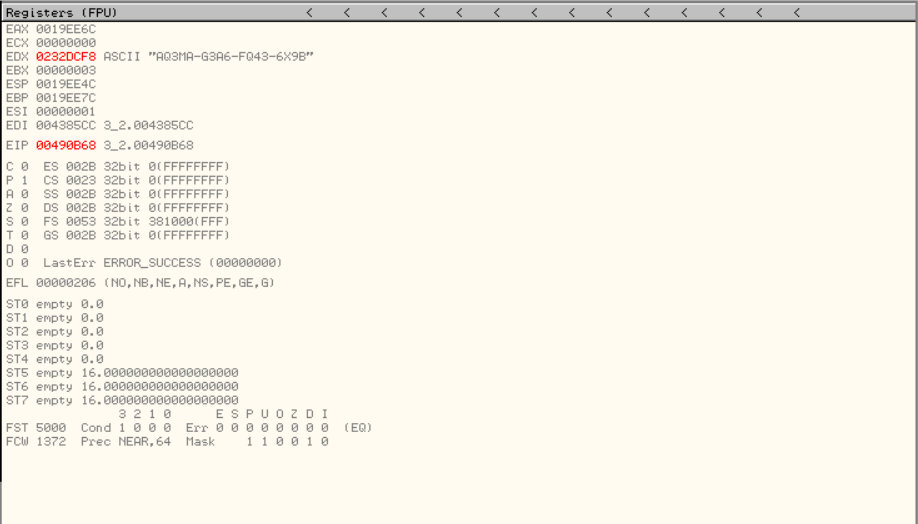


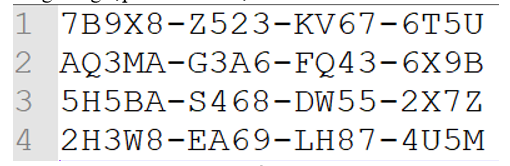
* Tìm trong “All reference text strings” ta sẽ thấy được goodboy của chương trình ở đây (sử dụng Search for text tìm những từ hay xuất hiện trong thông báo chính xác).
* Tìm ra được dòng 00490D33 chứa goodBoy.



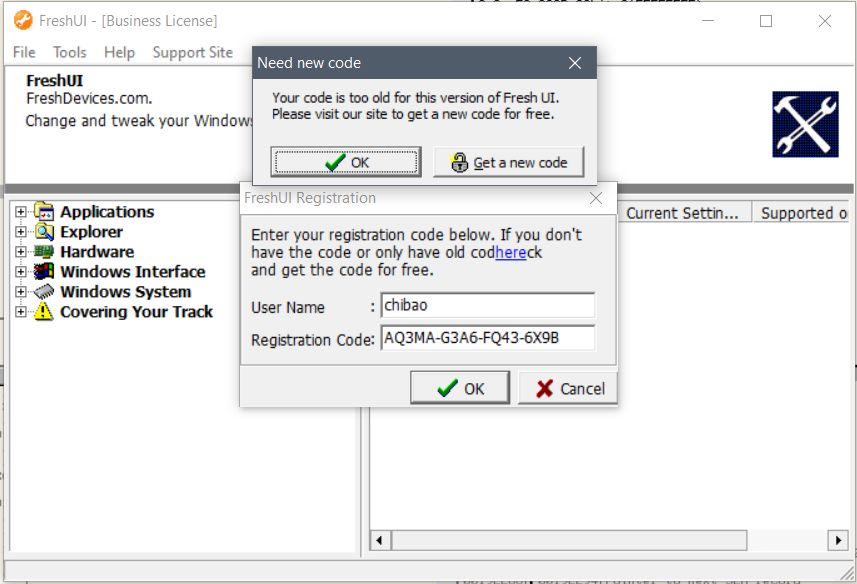
* Ta có thể thấy dòng 00490D1F là lệnh nhảy đến 00490D33 nếu như key đúng.
* Dựa vào câu lệnh nhảy ta có thể tìm ra được quy trình các bước kiểm tra name và registration key do người dùng nhập vào.
* Ta nhìn lên phía trên, thấy có câu lệnh Call ở địa chỉ 00490CF6.
* Ta đặt breakpoint ở địa chỉ 00490CF6 và dùng lệnh f7 để debug into vào trong cái hàm con đó.
* Sau đó dung Step over (F8) để đi xem từng dòng, để ý sự thay đổi giá trị thanh ghi bên bảng Registers.
* Tới đó vẫn chưa thấy gì thay đổi đáng kể, ta tiếp túc F8 xem sao.
* Khi nhảy đến dòng 00490B4B ta thấy có 1 vòng lặp.



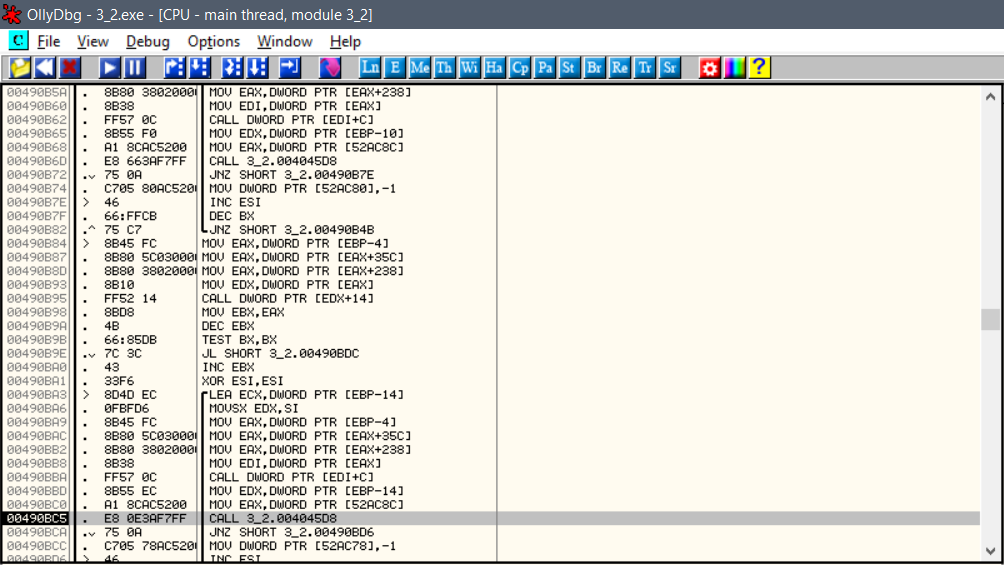
* Vòng lặp trên lần lượt lấy địa chỉ chuỗi Registation Code của người dung nhập để nạp vào thanh ghi EAX và địa chỉ một chuỗi nào đó có sẵn nạp vào thanh ghi EDX.
* Lệnh CALL 3\_2.004045D08 so sánh nội dung 2 chuỗi EAX và EDX. Nếu EAX = EDX thì PTR [52AC80] = -1.
* Tiếp tục dùng F8 chạy hết vòng lặp, và để ý bên bảng Registers thấy thanh ghi EDX chứa một chuỗi ASCII nào đó( có khả năng đó là key), EAX chứa key mà mình đã nhập, ESI dùng để lưu biến đếm của vòng lặp.
*  Sau khi chạy xong vòng lặp ta tìm ra được 4 chuỗi :

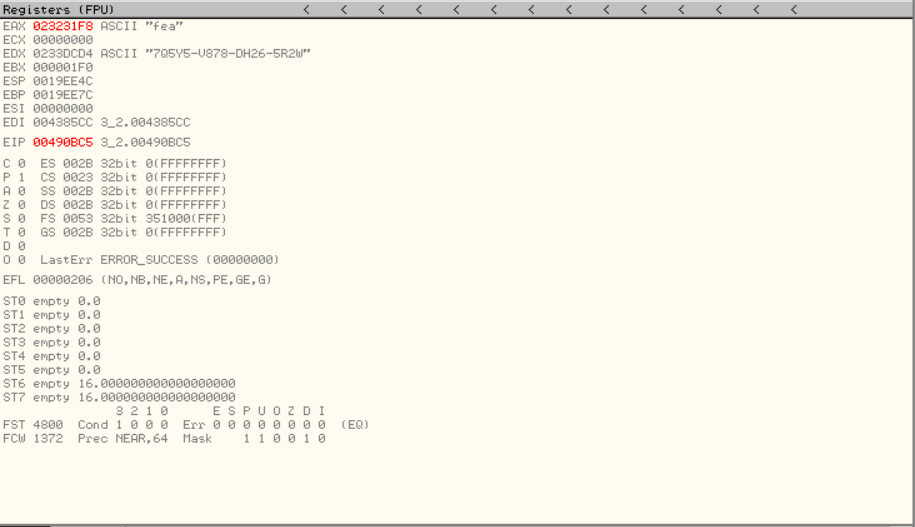


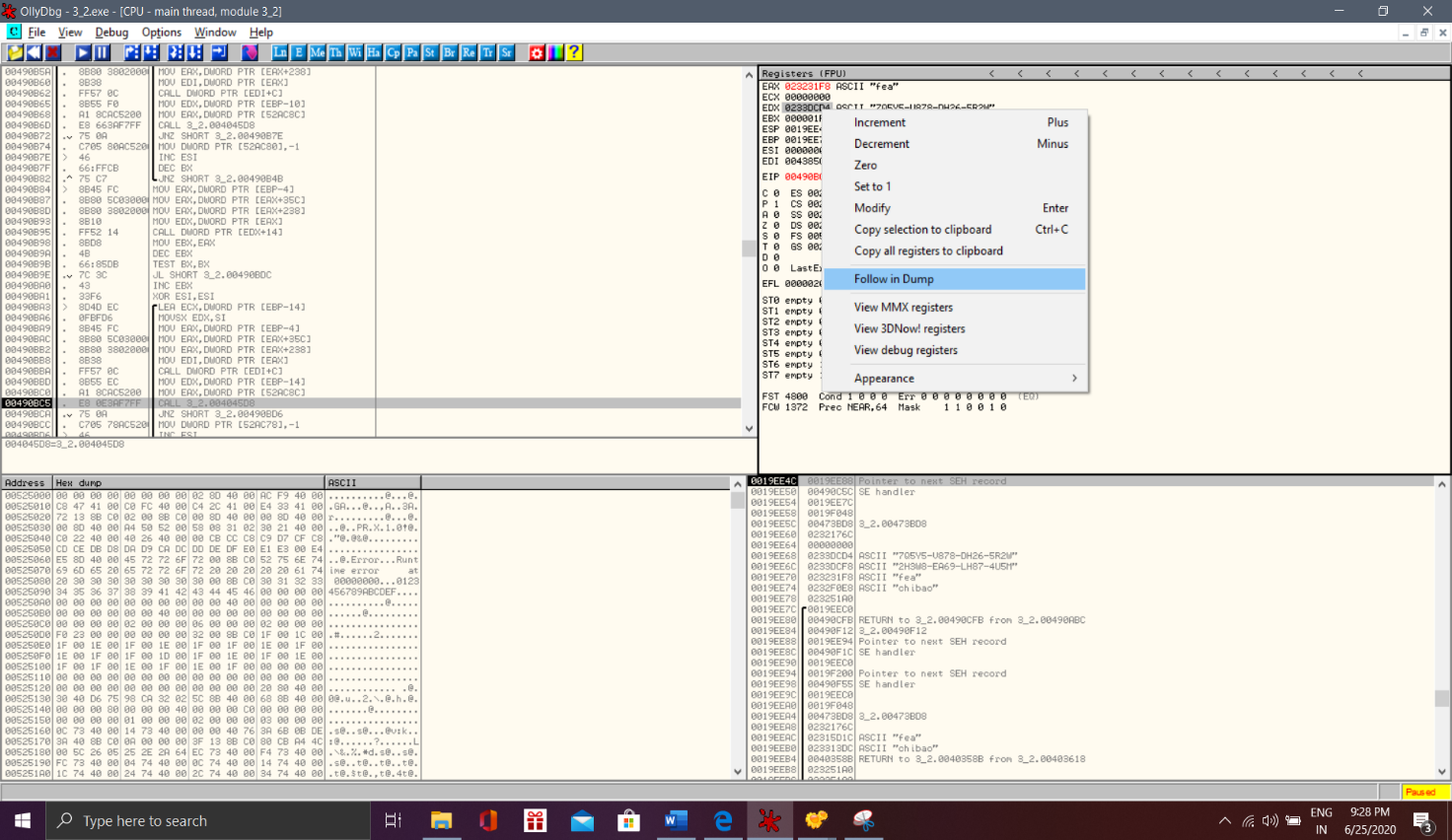
* Ta thử nhập từng chuỗi xem có phải key cần tìm không nhé.
* Ta nhận thấy tất cả các chuỗi vừa nhập là Key cũ.

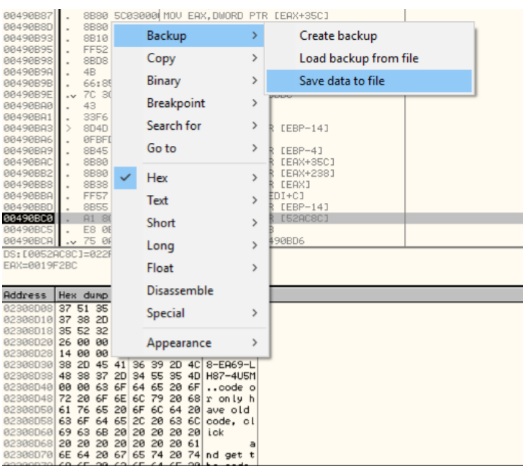


- Ta tiếp tục dùng F8 để đi qua các lệnh, và ta lại thấy xuất hiện thêm vòng lặp tương tự lúc nãy.(Hy vọng đây là các chuỗi key cần tìm).



* Ta nhấn F8 để chạy vòng lặp này, có thể thấy số lần lặp là rất lớn. Nên ta để ý thử xem có gì đặc biệt không. Thấy 1 địa chỉ chứa chuỗi ASCII nhảy liên tục theo vòng lặp. Và đó là địa chỉ của chứa chuỗi trên thanh EDX.
* Ta chú ý ở bảng Registers và theo code, ta thấy ESI là biến đếm của vòng lặp, và EBS là giới hạn số vòng lặp. Vòng lặp sẽ thoát ra khi EBS = 0.

* Nhấn chuột phải chọn “Follow in dump” ở thanh ghi chứa địa chỉ mã ASCII, sau đó click chuột phải chọn Backup -> save data to file, ta được 1 file lưu tất cả đoạn mã Registration code do chương trình tạo ra.



Ta được file Key.mem gồm 1000 dòng. Ta có thể nhận ra 4 dòng Key cuối là key cũ ta đã tìm được ở trên.

**Tổng kết quá trình crack:**

- Để ý thấy rằng, chương trình không kiểm tra USER NAME và chỉ kiểm tra mỗi REGISTATION CODE. Do đó, ta có thể đăng ký bằng một USER NAME bất kì để đăng nhập còn registration thì chọn bất kì từ 1 đến 996 đều chính xác.

- Vậy ta thu được 1 file thu được các mã key của chương trình (file key.mem kèm theo báo cáo). Trong đó dòng 1 đến dòng 996 là key dùng để crack chương trình, dòng 997 đến dòng 1000 là 4 key của “Phiên bản cũ”.