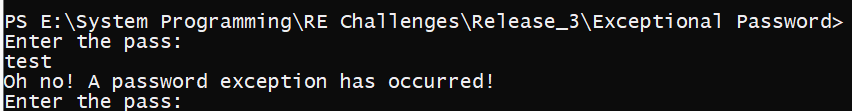
**LẬP TRÌNH HỆ THỐNG– LỚP NT209.L21.ANTN**

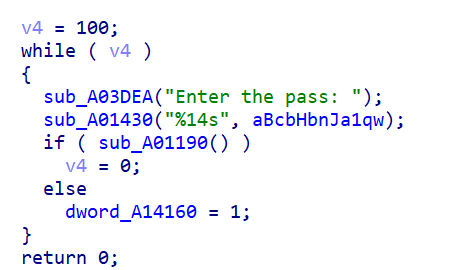
**RE CHALLENGES 3: Exceptional Password**

|  |  |  |  |
| --- | --- | --- | --- |
| **Giảng viên hướng dẫn** | **Phạm Văn Hậu** | | **ĐIỂM** |
| **Sinh viên thực hiện 1** | **Trần Đức Lương** | **19521815** |  |

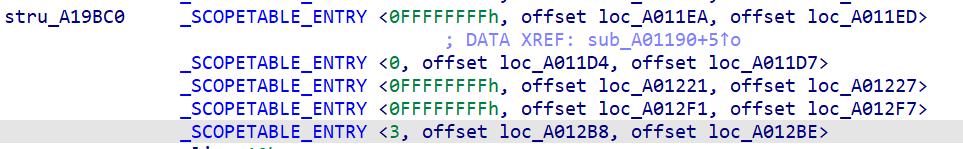
Đây là file exe PE 32-bit. Thử chạy chương trình với password nhập vào là “*test*” thì chương trình báo sai. Ý tưởng của bài này cũng là đi tìm password đúng để chạy chương trình thành công.



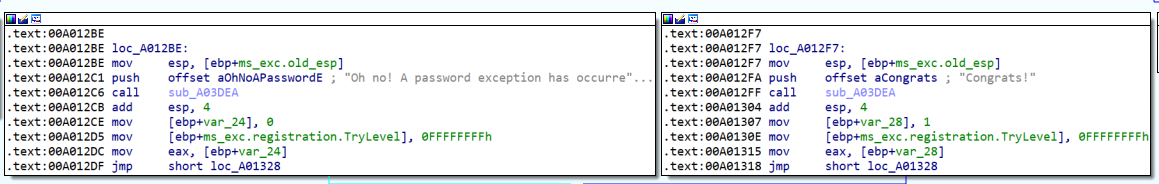
Mở file bằng IDA Pro bắt dầu quá trình dịch ngược. Hàm main của chương trình bắt đàu bằng việc nhập password gồm 14 kí tự và được lưu vào biến aBcbHbnJa1qw. Quá trình xử lí sẽ nằm trong hàm sub\_A01190.



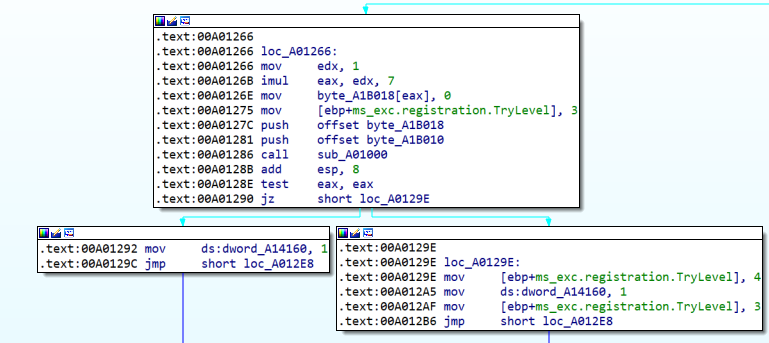
Mở hàm sub\_A01190 thì thấy chương trình là sử dụng cấu trúc try catch được lưu ở stru\_A19BC0. Giá trị [ebp + s\_exc.registration.TryLevel] chính là ví trí của hàm exception cần nhảy đến, ví dụ bằng 2 thì là hàm tại offset loc\_A01227, bằng 3 thì là hàm tại offset loc\_A012F7…



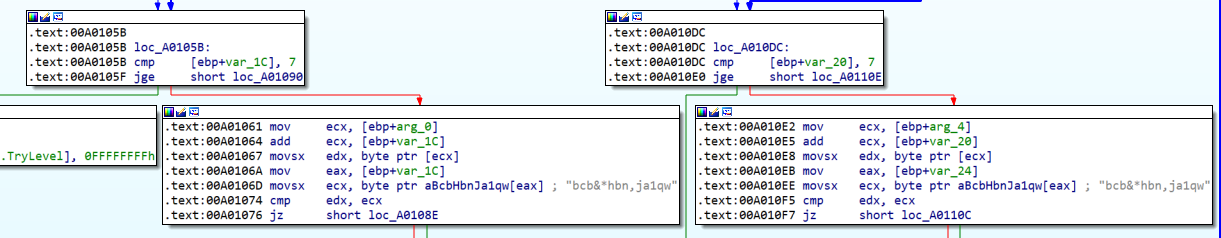
Quá trình đọc code và phân tích bằng debug tương đối dài nhưng tóm gọn cụ thể nhưng hàm loc\_A012F7 chính là mục tiêu cần hướng đến để in ra chữ “Congrats!”, nó tương đương với việc exception phải nhảy tới hàm này với [ebp + s\_exc.registration.TryLevel] == 3.

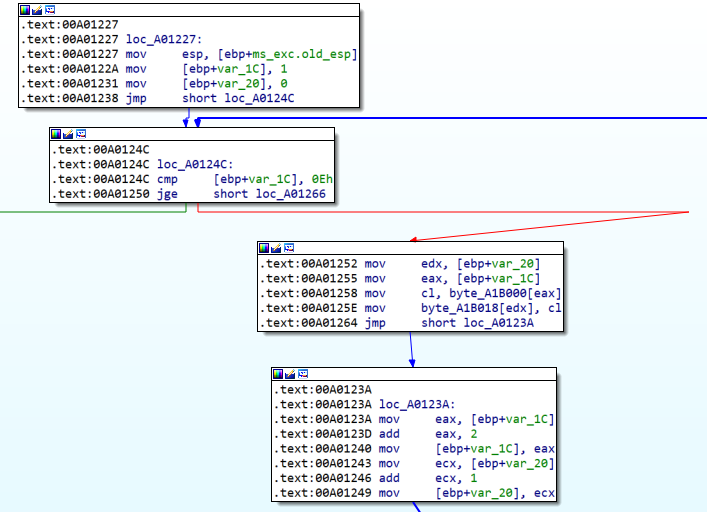


Để làm được điều đó, tại loc\_A01266 kết quả trả về của hàm sub\_A01000 với 2 tham số là 2 mảng byte\_A1B018 và byte\_A1B010 phải bằng 1. Nếu trả về 0 chương trình sẽ in ra “Oh no! A password exception has occurred!”.

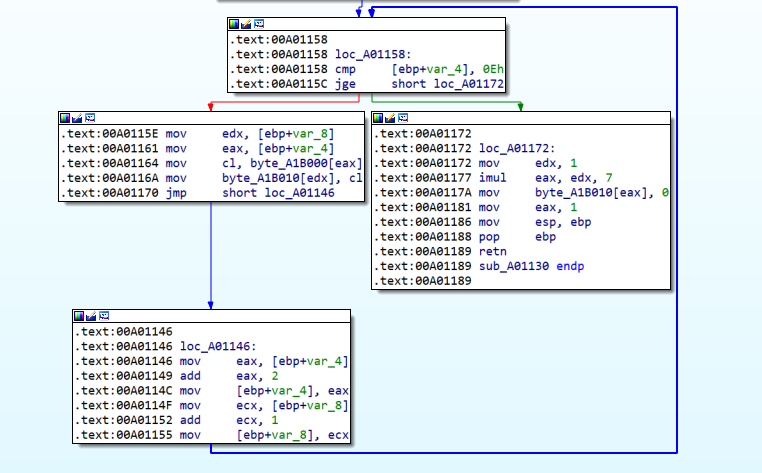


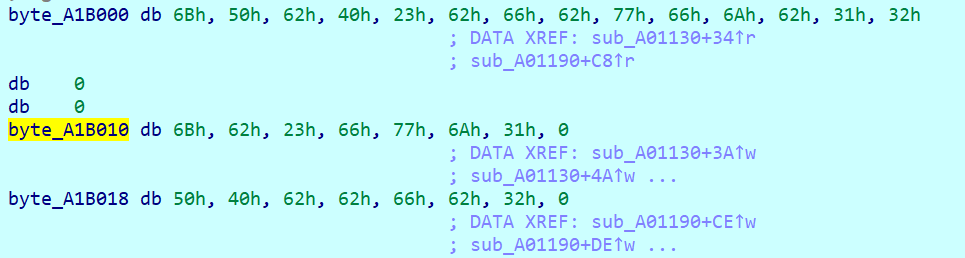
Kiểm tra hàm sub\_A01000 thì thấy chương trình so sánh password ta nhập vào với lần lượt với 2 tham số đầu vào là byte\_A1B018 và byte\_A1B010. Cụ thể với trường hợp này, 7 kí tự đầu của password nhập vào được so sánh với 7 kí tự trong mảng byte\_A1B010 và 7 kí tự sau với 7 kí tự trong mảng byte\_A1B018. Nếu kết quả là bằng nhau cả thì hàm trả về 1. Điều này chứng tỏ ta cần nhập password là chuỗi nối giữa 7 kí tự của mảng byte\_A1B010 và 7 kí tự của mảng byte\_A1B018.



Ta sẽ đi tìm giá trị của 2 mảng trên. Cụ thể, tại hàm loc\_A01227 chương trình gán các giá trị của mảng byte\_A1B018 lần lượt bằng các kí tự ở vị trí lẻ (tính từ 1) của mảng byte\_A1B000. 

Tương tự tại hàm sub\_A01130, các giá trị của mảng byte\_A1B010 lần lượt được gán bằng kí tự ở vị trí chẵn (tính từ 0) của mảng byte\_ A1B000.



Khi đó, từ chuỗi byte\_A1B000 = “***kPb@#bfbwfjb12***” ta suy ra được chuỗi byte\_A1B010 = “***kb#fwj1***” và byte\_A1B1018 = “***P@bbfb2***”. 

Như vậy pass chúng ta cần nhập là “***kb#fwj1P@bbfb2***”. Thử chạy và thấy thành công.

