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.

```
PS E:\System Programming\RE Challenges\Release_3\Exceptional Password>
Enter the pass:
test
Oh no! A password exception has occurred!
Enter the pass:
```

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.

```
v4 = 100;
while ( v4 )
{
  sub_A03DEA("Enter the pass: ");
  sub_A01430("%14s", aBcbHbnJa1qw);
  if ( sub_A01190() )
    v4 = 0;
  else
    dword_A14160 = 1;
}
return 0;
```

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…

```
stru_A19BC0     _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_A011EA, offset loc_A011ED>
                                  ; DATA XREF: sub_A01190+5↑o
                _SCOPETABLE_ENTRY <0, offset loc_A011D4, offset loc_A011D7>
                _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_A01221, offset loc_A01227>
                _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_A012F1, offset loc_A012F7>
                _SCOPETABLE_ENTRY <3, offset loc_A012B8, offset loc_A012BE>
```

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.

```
.text:00A012BE
.text:00A012BE loc_A012BE:
.text:00A012BE mov     esp, [ebp+ms_exc.old_esp]
.text:00A012C1 push    offset aOhNoAPasswordE ; "Oh no! A password exception has occurre"...
.text:00A012C6 call    sub_A03DEA
.text:00A012CB add     esp, 4
.text:00A012CE mov     [ebp+var_24], 0
.text:00A012D5 mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFFFh
.text:00A012DC mov     eax, [ebp+var_24]
.text:00A012DF jmp     short loc_A01328
```

```
.text:00A012F7
.text:00A012F7 loc_A012F7:
.text:00A012F7 mov     esp, [ebp+ms_exc.old_esp]
.text:00A012FA push    offset aCongrats ; "Congrats!"
.text:00A012FF call    sub_A03DEA
.text:00A01304 add     esp, 4
.text:00A01307 mov     [ebp+var_28], 1
.text:00A0130E mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFFFh
.text:00A01315 mov     eax, [ebp+var_28]
.text:00A01318 jmp     short loc_A01328
```

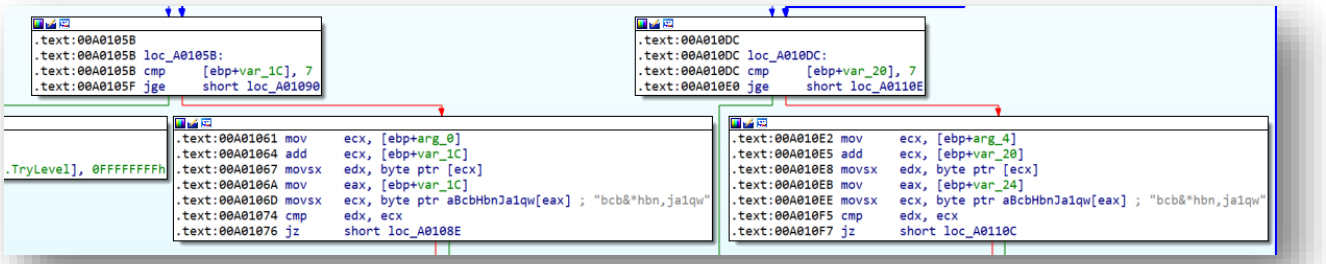
Để làm được điều đó, tại loc_A01266 kết quả trả về của hàm sub_A01000 với 2 tham số là 2 mảng byte_A1B018 và byte_A1B010 phải bằng 1. Nếu trả về 0 chương trình sẽ in ra “Oh no! A password exception has occurred!”.

```
.text:00A01266
.text:00A01266 loc_A01266:
.text:00A01266 mov     edx, 1
.text:00A0126B imul    eax, edx, 7
.text:00A0126E mov     byte_A1B018[eax], 0
.text:00A01275 mov     [ebp+ms_exc.registration.TryLevel], 3
.text:00A0127C push    offset byte_A1B018
.text:00A01281 push    offset byte_A1B010
.text:00A01286 call    sub_A01000
.text:00A0128B add     esp, 8
.text:00A0128E test    eax, eax
.text:00A01290 jz      short loc_A0129E
```

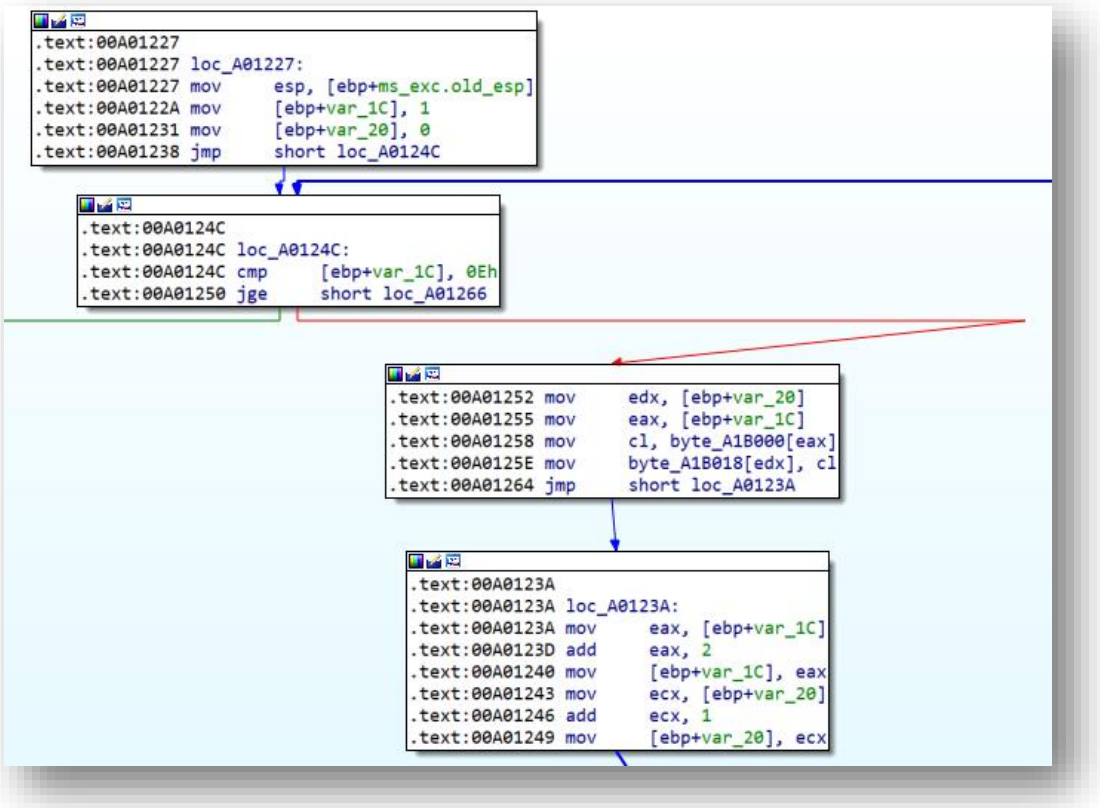
```
.text:00A01292 mov     ds:dword_A14160, 1
.text:00A0129C jmp     short loc_A012E8
```

```
.text:00A0129E
.text:00A0129E loc_A0129E:
.text:00A0129E mov     [ebp+ms_exc.registration.TryLevel], 4
.text:00A012A5 mov     ds:dword_A14160, 1
.text:00A012AF mov     [ebp+ms_exc.registration.TryLevel], 3
.text:00A012B6 jmp     short loc_A012E8
```

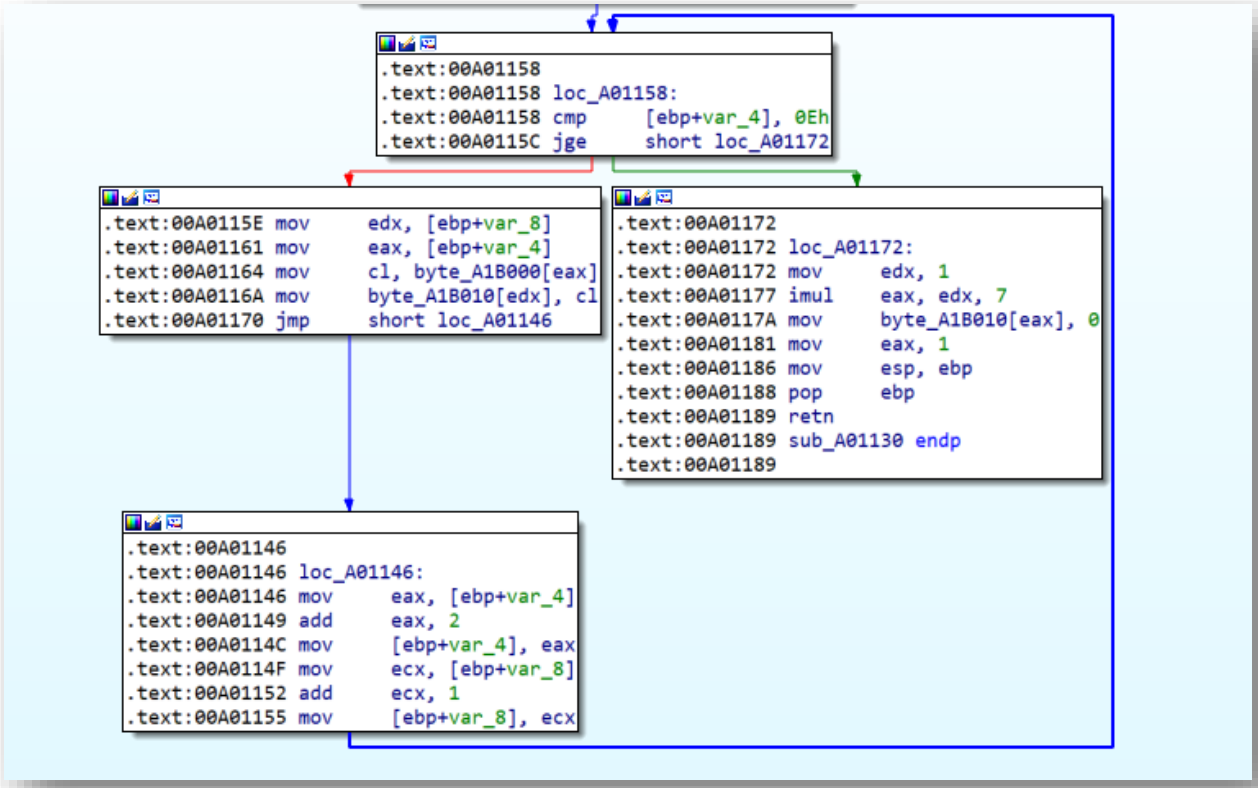
Kiểm tra hàm sub_A01000 thì thấy chương trình so sánh password ta nhập vào với lần lượt với 2 tham số đầu vào là byte_A1B018 và byte_A1B010. Cụ thể với trường hợp này, 7 kí tự đầu của password nhập vào được so sánh với 7 kí tự trong mảng byte_A1B010 và 7 kí tự sau với 7 kí tự trong mảng byte_A1B018. 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***".

```
byte_A1B000 db 6Bh, 50h, 62h, 40h, 23h, 62h, 66h, 62h, 77h, 66h, 6Ah, 62h, 31h, 32h
                                        ; DATA XREF: sub_A01130+34↑r
                                        ; sub_A01190+C8↑r
db    0
db    0
byte_A1B010 db 6Bh, 62h, 23h, 66h, 77h, 6Ah, 31h, 0
                                        ; DATA XREF: sub_A01130+3A↑w
                                        ; sub_A01130+4A↑w ...
byte_A1B018 db 50h, 40h, 62h, 62h, 66h, 62h, 32h, 0
                                        ; DATA XREF: sub_A01190+CE↑w
                                        ; sub_A01190+DE↑w ...
```

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