Exceptional Password

Chạy thử chương trình

```
C:\Users\BILL\Desktop\Release_3\Exceptional Password>.\excepional_pass.exe
Enter the pass:
fasfs
Oh no! A password exception has occurred!
Enter the pass:
sdfs
Oh no! A password exception has occurred!
Enter the pass:
```

Chương trình yêu cầu nhập pass, phân tích với IDA

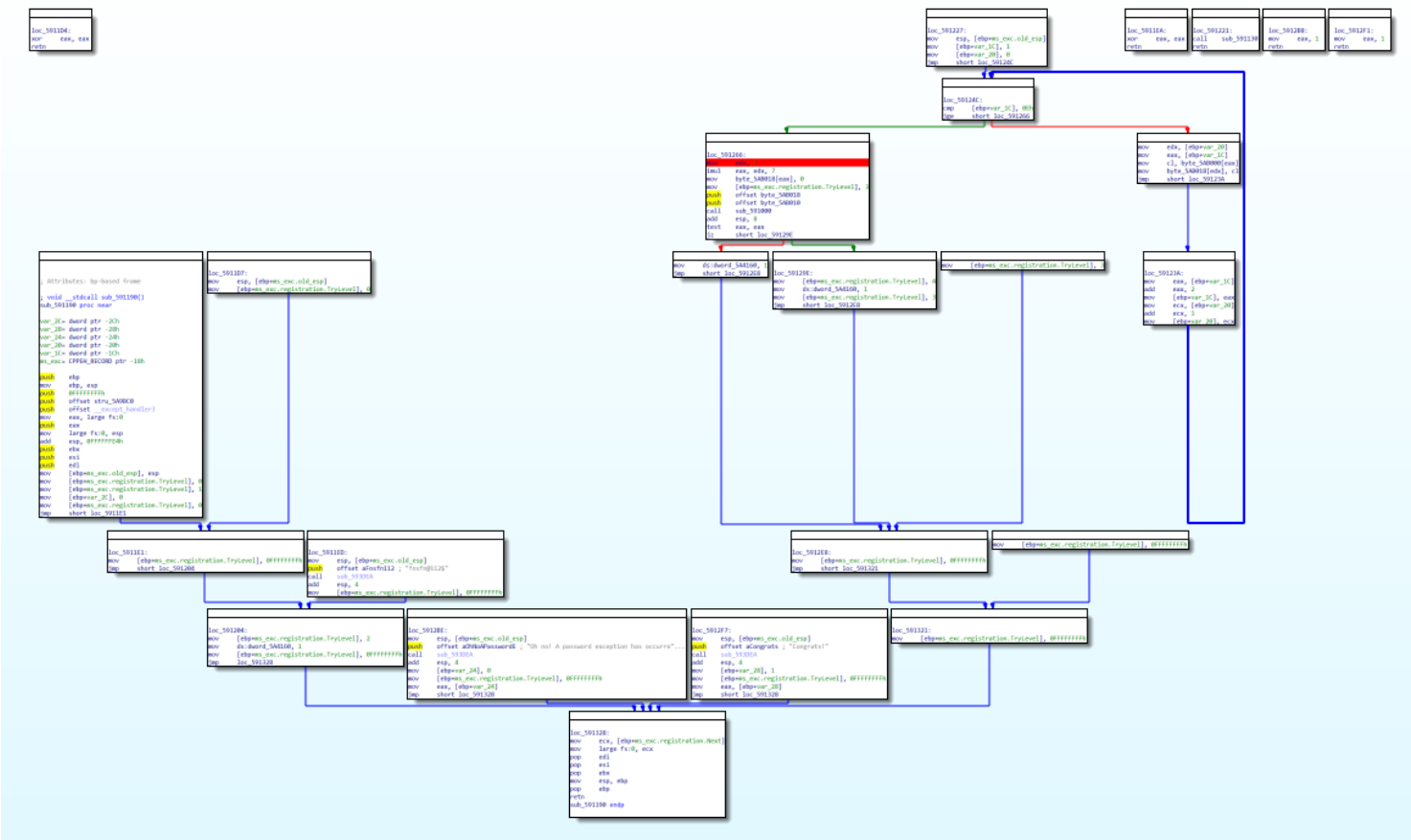
```
// write access to const memory has been detected, the output may be wrong!
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int v3; // eax
  int v5; // [esp+Ch] [ebp-1Ch]

  v5 = 100;
  while ( v5 )
  {
    sub_593DEA("Enter the pass: ");
    sub_591430("%14s", pass);
    sub_591190();
    if ( v3 )
      v5 = 0;
    else
      dword_5A4160 = 1;
  }
  return 0;
}
```

Chương trình yêu cầu nhập chuỗi có độ dài 14, lưu vào biến pass.

Sau đó gọi hàm sub_591190()

Hàm sub_591190 viết theo hướng sử dụng try catch, nên decomplie không thể dịch mã giả được, ta phải xem sơ đồ graph



Ban đầu chương trình đưa vào một struct try catch như sau, lưu ở vị trí stru_5A9BC0

```
push    ebp
mov     ebp, esp
push    0FFFFFFFFh
push    offset stru_5A9BC0
push    offset __except_handler3
mov     eax, large fs:0
push    eax
mov     large fs:0, esp
add     esp, 0FFFFFFE4h
push    ebx
push    esi
push    edi
```

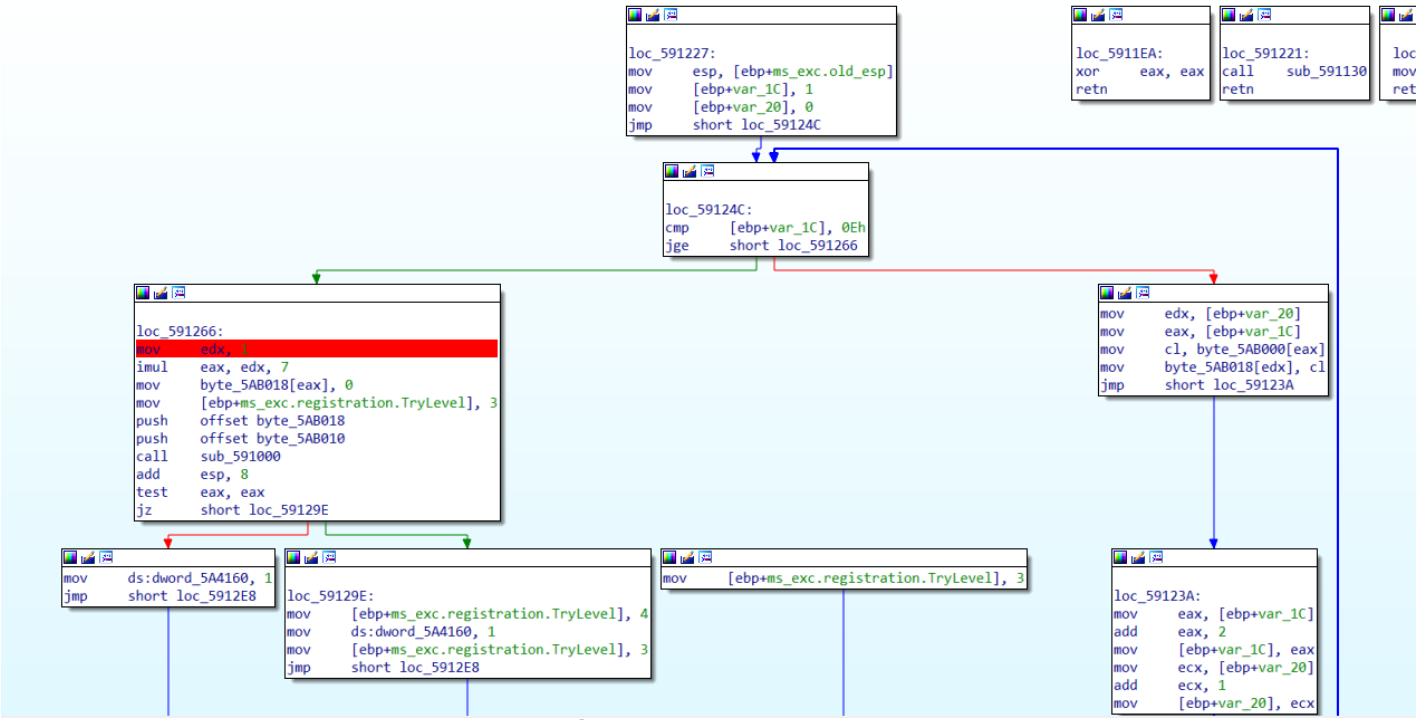
Đây là cấu trúc stru_5A9BC0

```
.rdata:005A9BC0 stru_5A9BC0     _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_5911EA, offset loc_5911ED>
.rdata:005A9BC0                                   ; DATA XREF: sub_591190+5↑o
.rdata:005A9BC0                 _SCOPETABLE_ENTRY <0, offset loc_5911D4, offset loc_5911D7>
.rdata:005A9BC0                 _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_591221, offset loc_591227>
.rdata:005A9BC0                 _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_5912F1, offset loc_5912F7>
.rdata:005A9BC0                 _SCOPETABLE_ENTRY <3, offset loc_5912B8, offset loc_5912BE>
```

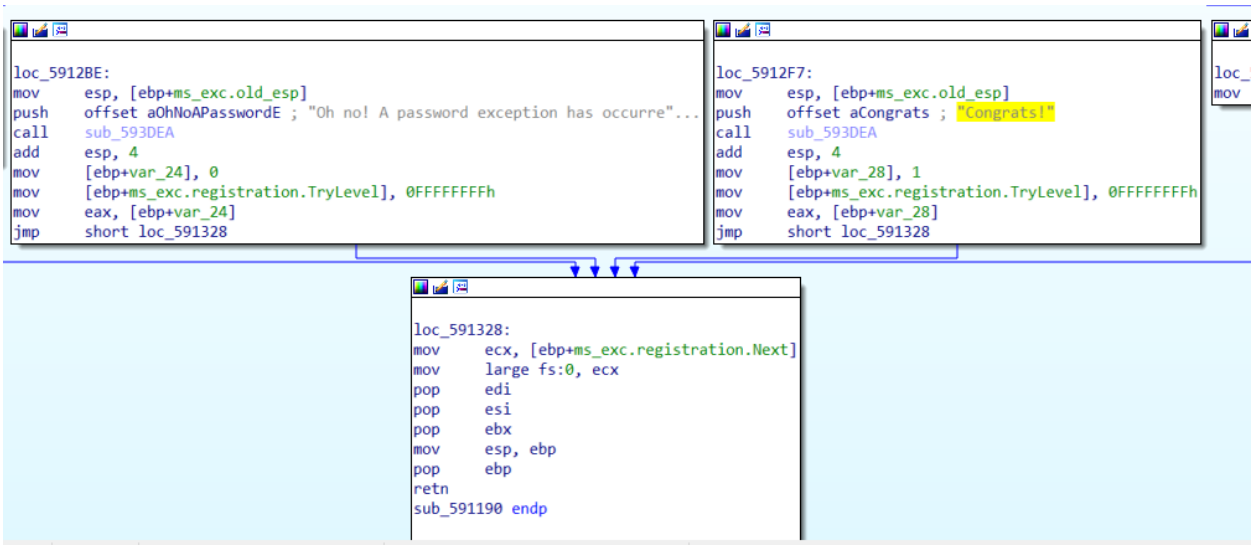
Đầu tiên khi debug, block loc_591204 gây exception ở câu lệnh mov ds:dword_5A4160, 1, vì ở đây là TryLevel2, nên hàm handle là loc_591227

```
loc_591204:
mov     [ebp+ms_exc.registration.TryLevel], 2
mov     ds:dword_5A4160, 1
mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFFFh
jmp     loc_591328
```

Đặt breakpoint và tiếp tục debug tiếp, đây là đoạn code tại loc_591227



Chương trình lặp từ 1->15, mỗi lần lặp tăng 2 và tạo ra chuỗi mới lưu tại byte_5AB018, sau đó gọi hàm sub_591000 với 2 tham số là byte_5AB018 và byte_5AB010. Kết quả trả về của hàm sẽ quyết định chuỗi xuất ra là "Congrats!" hay "Oh no! A password exception has occurred!".

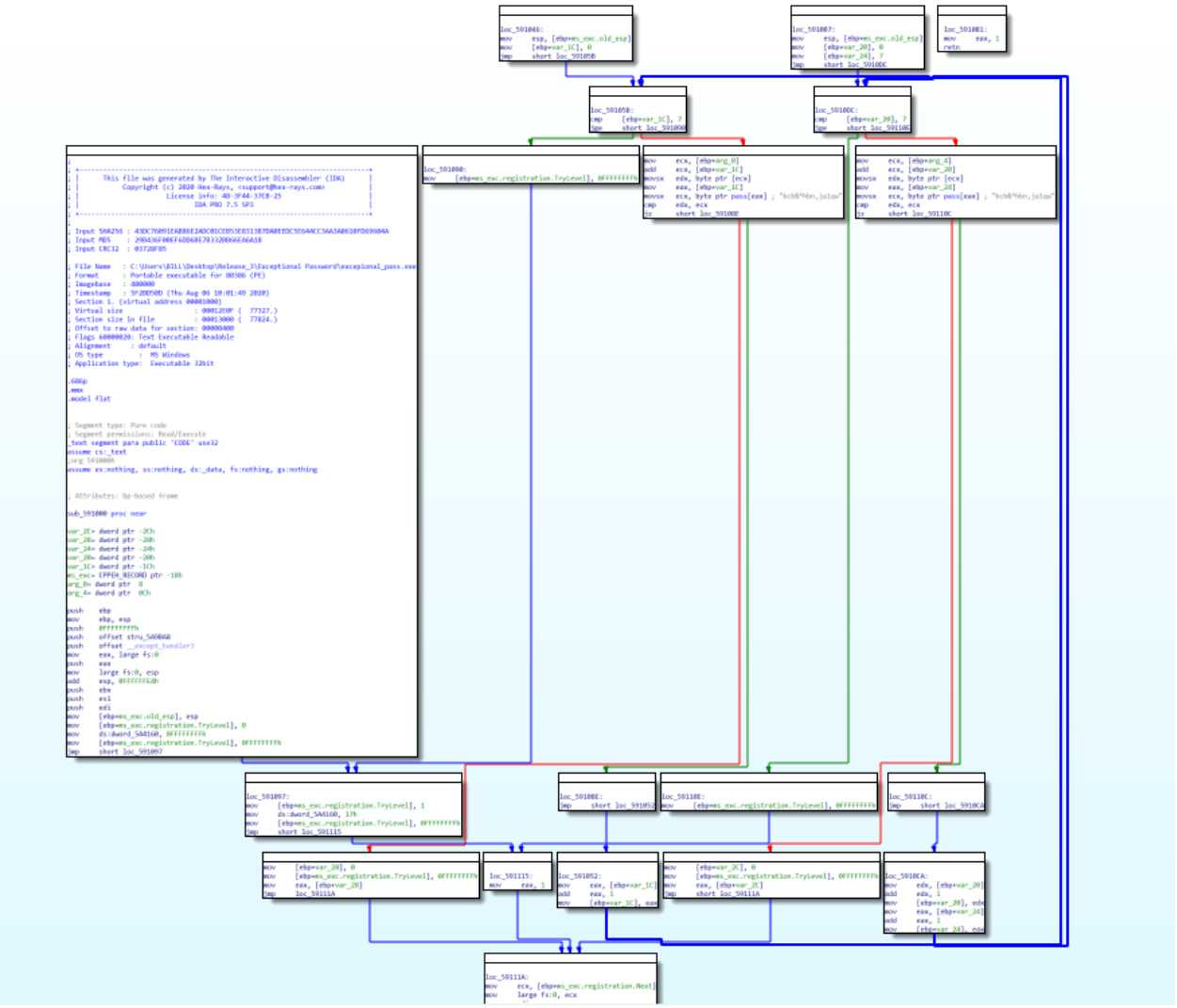


Đoạn code chúng ta cần nhảy vô là loc_5912F7 ở TryLevel3, đoạn code báo lỗi nằm ở loc_5912BE ở TryLevel4, như vậy giá trị trả về của hàm sub_591000 phải là khác 0 để lệnh mov ds:dword_5A4160, 1 gây ecxception ở level3. Còn nếu giá trị trả về bằng 0 thì loc_ 59129E được thực hiện, khi đó lệnh mov ds:dword_5A4160, 1 gây ecxception ở level4.

Đặt debug tìm giá trị 2 chuỗi byte_5AB018 và byte_5AB010 là

```
.data:005AB010 byte_5AB010 db 'k', 'b', '#', 'f', 'w', 'j', '1', 0
.data:005AB010                                         ; DATA XREF: sub_591130+3A↑w
.data:005AB010                                         ; sub_591130+4A↑w ...
.data:005AB018 byte_5AB018 db 'P', '@', 'b', 'b', 'f', 'b', '2', 0
.data:005AB018                                         ; DATA XREF: sub_591190+CE↑w
```

Tiếp tục đến với hàm sub_591000, cũng được viết dưới dạng try catch



Struct try catch mà hàm sử dụng

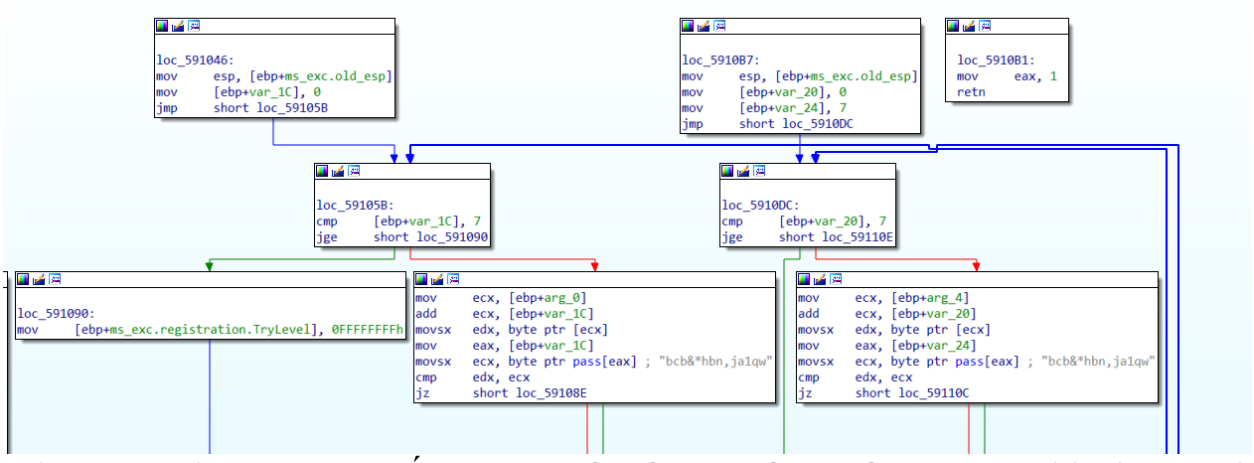
```
.rdata:005A9BA8 stru_5A9BA8     _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_591040, offset loc_591046>
.rdata:005A9BA8                                   ; DATA XREF: sub_591000+5↑o
.rdata:005A9BA8                 _SCOPETABLE_ENTRY <0FFFFFFFFh, offset loc_5910B1, offset loc_5910B7>
```

ở 2 câu lệnh mình tô đậm, gây ra 2 exption ở level1 và level2

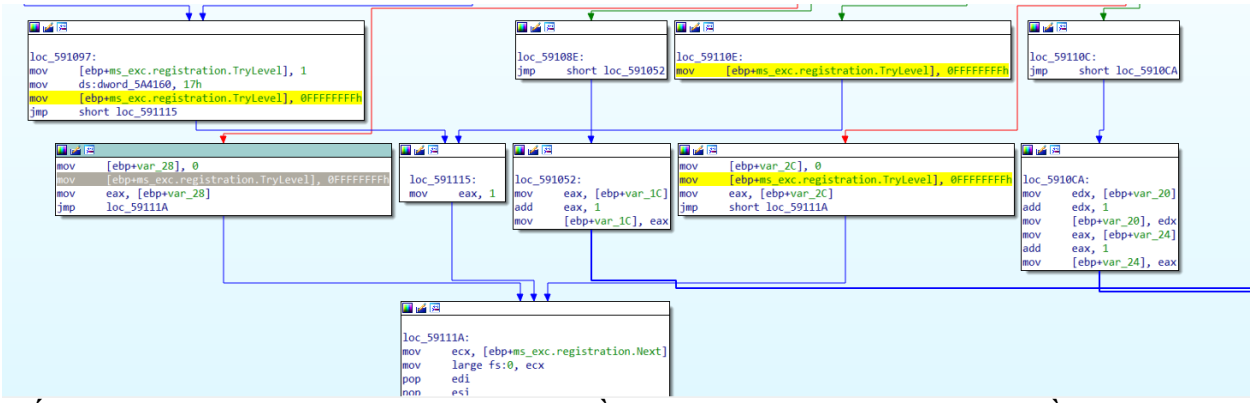
```
mov     ebp, esp
push    0FFFFFFFFh
push    offset stru_5A9BA8
push    offset __except_handler3
mov     eax, large fs:0
push    eax
mov     large fs:0, esp
add     esp, 0FFFFFFE4h
push    ebx
push    esi
push    edi
mov     [ebp+ms_exc.old_esp], esp
mov     [ebp+ms_exc.registration.TryLevel], 0
mov     ds:dword_5A4160, 0FFFFFFFFh
mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFFFh
jmp     short loc_591097
```

```
loc_591097:
mov     [ebp+ms_exc.registration.TryLevel], 1
mov     ds:dword_5A4160, 17h
mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFFFh
jmp     short loc_591115
```

Cả loc_591046 và loc_5910B7 đều có cách hoạt động giống nhau, lặp lại 7 lần và mỗi lần đều so sánh 1 kí tự của pass với 1 kí tự trong 2 chuỗi byte_5AB018 và byte_5AB010. loc_591046 so sánh 7 kí tự đầu của pass với chuỗi byte_5AB010 và loc_5910B7 so sánh 7 kí tự sau của pass với chuỗi byte_5AB018.



Chương trình sẽ return 0 nếu 1 trong các kí tự so sánh không khớp, 2 block có mũi tên đỏ chỉ tới sẽ gán eax bằng 0 và nhảy đến block return .



Nếu như ở hàm loc_591046 cả 7 kí tự đều khớp, chương trình sẽ trở về loc_591097 để tiếp tục gây exception để thực hiện loc_5910B7, nếu tiếp tục 7 kí tự sau đều khớp thì chương trình sẽ tiến tới loc_591115 để gán eax bằng 1 và return. Nếu 1 trong các kí tự không khớp thì hàm sẽ gán eax bằng 0 và return.

Chúng ta cần eax khác 0 tức cả 14 kí tự đều phải trùng khớp => chuỗi pass cần nhập là: kb#fwj1P@bbfb2 (nối 2 chuỗi chuỗi byte_5AB010 và byte_5AB018)

Chạy thử

```
C:\Users\BILL\Desktop\Release_3\Exceptional Password>.\excepional_pass.exe
Enter the pass:
kb#fwj1P@bbfb2
Congrats!
```

flag: kb#fwj1P@bbfb2