Họ tên: Võ Văn Đức

MSSV: 20110635

**Lab 1. Buffer Overflows**

A generic buffer overflow occurs when a buffer that has been allocated a specific storage space has more data copied to it than it can handle.

**Buffer overflow steps:**

1. Find the presense and location of buffer overflow vulnerability
2. Write mote data into the buffer than it can handle
3. Overwites the return address of a function
4. Changes the execution flow to the hacker code

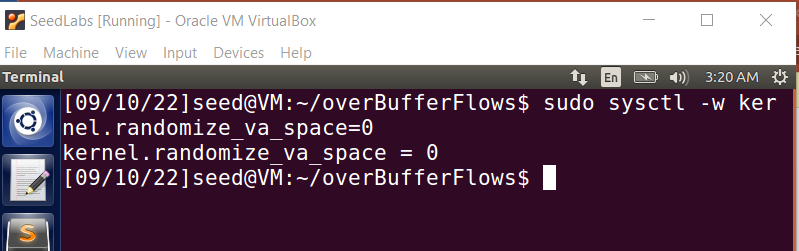
**Time duration:** 1 week

Lab guide:

**Step 0: Chuẩn bị**

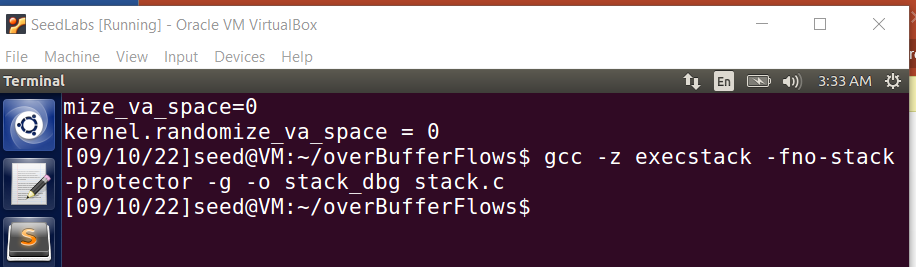
* **Ubuntu 16.04 (32-bit)**
* Source code: **stack.c, exploit.c/exploit.py**

**Step 1: Vô hiệu hoá địa chỉ ngẫu nhiên**

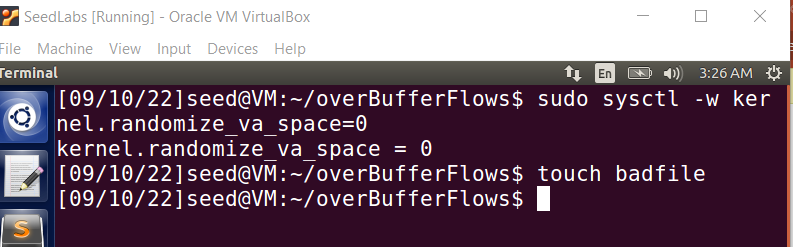


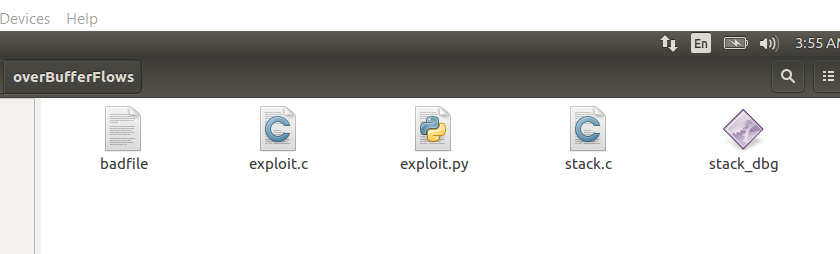
**Step 2: Tìm địa chỉ của inject code**

$gcc -z execstack -fno-stack-protector -g -o stack\_dbg stack.c

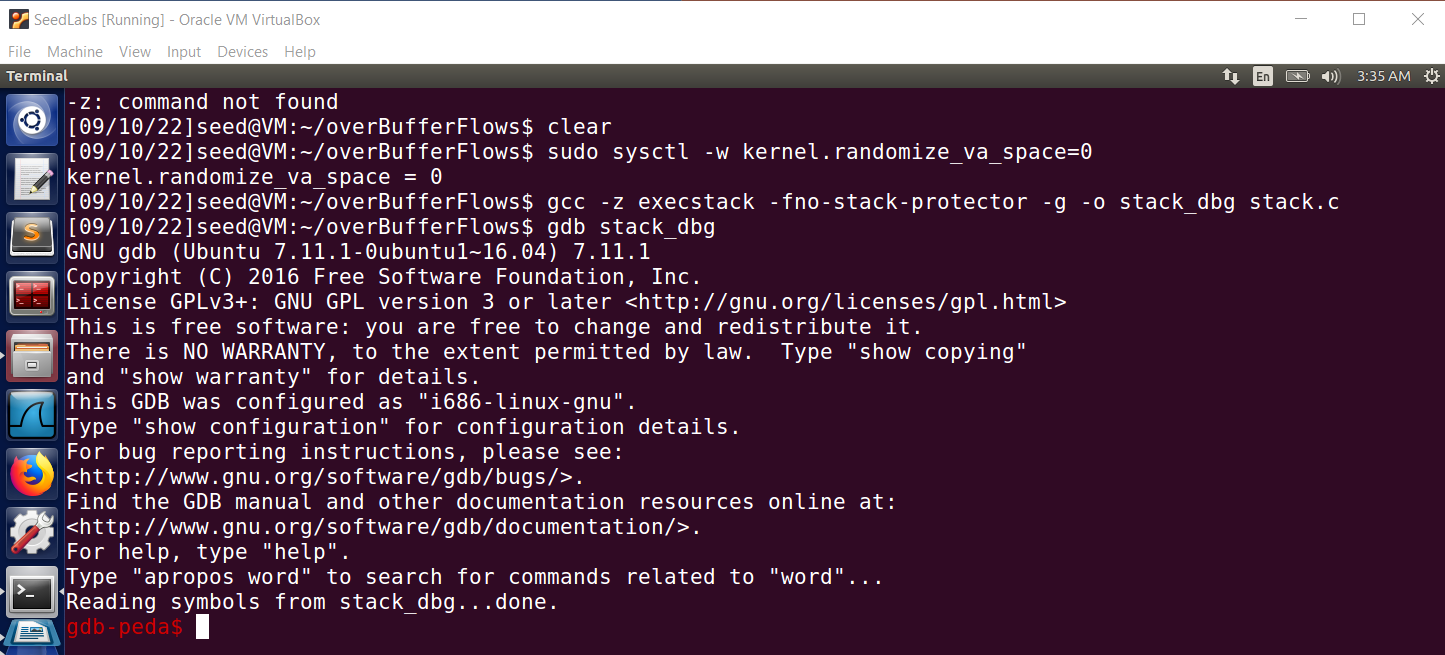


\_ Tạo 1 file trống để sử dụng touch: $ touch badfile

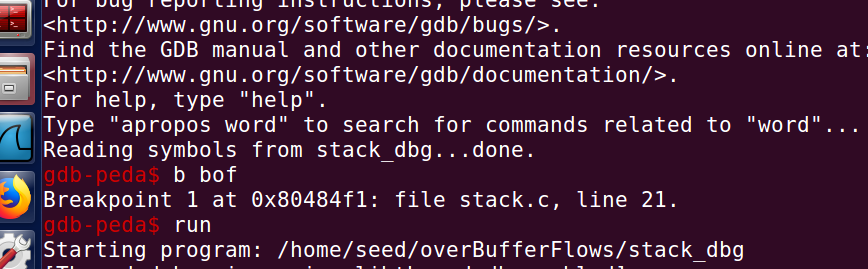
****



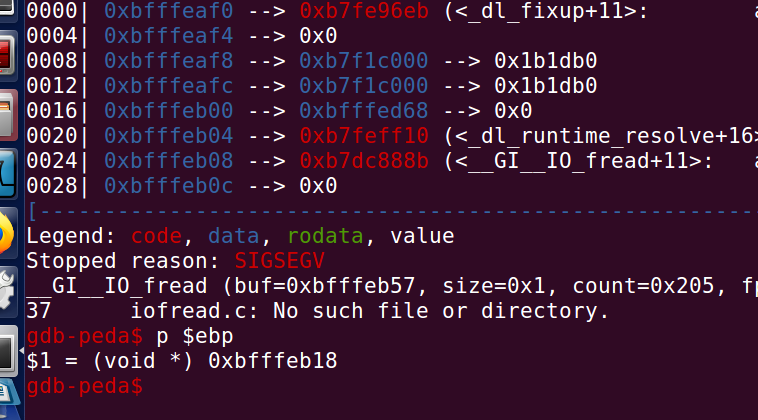
**\_ $gdb stack\_dbg :** chương trình sau khi được biên dịch stack\_dbg sẽ được debugged bằng cách sử dụng gdb

****

Đặt một breakpoint tại hàm bof và sau đó run

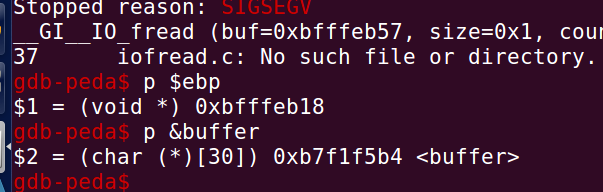


Tiếp theo tìm địa chỉ ebp: p $ebp



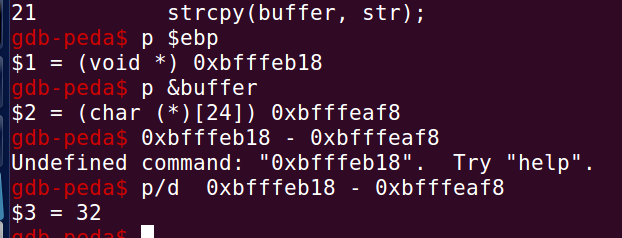
$1 = (void \*) 0xbfffeb18

Sau đó xác định địa chỉ của buffer



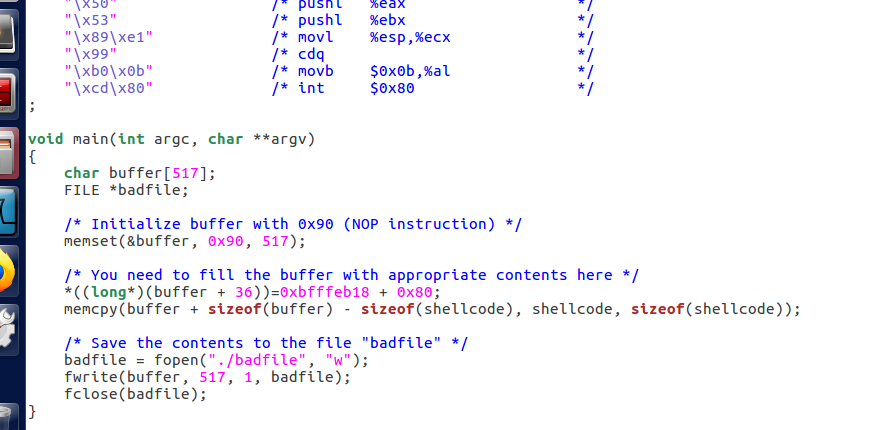
$2 = (char (\*)[30]) 0xbfffeaf8

Ta lấy hiệu 2 địa chỉ này để tìm khoảng cách:

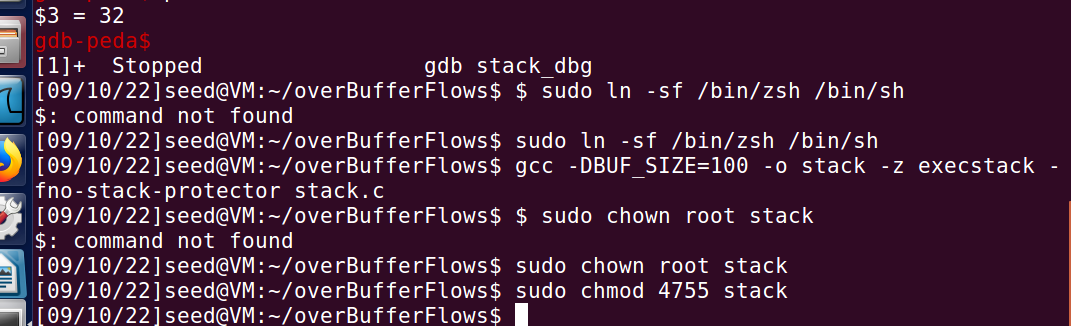


**Address = ebp + ( 32 + 4) = ebp + 36**

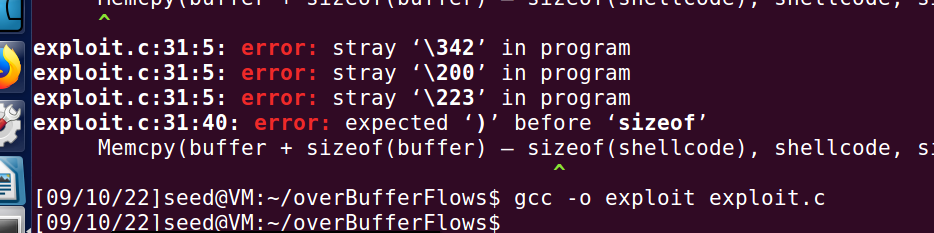
**Step 3: Sửa đổi code trong file exploit.c**

****

**Bước 4: Thực thi, biên idhjc file stack.c tạo ra file stack**

****

**Tiến hành gán các quyền cho stack**

****

kết quả

