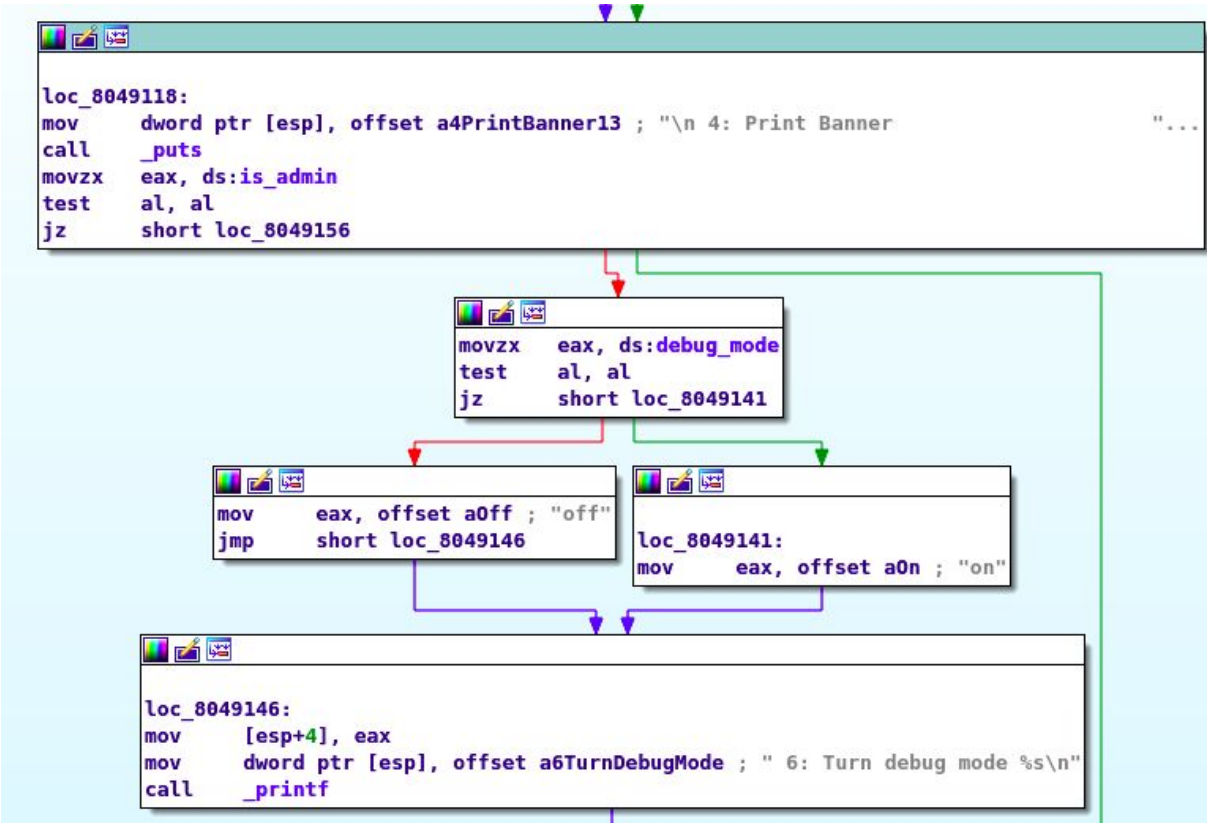
1/Tổng quan về chương trình:

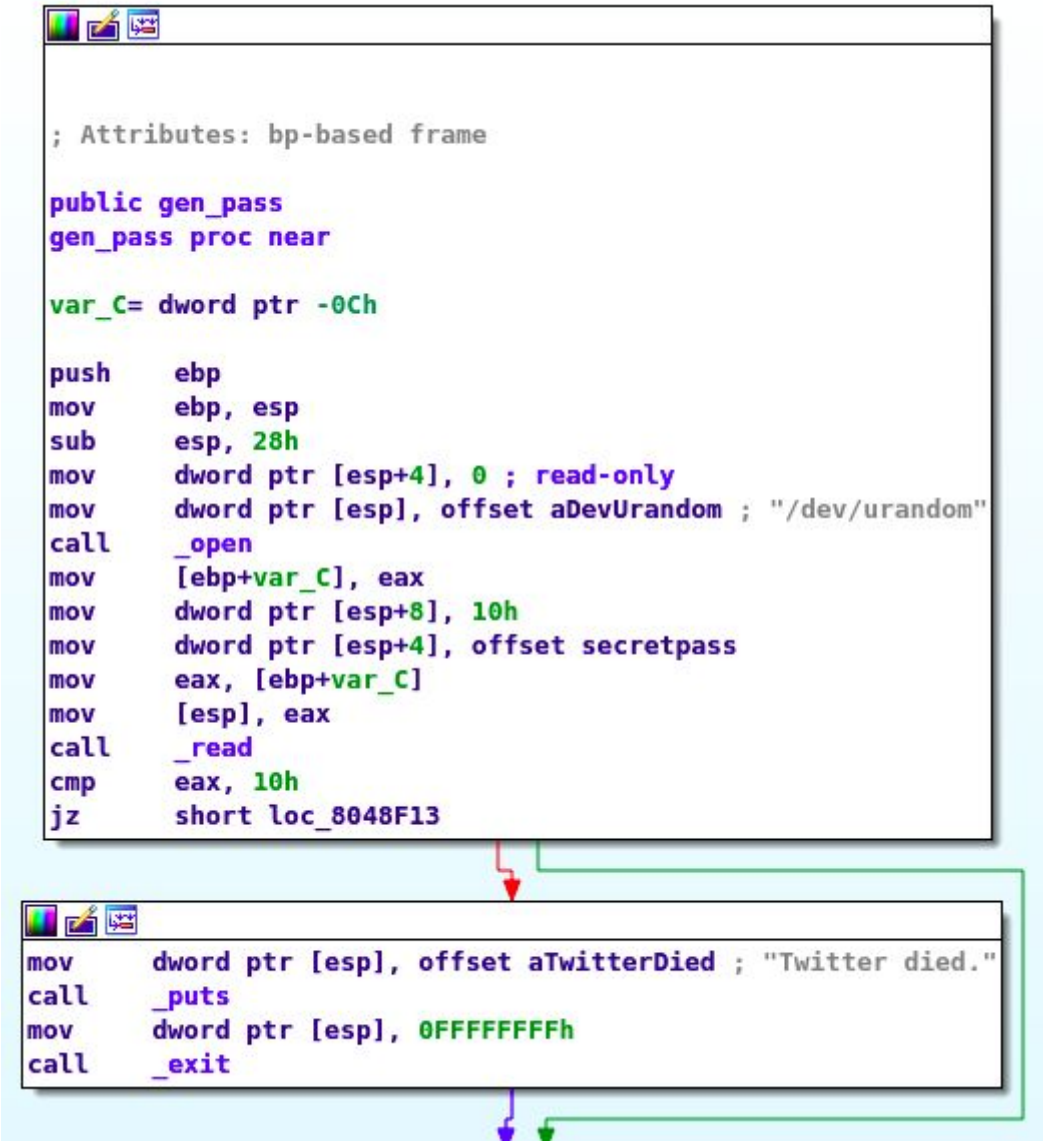
Chương trình sẽ bắt chúng ta nhập 2 giá trị user và salt để tạo ra 1 mã hash và sau đó hiện ra 4 chức năng trong đó

+Tw33t: cho phép chung ta nhập 16 bytes
+View Chainz:in ra những tw33t đã nhập

Nhưng khi mở bằng ida thì ta thấy nó còn có thêm 2 chức năng nữa là maybe_admin(bị ẩn) và debug_mode chỉ được hiện ra khi có có quyền admin



Để đăng nhập dưới quyền admin chúng ta sẽ cần phải nhập secret_pass khi mở bằng ida thì ta thấy secret_pass là 1 giá trị ngẫu nhiên được tạo bởi /dev/urandom

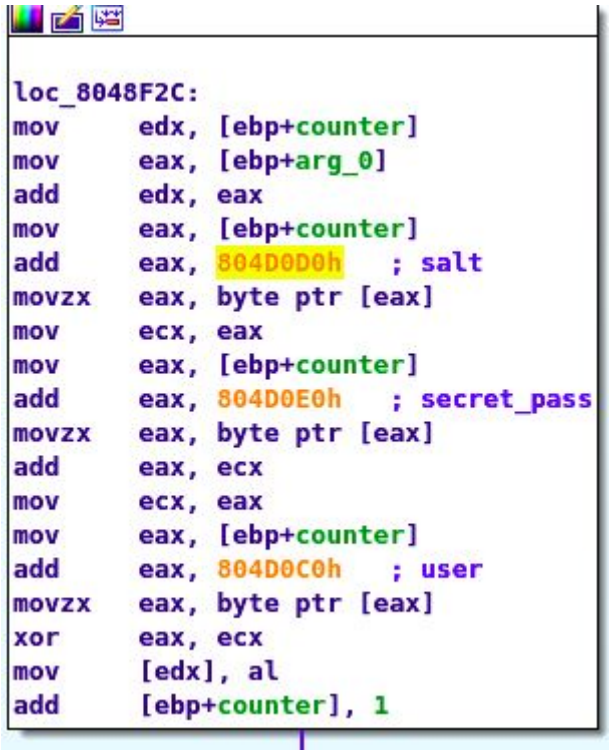


Điều đó làm cho việc chúng ta biết được secret_pass gần như bất khả thi nhưng sau khi xem các function khác thì ta có thể thấy secret_pass được dùng với username và salt để tạo ra giá trị hash được in ra trên màn hình lúc ban đầu

```
Enter Username:
user
Enter Salt:
salt
Generated Password:
ebb20a0a82a8b1a2e44e1802fe11574f
```

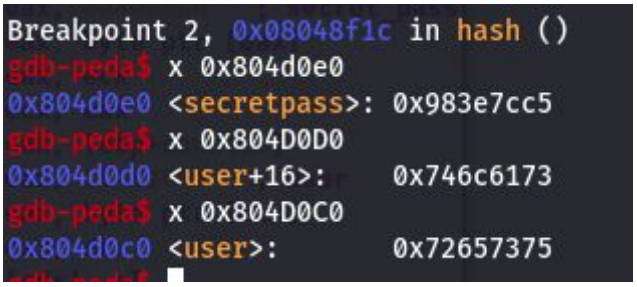
2/Password hashing

Sau khi nhập xong username và salt thì chương trình sẽ bắt đầu tính mã hash và in nó ra dưới dạng hexa bằng function print_pass .Và giờ chúng ta sẽ đi vào trong hàm hash để xem nó hoạt động như nào



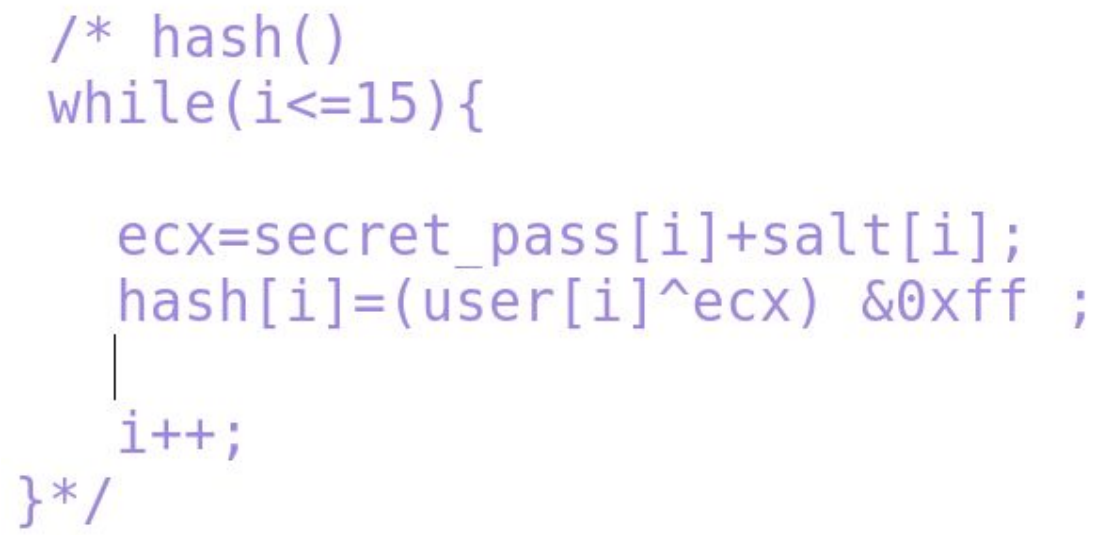
```
loc_8048F2C:
mov     edx, [ebp+counter]
mov     eax, [ebp+arg_0]
add     edx, eax
mov     eax, [ebp+counter]
add     eax, 804D0D0h   ; salt
movzx   eax, byte ptr [eax]
mov     ecx, eax
mov     eax, [ebp+counter]
add     eax, 804D0E0h   ; secret_pass
movzx   eax, byte ptr [eax]
add     eax, ecx
mov     ecx, eax
mov     eax, [ebp+counter]
add     eax, 804D0C0h   ; user
movzx   eax, byte ptr [eax]
xor     eax, ecx
mov     [edx], al
add     [ebp+counter], 1
```

Ở đây mình biết được các giá trị tại các ô nhớ kia là gì bằng cách set 1 breakpoint vào function hash và xem các giá trị kia là gì



```
Breakpoint 2, 0x08048f1c in hash ()
gdb-peda$ x 0x804d0e0
0x804d0e0 <secretpass>: 0x983e7cc5
gdb-peda$ x 0x804D0D0
0x804d0d0 <user+16>:    0x746c6173
gdb-peda$ x 0x804D0C0
0x804d0c0 <user>:       0x72657375
```

Sau khi reconstruct hàm này lại dưới dạng C thì ta có thể thấy cách hoạt động của hàm hash này



```
/* hash()
while(i<=15){

   ecx=secret_pass[i]+salt[i];
   hash[i]=(user[i]^ecx) &0xff ;

   i++;
}*/
```

Vì nó khá là đơn giản nên chúng ta có thể dễ dàng đảo ngược nó lại để tìm ra secret_pass

```
while(i<=15){
int secret_x

   ecx=user[i]^hash[i];
   secret_x=(ecx-salt[i]);
   secret[i]=secret_x &0xff;
   i++;

   }
```

Ở đây mình & 0xff là để lấy ra 1 chính xác 1 bytes vì khi mình làm thì có 1 số trường hợp khi thực hiện phép trừ ở trên nó sẽ cho kết quả là số âm và kết quả số âm đó sẽ được biểu diễn dưới dạng 4 bytes (size của int trong C)

3/Đăng nhập dưới quyền admin:

Sau khi tính secret_pass thì chúng ta sẽ có thể đăng nhập dưới quyền admin nhưng mà có 1 vấn đề là secret_pass có nhiều giá trị không nằm trong khoảng 0x21- 0x7E tức là chúng ta không thể nhập bằng bàn phím được (nếu chưa hiểu vì sao thì các bạn nhìn vô bảng ascii sẽ rõ)

```
0x804d0e0 <secretpass>: 0x6ba79414      0x5537b07d      0x0dc16440      0x7634de2d
```

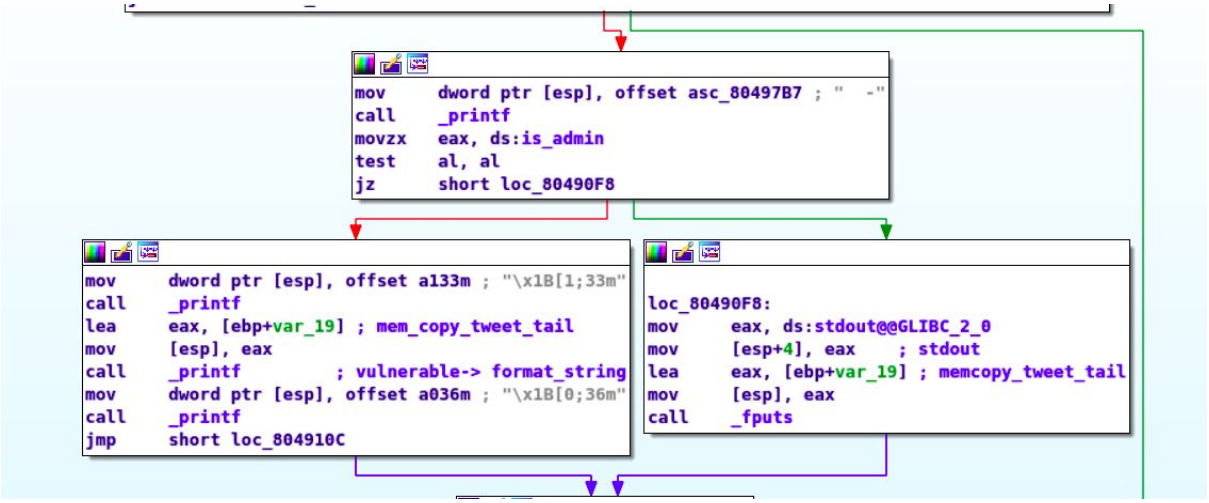
Và vì giá trị hash mỗi 1 lần sẽ khác nên mình không thể tính secret_pass bằng cách tạo chương trình A trên code block bằng C sau đó tiêm nó vào bằng cách lấy output của 1 chương trình A làm input cho 1 chương trình tw33tchainz qua câu lệnh | được :

```
python -c 'print(secret_pass")' | ./tw33tchainz
```

Điều này làm mình khá chật vật nhưng sau 1 thời gian tìm kiếm thì mình biết được pwntools 1 framework của python giúp cho chúng ta trong quá trình khai thác và nó hầu như xuất hiện trong tất cả bài CTF nên học cách dùng nó thông qua các bài giải CTF cũng là 1 cách

4/Lỗ hỏng:

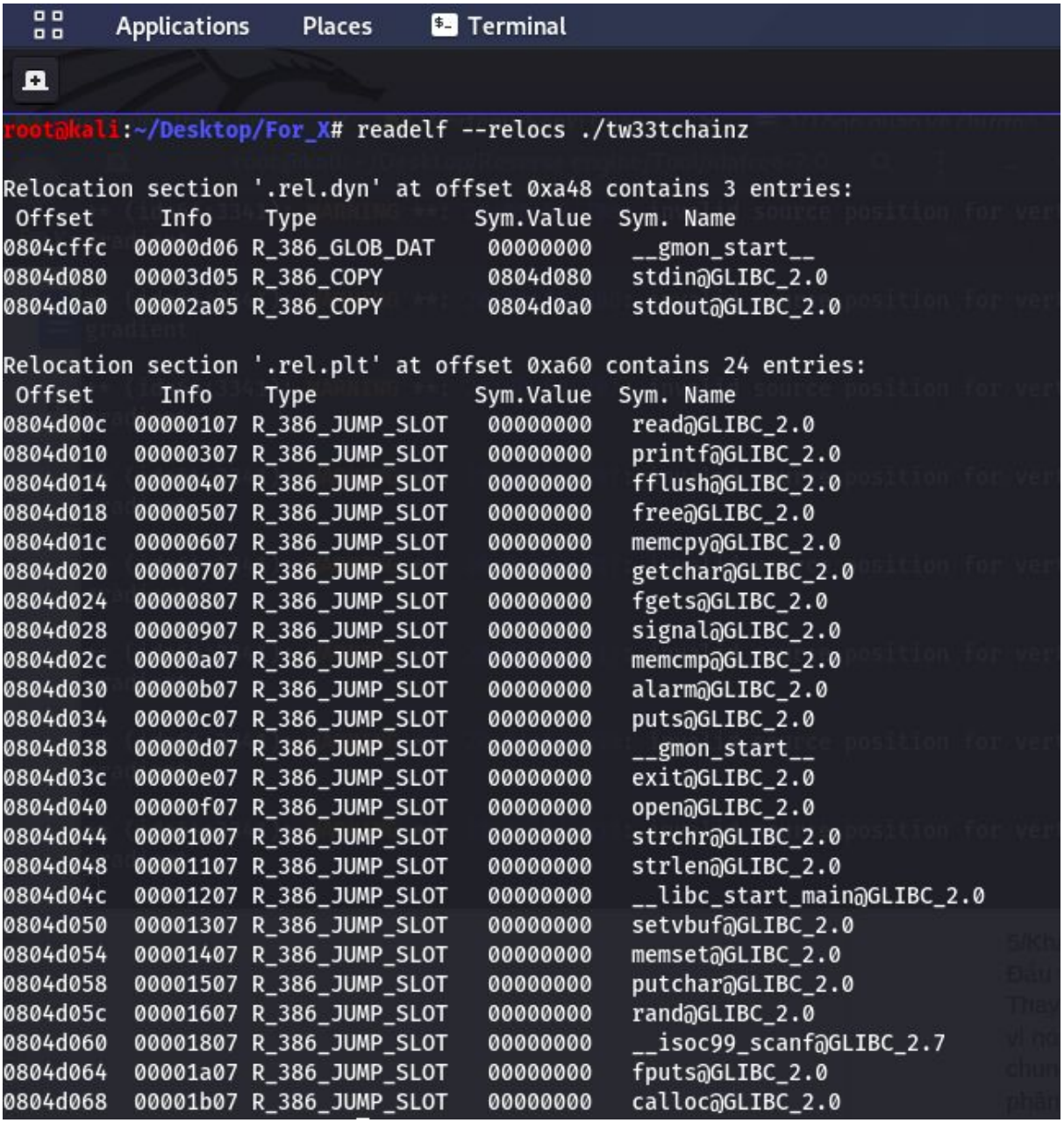
Sau khi nhìn qua hết tw33tchainz bằng ida thì tôi phát hiện ra lỗ hỏng trong tw33tchainz là string format năm trong function print_menu



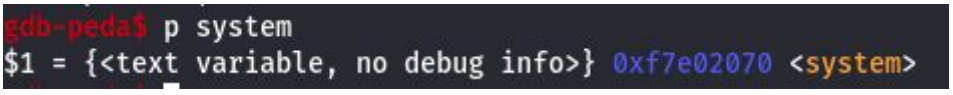
Nói ngắn gọn về lỗ hỏng format_string thì ở đây chúng ta sẽ có thể đọc bất kì thông tin nào chúng ta muốn(%s) và ghi bất cứ giá trị nào chúng ta muốn (%n ghi 4byte %hn ghi 2 byte) miễn là chúng ta biết địa chỉ của nó
Còn nói dài hơn về format_string thì các bạn có thể tham khảo ở đây , đây là bài viết mà mình đã xem trong khi làm bài này
https://cs155.stanford.edu/papers/formatstring-1.2.pdf

Chúng ta sẽ lợi dụng những tính chất này để ghi đè địa chỉ của fputs() thành system() bởi vì nó khá là thuận tiện trong trường hợp của bài này
Khi nhìn vào phần gọi fputs() ở trên chúng ta có thể thấy nó sẽ lấy giá trị chúng ta nhập vào(tweet_tail) làm tham số bây giờ chúng ta chỉ cần ghi đè địa chỉ của fputs()--> system() thì khi chúng ta nhập vào /bin/sh thì chương trình sẽ gọi system(/bin/sh) để tạo ra 1 process để thực hiện các shell command của bạn

5/Khai thác:
Đầu tiên chúng ta cần nói 1 xíu về GOT(global offset table):
Thay vì để 1 địa chỉ cố định cho các thư viện , các function được dùng bởi chương trình bởi vì nó sẽ có 1 vài xung đột khi 2 thư viện khác nhau của 2 chương trình khác nhau dùng chung 1 địa chỉ thì mỗi thì mỗi thư viện sẽ có 1 entry chứa địa chỉ của các function được phân bổ vào lúc chương trình được chạy và các địa chỉ này nằm trong GOT
Một điều làm cho GOT có thể được tận dụng để khai thác bởi format string là địa chỉ của nó sẽ không thay đổi dù chúng ta có chạy chương trình bao nhiêu lần.Chúng ta có thể dùng readelf hoặc objdump để biết được địa chỉ của từng function trên GOT

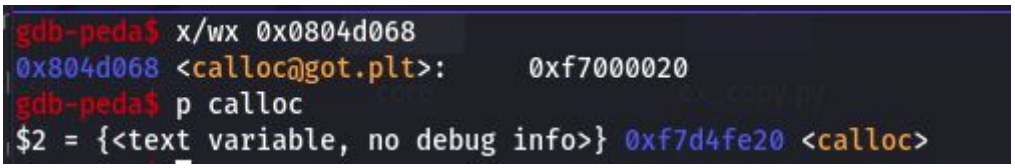


Đã có được địa chỉ của GOT nhưng chúng ta vẫn chưa biết được mình phải ghi gì bởi vì địa chỉ của function sẽ có khác nhau trên mỗi lần

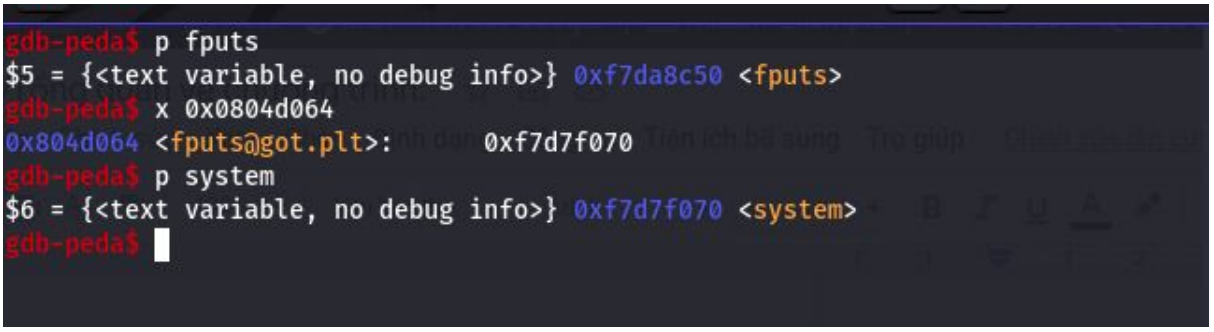


Nhưng dù địa chỉ có khác nhau nhưng khoảng cách giữa các function trong cùng 1 thư viện vẫn giống nhau vì vậy chúng ta có thể tính khoảng cách của system và 1 function A. Sau đó tận dụng tính chất đọc bất kỳ giá trị nào miễn là biết địa chỉ của lỗ hỏng format string để lấy địa chỉ của function A do đó
system=functionA + offset

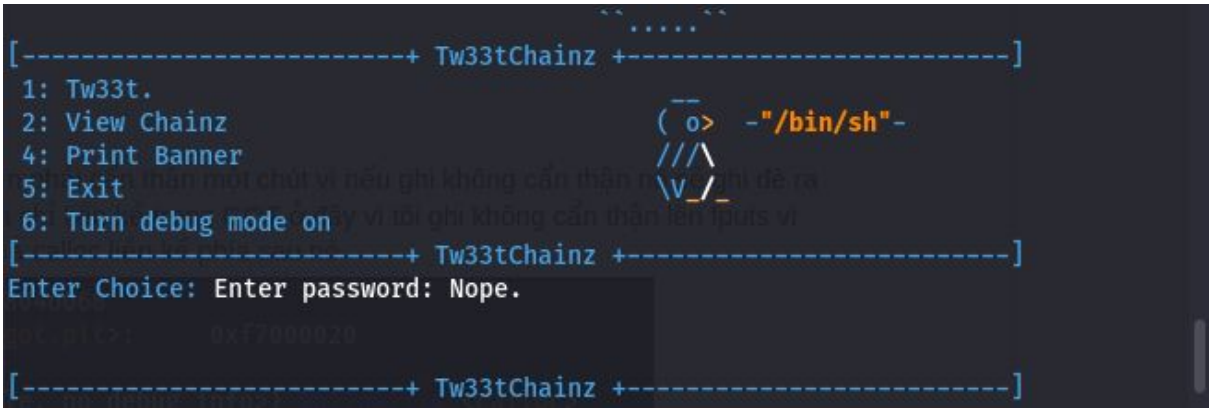
Ở phần ghi chúng ta cần phải cẩn thận một chút vì nếu ghi không cẩn thận nó sẽ ghi đè ra làm hỏng giá trị của địa chỉ liền kề trong GOT ở đây vì tôi ghi không cẩn thận lên fputs vì vậy đã ghi đè lên luôn cả calloc liền kề phía sau nó



Trở lại nội dung chính thì sau khi ghi đè xong địa chỉ của system lên fputs



Chúng ta chỉ việc nhập sai secret_pass để tắt quyền admin sau đó thêm /bin/sh để gọi system(/bin/sh) là xong chúng ta đã thành công



Và bây giờ thực hiện bất kỳ điều gì bạn muốn thôi