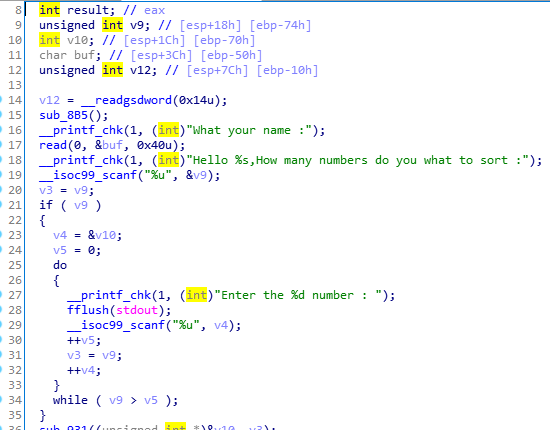
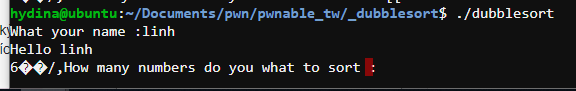
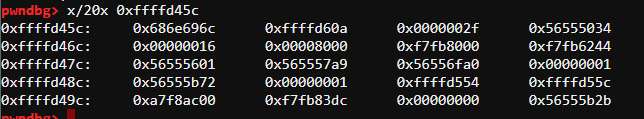
**Dubblesort**

****

Ở bài này, chương trình cho nhập 0x40 vào name và in ra màn hình. Sau khi chạy thử, ngoài những gì đã nhập, chương trình còn in ra một số ký tự lạ. Dự đoán có thể chương trình đã không gán “\x00” vào cuối chuỗi vừa nhập, dẫn đền hàm printf đã in ra một số ký tự trong stack đến khi gặp ký tự “\x00”.

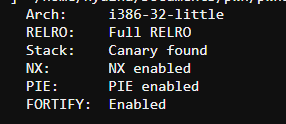


Kiểm tra bằng gdb thì đúng như đã dự đoán.



Ngoài ra, chương trình còn cho nhập n số nhưng lại không kiểm tra giá trị nhập vào có hợp lệ hay không.

Vậy là ta đã có thể tùy ý sửa chữa các giá trị trên stack



Nhưng ở bài này, chương trình bật full bảo vệ, nên muốn exploit thành công ta cần giải quyết các vấn đề:

1. Bypass canary

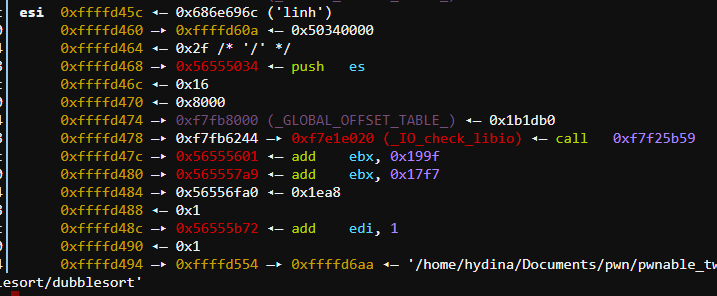
2. Tìm giá trị của libc

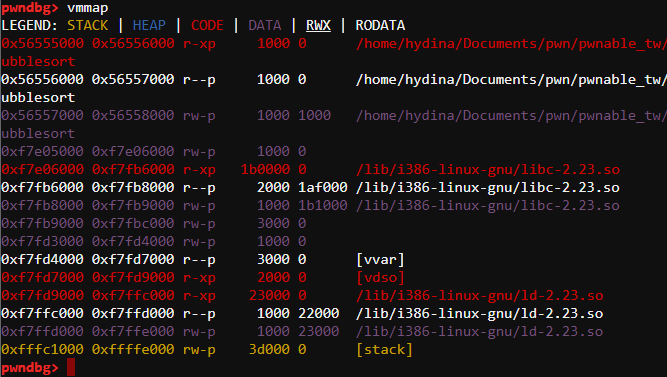
***Vấn đề 1:***

Ta chỉ cần nhập “+” hoặc “-” vào vị trí của canary là có thể bypass thành công.

***Vấn đề 2:***

Khi kiểm ra stack, mình phát hiện có địa chỉ của vùng got nằm trong khoảng nhập của biến name, tức là mình có thể leak được địa chỉ này. Nhưng ở đây mình đã không biết được cách để tìm ra địa chỉ libc\_base nên cần tham khảo write up bài này trên mạng. ( <https://drx.home.blog/2019/04/09/pwnable-tw-dubblesort/>)

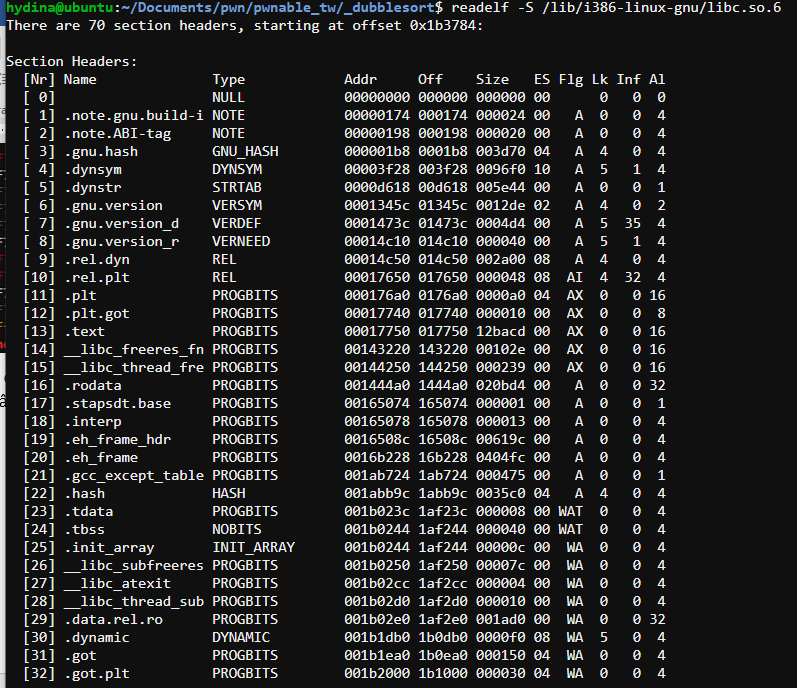




Từ địa chỉ 0xf7b8000 leak được, trừ đia địa chỉ 0xf7e06000 sẽ được offset từ phân vùng got đến libc\_base.

*Offset = 0xf7fb8000 - 0xf7e06000 = 0x1b2000*

Với cách làm tương tự, chúng ta có thể tìm được offset tới libc\_base của server.



Vậy là ta đã có đủ những thứ cần thiết để exploit.

Stack:

*ebp + 4: system\_add*

*system\_add*

*binsh\_add*

Và kết quả:

