**Challenge: ret2shellcode\_noPie**

- Kỹ thuật ret2shellcode là một nhánh của buffer overflow, nơi bạn không gọi một hàm có sẵn như trong ret2win, mà thay vào đó tự viết và nhúng shellcode vào payload, rồi hijack RIP để nhảy vào shellcode đó.

- Các chế độ bảo vệ ảnh hưởng đến ret2shellcode:  
A screenshot of a black screen

AI-generated content may be incorrect.

Với file chall:

**\* Dùng checksec để kiểm tra:**

A screen shot of a computer program

AI-generated content may be incorrect.

- No canary -> có thể buffer overflow

- NX ko bật -> có thể thực thi shellcode trực tiếp trên stack

- PIE: No PIE -> ko cần leak địa chỉ, các địa chỉ của file bin sẽ cố định

- còn lại ko ảnh hưởng đến ret2shellcode

**\* Xem pseudo code ở IDA xem, chương trình hoạt động ntn:**

- Có 3 hàm chính: run(), init(), main():

Mô tả chương trình: tại hàm main() trước tiên gọi hàm init() khởi tạo các bộ đệm để chuẩn nhập xuất tiếp đến gọi hàm run() nhập a1 và v2 rồi rồi return a1 kết thúc chương trình

A screenshot of a computer program

AI-generated content may be incorrect.

A screenshot of a computer program

AI-generated content may be incorrect. A computer screen shot of a program

AI-generated content may be incorrect.

**\* Nhận diện ret2shellcode:**

- Không có sẵn hàm system() hoặc hàm gọi shell trong chương trình

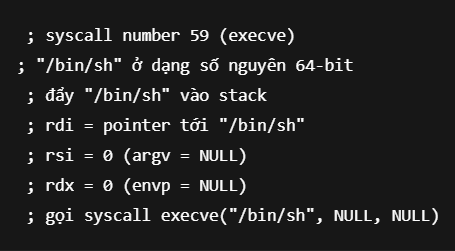
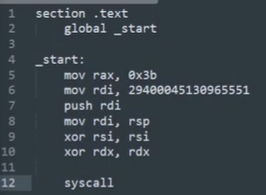
- Chương trình có đoạn code thực hiện hàm read() vào buffer trên stack

-> có thể dùng buffer overflow để điều khiển thanh ghi rip, inject shellcode vào chính buffer (đảm bảo NX bị tắt)

- Shellcode là gì?

Ta có đoạn code assembly sau:

Code Giải thích



Chức năng của đoạn mã này là gọi syscall execve(“/bin/sh”, NULL, NULL) để mở shell

A screenshot of a computer program

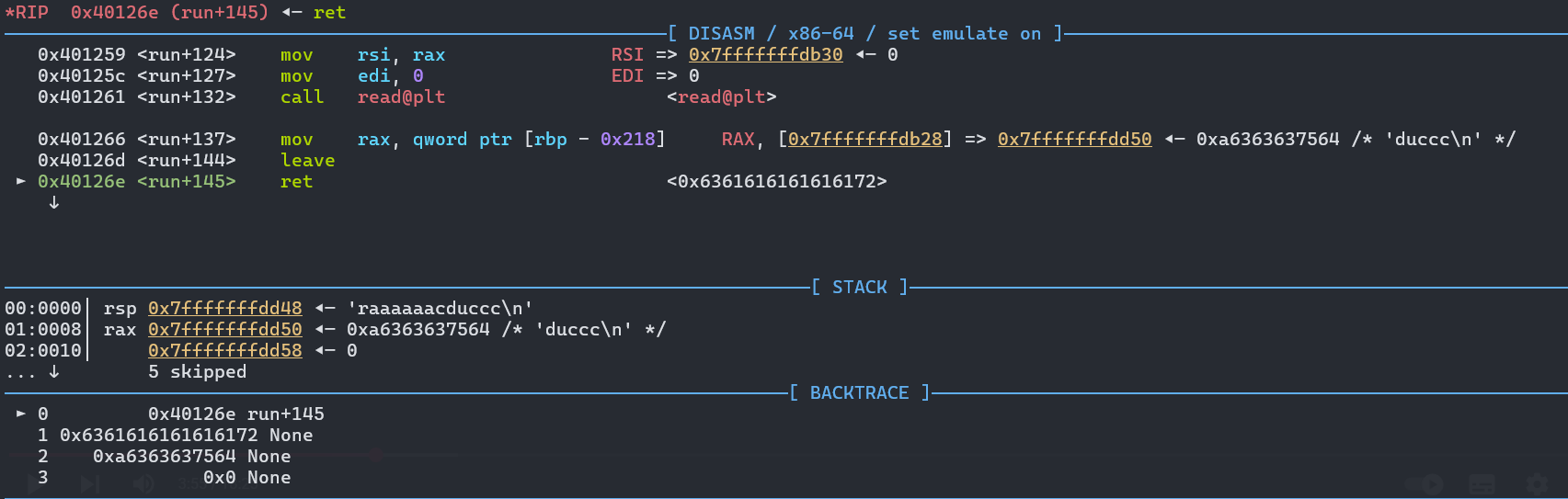
AI-generated content may be incorrect.

-> shellcode này những byte xác định trên, mỗi câu lệnh asm tương ứng với những byte xác định đó

-> Mục tiêu: cho chương trình return vào 1 con trỏ mà con trỏ đó trỏ vào shellcode chứa ‘/bin/sh’ của ta

**\*Khai thác bằng pwndbg:**

Theo ida ta thấy lỗi bof chỉ có ở chuỗi v2, chuỗi a1 khai báo đủ nên không có lỗi, ta nhảy vào hàm run() nhập chuỗi 544 kí tự bằng lệnh cyclic xem rip có bị đổi không



-> địa chỉ ret đã bị overwrite -> có quyền điều khiển chương trình

- Ta có thể truyền các byte cố định của mã asm lên stack và cho thực thi các byte đó để thay đổi return thành syscall execve chiếm shell

**\* Viết script shellcode và payload bằng python**

Ta có đoạn mã shellcode sau

A screen shot of a computer code

AI-generated content may be incorrect.

- Để ý chuỗi đầu tiên ta nhập nó nằm trên thanh ghi rax, nếu overwrite địa chỉ rax và địa chỉ của ret thì ta sẽ có thể thực thi đc shellcode ta để trong.

- Ta dùng tiện ích ROPgadget để tìm cái gadget nào đó của call rax or jmp rax

A screen shot of a computer code

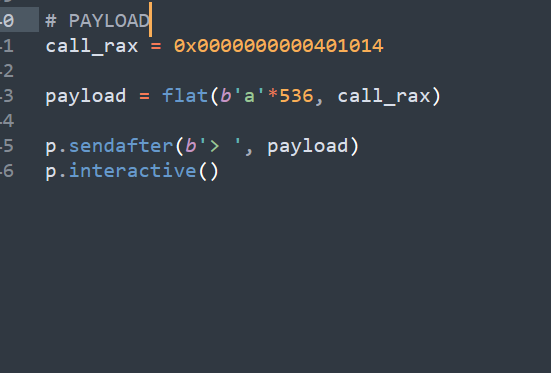
AI-generated content may be incorrect.

-> call\_rax = 0x0000000000401014 ; jum\_rax = 0x000000000040110c

- Viết script:

+ ở lần nhập đầu tiên ta sẽ truyền shellcode vào bằng sendafter

+ ở lần nhập thứ 2 ta truyền vào payload return đến call\_rax



p/s: ta truyền vào 536 byte rác vì đã truyền chuỗi “/bin/sh\0” bao gồm 8 byte rồi nên cần truyền thêm 544 – 8 = 536 byte

**\* Kiểm tra lại bằng debug động xem đã gửi shellcode với payload đúng chưa**

- Kiểm tra shellcode đã được truyền vào tại thanh ghi rax chưa:  
A screen shot of a computer program

AI-generated content may be incorrect.

-> đã giống với đoạn asm mình viết

A computer screen shot of a program

AI-generated content may be incorrect.

- Giờ gửi solve.py thử: A screen shot of a computer

AI-generated content may be incorrect.

=> Đã chiếm đc shell