Dreamhack Systemhacking basic\_exploitation\_000 문제 풀이

문제 파일 다운로드 후 문제를 살펴보겠습니다.

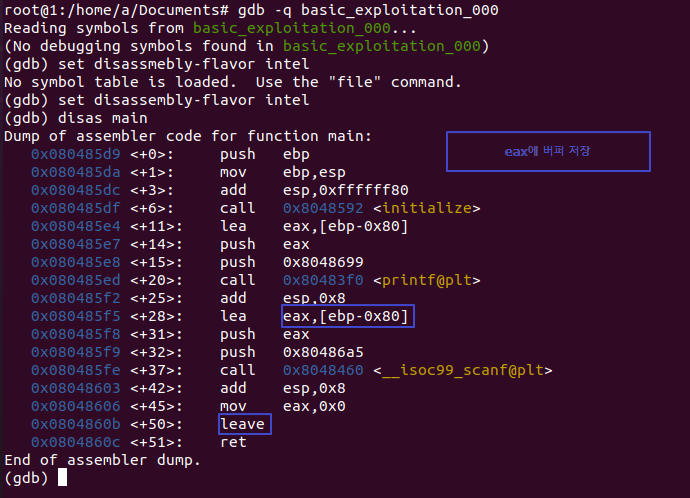


먼저 문제를 다운받은 후 VM 우분투로 옮겨 파일을 열어보았습니다.

Main 함수안에 버퍼가 0x80으로 선언되어 있고 buf를 출력 후 scanf로 사용자 입력값을 받는다는 것을 확인할 수 있습니다. 선언된 값의 크기보다 사용자 입력값이 더 크다는 것을 알 수 있었고,

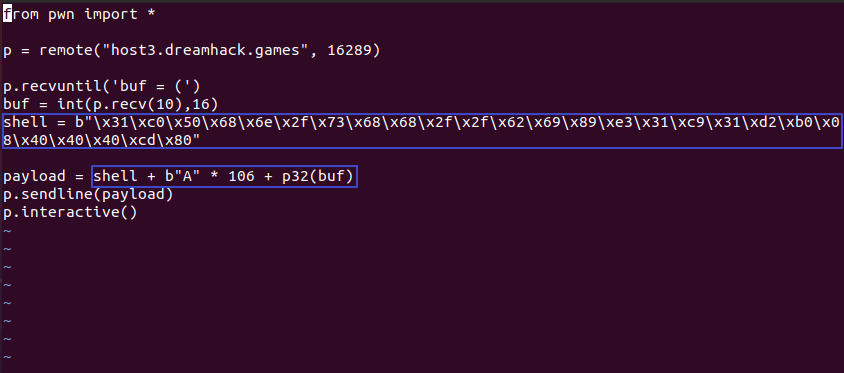
BOF 취약점이 발생할 수 있다는 것을 인지했습니다. return값을 바꿔줘야 하기 때문에 128byte + 4byte(sfp)를 더한 132byte를 채워줘야 합니다. 25byte의 쉘코드를 일반적으로 쓰지만 scanf 함수는 25byte의 쉘코드(\x09, 0x0a, \x0b, \x0c, \x0d, \x20)를 읽지 못하므로 26byte의 쉘코드를 썼습니다.

Gdb로 자세하게 살펴보겠습니다.



Gdb로 살펴본 결과 0x80의 크기만큼 eax에 저장되는 것을 알 수 있습니다.

페이로드를 작성해보도록 하겠습니다.



p.recvuntil() 함수는 괄호 안에 있는 부분까지 데이터를 받는 함수입니다. 개행 으로 쓸 시에는 개행 까지 받으므로 한 줄을 받아 오기 때문에 조심하는 게 좋습니다.

p.recv() 함수는 데이터를 받는 함수입니다. Int로 선언을 해준 후 10byte를 16진수로 받겠다는 의미 입니다.

26byte의 쉘코드를 선언해줍니다.

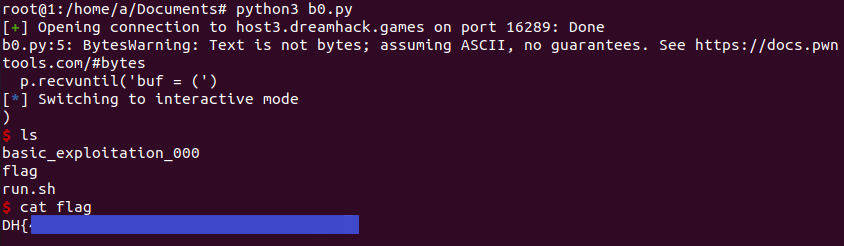
Payload 함수에는 선언한 쉘코드에 106 byte의 더미 값을 입력합니다.

p32() 함수는 32bit 리틀 엔디안으로 설정할 때 사용합니다.

그 후 p.sendline() 함수로 데이터를 보내주고

p.interactive() 함수는 쉘과 직접적으로 명령을 전송, 수신할 수 있는 함수 입니다.

이렇게 페이로드를 작성한 후 실행해보도록 하겠습니다.



Pwntools가 python3에서 동작하는 버전이기 때문에 python3로 실행해 주었습니다. 실행결과 flag값을 획득할 수 있었습니다.