**Pwnable.kr**

**- bof -**

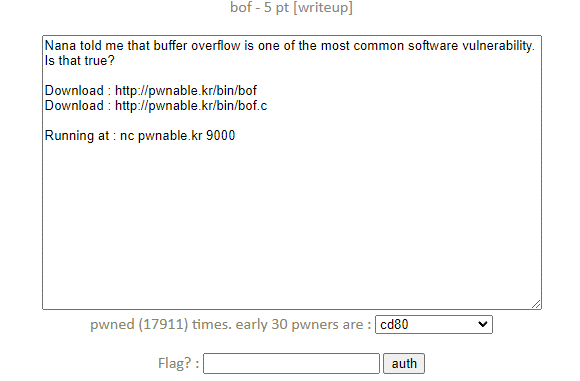
**Download : http://pwnable.kr/bin/bof**

**Download :** [**http://pwnable.kr/bin/bof.c**](http://pwnable.kr/bin/bof.c)

**nc pwnable.kr 9000**

**pw : guest**

**0. 문제 살펴보기**



문제에서 버퍼 오버플로우에 대한 얘기를 하고 있는 것을 알 수 있다.

이전 문제들과는 다르게 파일을 직접 다운로드하여 그 답을 해당 원격 접속으로 입력하는 방식인 것 같다.

**1. 파일 다운로드 및 살펴보기**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

텍스트, 폰트, 스크린샷이(가) 표시된 사진

자동 생성된 설명

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명

wget 명령을 이용하여 파일을 다운로드하고 bof.c 파일을 살펴보았다.

overflowme라는 배열이 32의 크기로 선언되어 있고, 입력을 받는데 그 크기 제한이 없는 것을 확인할 수 있다.

우리는 이 부분을 이용하여 버퍼 오버플로우를 걸어 공격을 하면 될 것 같다.

**2. gdb**

텍스트, 스크린샷, 폰트이(가) 표시된 사진

자동 생성된 설명



스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

gdb를 이용해 bof 파일을 열고 문법을 intel식으로 바꾼 다음 func 함수를 disassemble 하였다.

main 함수에서 func 함수로 인자를 넘길 때, 그 인수는 스택에서 return address 위에 저장된다.

<func+35>에서 gets 함수를 호출하기 전, ebp-2c 위치를 eax 저장하는 것을 보니 이곳이 overflowme 배열인 것 같다. 2c는 10진수로 44이고 배열의 크기가 32인 것을 고려하면 중간에 dummy 값이 12인 것을 확인할 수 있다.

따라서, 이를 바탕으로 간단한 스택을 그리면 다음과 같다.

텍스트, 스크린샷, 폰트, 번호이(가) 표시된 사진

자동 생성된 설명

**3. 공격**



그렇다면 우리는 overflowme의 시작주소로부터 시작하여 0xdeadbeef를 0xcafebabe로 뒤덮으면 될 것이다.

그 길이는 overflowme의 상대적 위치가 44이므로 44 + SFP + RET의 크기를 합하여 52라는 값을 알 수 있다.

따라서, 앞에 52만큼의 크기는 의미 없는 값으로 채운 뒤, 0xcafebabe를 리틀 엔디안 방식으로 파이썬 출력을 통해 문장을 생성하고 cat으로 이 값을 잡아준 뒤에 원격 접속을 시도하였다.



접속 후에는 아무런 말이 뜨지 않는데, 이 때, id 명령을 사용해 보니 bof 유저의 권한을 얻은 것을 확인할 수 있다. /bin/sh이 실행되었다는 것이다.



이후, ls 명령을 통해 현재 디렉토리의 파일들을 확인해보니, 우리가 이전까지 보았던 flag 파일이 있었고, cat 명령을 통해 flag 파일을 볼 수 있었다. 이후, 우리는 flag 값을 얻을 수 있다.