Assignment –3

Binary Exploitation - Stack based buffer overflow exploitation

- This assignment is designed to be intentionally vulnerable to multiple stack-based buffer overflow and is a gamified form of exploitation.
- Given are the files to replicate the scenario on your systems.
    a. Setup.sh
    b. printer.c
    c. References.txt
    d. Shellcode.txt
    e. Shellcodetester.c
    f. Dummy.pgn
- Downloads required for assignment
    a. Ubuntu 12.04 32-bit ISO for tutorial.
    b. Ubuntu ISO file 14.04/16.04/18.04 as per (mod 3) for your roll number. E.g. roll no%3 == 0 will download 14.04 and roll no % 3 == 1 will download 16.04 and so on for actual assignment.
    c. All must be in 32-bit version.
    d. For your current host system download VirtualBox and install the guest additions tools for better ease of use.
- Before coming for the tutorial VirtualBox must be installed with guest additions and Ubuntu 12.04.

About

The printer source code is a poorly written format parser for a PGN file format. This file format is used to represent a chess board's state in a game. Original PGN format supports multiple advanced tags but for this assignment only the tags mentioned in the dummy .pgn file will suffice. Program accepts a valid .pgn file as input and a function validates the input layout of board. To generate valid output, either dummy file can be used, or the python library is also provided in reference for the same.

Questions [To be performed on your assigned Ubuntu versions].

1 – Perform a stack based buffer overflow on the given binary **bin1** and achieve shell execution in root or normal privilege level. [30 marks].

2 - Create your own shellcode for seteuid() function and patch the provided the shellcode with your newly generated one and achieve root level shell execution on **bin2**. [20 marks].

3 – For **bin3** the ASLR protection will be turned on again. Come up with a strategy to bypass ASLR protection in your binary, this simply requires you to use commonly used technique and adapt it to your program. [25 marks].

For turning on the ASLR on again use command

echo 1 > /proc/sys/kernel/randomize_va_space

4 – For **bin4** executable stack protection will be enabled again. Perform a standard ret2libc attack on the binary. [25 marks]

Submission

For all questions kindly build a python code which can generate a malicious file for assisting you in exploitation and also a minor report with screenshots for each question.

Each question must have out-put of 'uname -a' command and MD5 hash for the binary of question in use at the top in a screenshot.

For testing around the binary for assignment you can generate a separate binary using gcc with –ggdb flags as well.

Setup

Run : **sudo bash setup.sh** in your virtual machine.

For an ideal initial setup output must look like this with our dummy file. This means all is ready for assignment.