GRENOBLE - INP ESISAR

OS 430
Lab - 3

Cyril Bresch

Supervisor : Arthur Desuert (*arthur.desuert@grenoble-inp.org*)

1 INTRODUCTION

During the last two labs, we have always worked with low level security binaries. Indeed, the stack was executable, the stack smash protector was disabled as well as the ASLR. The previous lab introduced the return to libc attack and demonstrated that the NX can easily be bypassed. In addition, we saw that the 32 bits ASLR has a very low entropy making it vulnerable to brute-force attack. However, for 64 bits binaries the ASLR is too high making the brute-force attack impractical. In this lab we will learn the technique of Return Oriented Programming. This advanced exploitation technique bypasses the ASLR, the non-executable stack and even the SSP in some cases.

1.1 ADVICE

1. Same as Lab 2.
2. Use vim and gdb.
3. Run away from exercise 3

1.2 DEFINITION

In the following exercises, the **padding** is defined as the number of bytes needed to reach the return address in the stack (without overwriting it).

2 Exercise : 'ROP for Dummies'

1. When a binary is compiled without the PIE option, which in-memory sections are not affected by the ASLR?
2. Pass root and type the following command:

   ```
   echo 2 > /proc/sys/kernel/randomize_va_space
   ```

   What is the effect of this command?
3. Find the overflow if it exists and give the **padding**.
4. Explain the Return Oriented Programming method. What is the ROPgadget tool?
5. Use the buffer overflow to make the binary jumping into the hidden function and validating the condition. Write an exploit script in Python
   Hint: You can use the data section.

3 Exercise : 'Return Oriented Programming'

1. Remind the main general purpose registers involved during a 64 bits execve syscall.
2. Exploit the security flaw to make the binary open a shell. Write an exploit script in Python. You can use the Pwntools library.
3. Explain how compiling with the PIE option complicates the implementation of a Return Oriented Programming attack. What would be the condition to set up such an attack in case the binary is compiled as a PIE?

4 Exercise : 'Hardened Binary PIE'

1. Now you are the analyst, find a way to exploit it.