Networks and Systems Security, Assignment 2

Due date: Feb 15, 2019
Total points: 45

Part I
Basic Buffer Overflow Vulnerability (30 points)

The objective of this assignment is to familiarize you with writing shellcode to exploit programs. You need to write a shell code using Intel X86-64 assembly language using any assembler of your choice (GNU AS, NASM etc.). The shellcode should print ``Hello World!'' on the terminal and thereby exit.

Once you have the shellcode ready you need to devise a way to pass it as an input to program (`victim`) so that at termination the input code is executed (as it normally happens with shell code execution).

The program is written for Linux/X86_64 architecture. The following stack smashing protection has have been disabled:

1. Address Space Layout Randomization (ASLR) turned off
2. Stack smashing protection (SSP) disabled
3. Enabled execution of code from the stack.

The program binary (`victim`) has been uploaded on backpack.

Grading Rubric

- Successful compilation of the shellcode using Makefile (5 points).
- Working standalone shellcode that uses system calls (victim) to print ``Hello World'' (10 points).
- Correctly passing the shellcode to the program forcing it to correctly ex-
ecute it (10 points).
- Description of the shellcode code, commands to test the shellcode and the assumptions that you made (5 points).

Part II
Basic ROP Exploit (15 points)

The objective of the assignment is to familiarize you with Return Oriented Programming (ROP). As a part of the assignment you discover how to use libc gadgets that we discussed in the class. Your objective is to create and pass an input shell code that actually has ROP gadgets which calls the execve() function of the libc library that eventually calls the `halt' program, which results in shutting down the system. You may search through libc library using objdump function or you may use the ROPgadget (https://github.com/JonathanSalwan/ROPgadget)
tool to search for gadgets (POP RET sequences)

The (victim) program is compiled for Linux/X86_64 architecture. The following stack smashing protection has have been disabled:

4. Address Space Layout Randomization (ASLR) turned off
5. Stack smashing protection (SSP) disabled
6. Execution of code from the stack turned off.

The victim program binary (`victim-nonexec-stack`) has been uploaded on backpack.

You may use the following URLs for reference:

https://crypto.stanford.edu/~blynn/rop/

https://github.com/finallyjustice/security/blob/master/rop/demo1/README.txt

Grading Rubric:

- Input which can actually the attack, involving the system call execve() (10 points).
- Description of the ROP shellcode input, how it works and the assumptions you made (5 points).

Submission guidelines:

Deadline 1: March 12, 2019 2359 Hrs: No points deducted.
Deadline 2: March 14, 2019 2359 Hrs: 5 points deducted.
Deadline 3: March 16, 2019 2359 Hrs: 10 points deducted.

Submissions after March 16, 2019 (2359 Hrs) would not be considered for being graded.