Binary exploitation assignment

Introduction

In this assignment, you will exploit a vulnerable program that creates a copy of a file. The binary accepts two command line arguments: the source file and the destination file.

Question 1

In this question, you have to attack the vulnerable copy program which has all protections disabled. Write a program to generate the exploit and store the exploit in a file named "exploit.txt". We will automatically invoke the vulnerable program with your exploit as input. Your exploit should read the file "flag.txt" and print it's contents. You have to use shellcode injection to display the contents. We recommend that you use the shellcode you wrote in the second assignment.

How we will test your program

We will simply invoke the run.sh file using "sh". Be sure that your run.sh file is compatible with "sh". We will test your submissions on Ubuntu 14.04 and by default, Ubuntu symlinks sh to dash. If you are using a different Linux/Unix based OS, be sure to test with dash. For most simple commands, dash is mostly a drop-in replacement for sh but if you run some fancy commands in run.sh, expect issues. We will not pass any arguments to the run.sh file.

Question 2

In this question, you have to attack the vulnerable copy program which has no execute protection enabled(i.e. all data regions such as stack are marked non-executable) but ASLR disabled. Write a program to generate the exploit and store the exploit in a file named "exploit.txt". We will automatically invoke the vulnerable program with your exploit as input. Your exploit should read the file "flag.txt" and print it's contents. You can use any technique to display the contents(the simplest would be what we discussed in the lectures).

How we will test your program

We will simply invoke the run.sh file using "sh". Be sure that your run.sh file is compatible with "sh". We will test your submissions on Ubuntu 14.04 and by default, Ubuntu symlinks sh to dash. If you are using a different Linux/Unix based OS, be sure to test with dash. For most simple commands, dash is mostly a drop-in replacement for sh but if you run some fancy commands in run.sh, expect issues. We will not pass any arguments to the run.sh file.

Note: Please do not use the skill you develop after completing this assignment to attack real world applications. If you do find security bugs in real world applications, please let us know and we will help you disclose it responsibly. If you don't do so, the software vendor can sue you and bad things will keep happening to you after that.