CPSC 525/625: Principles of Information Security (F19)
Instructor: Ryan Henry <ryan.henry@ucalgary.ca>

UNIVERSITY OF CALGARY

Name: _____ UCID: _____

Assignment 02

Due 2019-10-14 @ 11:59

1. On D2L, you will find six (horrendously stupid) C programs—`overflow.c`, `format1.c`, `format2.c`, `format3.c`, `env.c`, and `tocttou.c`—and a `Makefile`. Your task is to 1) provide carefully crafted inputs in order to fool each of the programs into spawning a shell (specifically, `/bin/SH` – note the capitalization), and 2) explain precisely how you arrived at your final exploit code (e.g., what *is* the vulnerability, by what process did you determine relevant memory addresses, etc.) and comment on how the vulnerability could be repaired.

   Rules for exploitation

   (a) Your exploits may not require modifications to the provided source code; they must work properly when the provided source code is compiled using the provided `Makefile`.

   (b) You must successfully exploit all six programs to receive full credit. **Eight marks** will be awarded for each successful exploitation, and an additional **two marks** will be awarded for your explanation of how you arrived at your exploit code.

   (c) There is a specific execution procedure for your exploit programs when they are tested (i.e., graded):

   - Your exploits will be run on `mocha.cs.ucalgary.ca`;
   - execution will be from a clean directory that contains only the six binaries produced by running `make all` (and the provided `shellcode.h`);
   - your exploits will be compiled by running `make sploits` (it is up to you to ensure the Makefile does the right thing when we type this);
   - exploits must not require any command line parameters;
   - exploits must not expect any user input;
   - if your exploit requires additional files, it must create them itself; and
   - by the time your exploit halts, it should have cleaned up any files create / rolled back any changes it made to the system during its execution.

   (d) Be polite. After ending up in a root shell, the person grading your exploit must still be able to exit the shell and continue grading without rebooting the virtual machine.

   (e) Give feedback. In case your exploit program might not succeed instantly, keep the user informed of what is going on.

   Note 1: The name of each program is highly suggestive of the type of vulnerability you are expected to exploit.

   Note 2: If you wish to run the exploits locally, it will be helpful to (temporarily) disable ASLR. Be sure to re-enable it when you are done! (In fact, you might want to work in a virtual machine…) The command to disable ASLR in Linux is `sudo sysctl kernel.randomize_va_space=0`; you can re-enable it with `sudo sysctl kernel.randomize_va_space=2`.

   Note 3: Some of you already have an account on mocha. For those of you who do not, please generate an SSH keypair and upload your public key to the SSH public key dropbox on D2L.