CECS 478 Assignment #7 - Buffer Overflow Lab

Due Date: 03 MAY 2018

20 points

Assignment Description. This assignment focuses on buffer overflow attacks and how they can be carried out on poorly-programmed system programs. You may not be able to complete this assignment on a modern operating system, as there are canaries built-in to modern shells (and kernels) to prevent such a thing from occuring. I would recommend using an older Linux distribution in a virtual machine for this assignment (something prior to Linux kernel version 2.4), but you are welcome to attempt this on a modern OS and see if you can get it to work. Review the article Smashing the Stack for Fun and Profit for a very good, detailed introduction on how to perform a stack smashing attack.

Assignment. Given the following C code file, perform a stack smash on the *vuln.c* code file using a C program that you create named *exploit.c*.

```
1  //vuln.c
2  #include <stdio.h>
3  #include <string.h>
4  int main(int argc, char **argv) {
5          // Make some stack information
6          char a[100], b[100], c[100], d[100];
7          // Call the exploitable function
8          exploitable(argv[1]);
9          // Return everything is OK
10         return(0); }
11
12 int exploitable(char *arg) {
13         // Make some stack space
14         char buffer[10];
15         // Now copy the buffer
16         strcpy(buffer, arg);
17         printf("The buffer says .. [%s/%p].\n", buffer, &buffer);
18         // Return everything fun
19         return(0); }
```

Note: when running many versions of Linux, you may need to disable some address randomization.

Honors Section. No special instructions for Honors students on this assignment, other than I will expect a more flawless program on execution. This assignment might be difficult to get right across multiple executions, so be prepared for some unusual flaws!

Deliverables. Submit your *exploit.c* to Beachboard Dropbox (*no compression!*) along with a writeup of how you attempted the stack smashing attack and screenshots of the output or result of a successful attack. If you are not able to succeed with the attack due to OS constraints, detail that in your writeup and explain how you would go about performing such an attack on this system (along with your C code).