UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

# COS 330

Practical 3

Specification

**Release Date: 19 September 2014 (16H30)**

**Due Date: 30 September 2014 (23H59)**

## Instructions

In this practical we are going to use **a linux environment** to discover the and generate a buffer overflow exploit as well as suggesting and implementing a countermeasure thereof.

Note: The aim of this practical is NOT to test anything other than Computer Security concepts that you are expected to know by now.

Upload a zip archive of *screenshots and/or code* of your answers onto the CS website by the above-mentioned due date. If you do not demonstrate your work in one of the practical sessions, then you will not be allocated any marks (even if you did upload), i.e. you must be present in person at the demo session to be able to receive a mark. *Also, you might need to re-run some of the commands during your demo should there be any uncertainties with your marker.*

**Note:** The statement printed by the showSecret() function must contain your student number.

***Everything that you submit might be checked for plagiarism. Instances of plagiarism will be dealt with in a serious manner.***

## Background

The act of storing more data than in a buffer than it was intended to hold results in buffer overflows. If this weakness is not mitigated it can lead to catastrophic effects. This practical will expose you to some of the practical issues thereof.

You have been provided with a C source file with basic access control functionality.
In a nutshell, it requests a password and calls one of two functions depending on the entered password (*showSecret*() and *unauthorized*()).

**NB:** Use this command to compile the program: gcc -fno-stack-protector -lcrypt -o filename.c

This program allocates an 8 byte buffer on which it stores the entered password. However, it fails to perform bounds-checking and as a consequence this causes the buffer to become vulnerable to overflows.

For background knowledge on the buffer overflow concept, see chapter 6 of the prescribed textbook.

## Task 1 [10 Marks]

For this task you are required to use an overflow exploit in order to cause a *false positive*. That is, you must use this attack to cause the program to see you as an authorized user (In this case just to call the showSecret() function).

You may need to load the program in **gdb** in order to track what happens to the stack. Visit http://www.spectrumcoding.com/tutorials for help.

## Task 2 [10 Marks]

Now you need to take a security professional's seat and solve the above-mentioned problem. Make sure your solution completely leaves any potential buffer overflow exploit impossible to perform. See the list of countermeasures in the prescribed textbook. Combining countermeasures may get you more marks, thus a working program will be awarded 6 out of 10. To earn a higher mark your program has to do more than just the basics.

**Upload Instruction: Put your code into a single document and upload a zipped copy of the document.**

---

## Total Mark: 20