School of Computing & Information Technology

CSCI862 System Security
Spring 2016

Assignment 2 (16 marks, worth 8%)

Due 11:59pm Saturday 24th September 2016

Part One: Short Answer — 13 Marks

For questions where you are referencing material outside the lecture notes, you should provide appropriate referencing.

1. Consider that you have two puzzles

   `Puzzle A:` One sub–puzzles. $k = 8$.

   `Puzzle B:` Four sub–puzzles. $k = 6$.

   For each puzzle provide

   (a) A graph of the distribution of the number of hashes needed. **1 Mark**

   (b) The average number of hashes needed. **0.75 Mark**

   (c) The standard deviation for the distribution of the number of hashes needed. **2 Marks**

   (d) Describe the method you used to obtain your solutions to (c). Don't go into too many details or show working, it's more "I wrote a C++ program to ... and then using ... I ...". **0.25 Mark**

   You should assume that if there are N possible solutions you check the N^{th} by hashing even if all others have failed and there has to be a solution.

2. Using a TCP SYN spoofing attack, the attacker aims to flood the table of TCP connection requests on a system so that it is unable to respond to legitimate connection requests. Consider a server system with a table for 1024 connection requests. This system will retry sending the SYN-ACK packet five times when it fails to receive an ACK packet in response, at 45 second intervals, before purging the request from its table. Assume that no additional countermeasures are used against this attack and that the attacker has filled this table with an initial flood of connection requests. At what rate (per minute) must the attacker continue to send TCP connection requests to this system in order to ensure that the table remains full? Assuming that the TCP SYN packet is 50 bytes in size (ignoring framing overhead). How much bandwidth does the attacker consume to continue this attack? **1 Mark**

3. What is a password mangler and why would we use one? **1 Mark**

4. What is a Cinderella attack? In particular, describe the target, the vulnerability being exploited, and the likely effect. **1 Mark**

5. Every hour the malware **X** spreads from each infected computer to two previously uninfected computers. In answering these questions you should explain how you determined your answers.

   (a) Give a table showing the number of infected computers at each hour across a 24 hour period. At time $t = 0$ the number of **X** infected computers is $N = 1$. **0.5 Mark**

   (b) By time $t = 10.5$ a counter worm **W** has been developed and it is deployed on one infected computer. **W** removes malware **X** from any host **W** is on. The counter worm **W** spreads slightly more quickly than **X**, with each **W** spreading to three **X** infected hosts each hour, provided such hosts are available.

   Provide another table showing the spread of **W** and the impact on **X** across a relevant time frame, starting from $t = 0$ again.

   Note the offset in time means that at $t = 10.5$ the number of **X** infected computers reduces by 1, so the spread of $t = 11$ will be slightly smaller than before. **1.5 Marks**

   (c) Graph the two cases against each other, clearly indicating on it where $N = 0$. **0.5 Mark**

   (d) Assume that at time $t = 12$, **X** evolves to spread to three uninfected computers each hour. What subsequently happens? **0.5 Mark**

6. In the context of phishing, list 8 points that can be used in checking the legitimacy of an email. Justify why each is appropriate as an indicator. Note that some points could relate to characteristics of legitimate messages, and others could be indicators of a phishing message. **2 Marks**

7. What protection is provided at the memory level by using the private access specifier in declaring a class in C++? Is it possible to overflow into private variables? **1 Mark**

Part Two: Buffer overflow 3 Marks

The Windows executable `overIT.exe` is given. The corresponding C/C++ code was compiled with Dev–C++ and is executable on PC's only. It should work happily in the lab. It can be used as follows:

`overIT your_student_id Additional_input`

where `your_student_id` is your UOW student number, possibly with an additional digit (or more) appended (see below), and `Additional_input` is an arbitrary string that will, in a normal run of the program, be echoed.

`overIT.exe` consists of some functions including a set of "bar" functions. Your task is to invoke it by altering the return address of a given function that improperly uses a C/C++ function that could allow a buffer overflow. When bar is invoked, it will print out a message: "I have been hacked!".

You need to find out the return address of the given function. Do not try to compare it with other students since it is dependent on your student ID. However, if you cannot alter the return address because the address contains unalterable numbers (such as a space, etc.), then you are allowed to append an additional digit or digits to your student number to make a different stack. Provide the number in your report.

When `overIT.exe` is executed, it will provide you with the address of bar and a list of addresses on the associated memory stack. This is the only information you have. You then hack the code in terms of your own inputs in the command line. To do so, you need to create a Perl file that contains your input.

Your solution must include the following information:

1. Your name and student ID.
2. The additional digit if you used it.
3. The stack contents before hacking and the memory address of bar, which will be printed out if you have used the code properly. Mark the location of the return address that you attempt to overwrite.
4. The output when you launch your successful buffer overflow. Remember there should be a message "I have been hacked!".

You should also submit the perl script `attack.pl` used in your successful attack.

Submission

Submission is by Moodle, in a single zip file without directories in it, by 11:59pm Saturday 24th September 2016.