Assignment

1. In the ***StackGuard*** approach to solving the buffer overflow problem, the compiler inserts a ***canary*** value on the memory location before the return address in the stack. The canary value is randomly generated. When there is a return from the function call, the compiler checks if the canary value has been overwritten or not. Do you think that this approach would work? If yes, please explain why it works; if not, please give a counterexample.

2. A thief walks up to an electronic lock with a 10-digit keypad and he notices that all but three of the keys are covered in dust while the 2, 4, 6, and 8 keys show considerable wear. He thus can safely assume that the 4-digit code that opens the door must be made up of these numbers in some order. What is the worst-case number of combinations he must now test to try to open this lock using a brute-force attack?

3. In a special case of a permutation cipher, we take a message, M, and write its letters in an $s \times t$ table, in a row-major fashion, and then let the ciphertext be a column-major listing of the entries in the table. For example, to encrypt the message ATTACKATDAWN, using a 3×4 table, we would write the message as
   ATTA
   CKAT
   DAWN
   and then write down the ciphertext as ACDTKATAWATN. The secret key in this cryptosystem is the pair (s, t). How is decryption done in this cryptosystem? Also, how hard would it be to attack this cryptosystem using a ciphertext-only attack?

4. Suppose Alfred has designed a client-side approach for defending against cross-site scripting attacks by using a web firewall that detects and prevents the execution of scripts that have signatures matching known malicious code. Would Alfred's system prevent the most common XSS attacks? Which types of XSS attacks are not detected by Alfred's system?