COM 402 exercises 2019, session 2: Web and software vulnerabilities

Exercise 2.1

You are writing a web application to sell Whisky. You want to save your customer's names in a database. You know that single quotes (') can break SQL request.

- What can you do to be able to accept single quotes in names and still have no problem with SQL injections.

Exercise 2.2

Consider the following code, taken from the Mitre Common Weakness Enumeration website (https://cwe.mitre.org). A web application has a function to run a backup of a database. The backup can be of type `full` or `incremental`. The type of backup is selected by the user and is sent to the server in the parameter named `backuptype`. The following code is used on the server:

```
...
String btype = request.getParameter("backuptype");
String cmd = new String("cmd.exe /K \"
c:\\util\\rmanDB.bat "
+ btype +
"&&c:\\utl\\cleanup.bat\"")

System.Runtime.getRuntime().exec(cmd);
```

- name the type of attack that could happen here and explain its possible consequences.

Exercise 2.3

Memory pages can be protected against writing or execution.

- explain why it is dangerous to have pages where both execution and writing are permitted

Exercise 2.4

At the end of a function call, two addresses are often popped from the stack.

- What are those two addresses used for ?

Exercise 2.5

Local variables are on the top of the stack and the return address at the bottom.

- How can a buffer overflow overwrite the return address that is below the variable on the stack?

Exercise 2.6

- Why must a stack canary have a random value ?

Exercise 2.7

Imagine a program where then name of the price of a product is stored in memory just above a variable containing the shipping address of the product.

Be entering an extra long shipping address, the customers are able to modify the price of the product and buy it for cheaper.

For each of the following protection methods, explain if it could prevent this attack:

- address space layout randomization (ASLR)
- marking the memory page as non executable or non writeable
- using a stack canary