1 Overall marking scheme

The coursework for COMP522 consists of two assignments, contributing to 25% of the final mark. The contribution of the single assignments is as follows:

<table>
<thead>
<tr>
<th>Assignment</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assignment 1</td>
<td>12.5%</td>
</tr>
<tr>
<td>Assignment 2</td>
<td>12.5%</td>
</tr>
<tr>
<td>TOTAL</td>
<td>25%</td>
</tr>
</tbody>
</table>

Failure in any assignment may be compensated for by higher marks in other components of the module.

This document describes Assignment 1. Assignment 1 will be marked according to the following broad criteria:

- correctness of the program;
- presence/absence of the report on the experiments;
- clarity of the arguments explaining the observed behaviour;
- original contribution in implementation, or analysis.

2 Aims of the Assignment 1

- to illustrate the practical complexity of brute-force search attacks on the password-based encryption;
- to test the students skills of using symmetric cryptography primitives in Java programmes;
- to test the students skills in the analysis of the experiments.
3 Brute-force search attack on the password-based encryption

This exercise asks you to write a program implementing password-based encryption and decryption, and then to extend it with the class(es) implementing brute-force search attack. You need to

• implement a program which takes an user password as the input and performs encryption of the predefined plaintext; then it asks the password again and decrypts the ciphertext;

• extend your program with the class(es) implementing brute-force search attack on your encryption/decryption procedure;

• assume that the attacker knows:
  – the predefined plaintext;
  – the ciphertext produced;
  – the salt;
  – the iteration count;
  – but no password.

• thus an attacker should iterate over all passwords of the given length $n$, encrypt the plaintext and compare the result with the given ciphertext;

• assume for simplicity, that the password consists of the decimal digits 0,...,9 only; you can implement your own iterator over (generator of) passwords (strings) or re-use one of the many available online, see for example

  https://stackoverflow.com/questions/9175976/generate-all-possible-string-from-a-given-length

In the latter case please don’t forget to acknowledge the use of third party’s code.

• estimate average time required to find a correct password for the predefined plaintext/ciphertext, fixed value of the salt, fixed value of the iteration count and small values of $n$, say $n \leq 6$.

• investigate how the search time depends on the iteration count value.

• formulate your recommendation on the length of the password, which would allow to withstand brute-force search attack for one hour (1 hour);

• perform the above for DES and AES encryption and conclude which (if any) is more resistant to the brute-force search attack.
4 Useful information

You may find it useful to have a look on the simple program implementing password-based encryption:
http://www.csc.liv.ac.uk/~alexei/COMP522/PBEs.java
JCE Reference Guide can be found at

5 Submission

You need to submit:

- Java code and compiled classes of your program(s)
- short report on experiments

The work must be submitted electronically by going to the Web page at
https://cgi.csc.liv.ac.uk/login.php
and follow the link “Coursework submission.” This must be done by

17.00 on Friday, November 9, 2018

Please be aware that the standard University policies

- on plagiarism, collusion and fabricated data
  www.liv.ac.uk/tqsdpol_strat_cop/cop_assess/cop_assess.doc, Section 8
  and
- on late submission
  www.liv.ac.uk/tqsdpol_strat_cop/cop_assess/cop_assess.doc, Section 6

are applied to this assignment.