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 2. Assignment 2 will be marked according to the following broad criteria:

- correctness of the program (40%);
- presence/absence of the report on the experiments (10%);
- clarity of the arguments explaining the observed behaviour (20%);
- analysis of the possible practical issues with the implemented scheme (30%).

Aims of the Assignment 2

- to illustrate the practical aspects of using asymmetric cryptography and hash functions for the message authentication;
- to test the students skills in programming with JCE/JCA;
- to test the students skills in the analysis of the experiments.
Message authentication using SHA-1 hash algorithm and RSA encryption

This exercise asks you to write a program in Java using JCE which implements message authentication protocol shown as a variant b) on the page 9 of lecture notes on Message Authentication and Hash Functions.

You need to implement a program which models activities of two participants, Sender and Verifier.

**Sender:**
- takes a text of the message;
- calculates the message digest (hash function) using SHA-1 algorithm;
- generates a pair of RSA private/public keys;
- encrypts the produced digest (hash) with a private key;
- passes the original message, encrypted digest and public key to the Verifier.

**Verifier:**
- decrypts the digest he has received from the Sender with the Sender’s public key;
- recalculates a new digest from the text of the message he has received;
- compares these two digests.

Extend your program with the third participant, **Adversary**, which is placed in between Sender and Verifier and may change either the original message, or encrypted digest, or both. Show that Verifier may detect Adversary’s attacks.

**Digital Signature Algorithm**

Implement a version of a program for the message authentication which utilizes Digital Signature Algorithm (DSA) available in JCA. See details on use of DSA in [http://docs.oracle.com/javase/7/docs/technotes/guides/security/crypto/CryptoSpec.html#SigEx](http://docs.oracle.com/javase/7/docs/technotes/guides/security/crypto/CryptoSpec.html#SigEx).

Show that again Verifier may detect Adversary’s attacks. Compare usability of two solutions: SHA-1+RSA and DSA.

**Report**

In your report please give a short explanation of your solution, the description of experiments and the explanation of the observed behaviour of the authentication schemes, the comparison of SHA-1+RSA and DSA schemes.
Useful information

You may find it useful to have a look on the simple program implementing RSA encryption/decryption with keys generated randomly:
http://www.csc.liv.ac.uk/~alexei/COMP522/RandomKeyRSAExample.java
and on the simple program generating a digest of the message using SHA-1:
http://www.csc.liv.ac.uk/~alexei/COMP522/MessageDigestExample.java
(there are links to these programs from the web page of the module)

Submission

You need to submit:

- Java code and compiled classes of your program;
- Report

The work must be submitted electronically by

17.00 on Friday, December 7, 2018

Please be aware that the standard University policies

- on plagiarism, collusion and fabricated data
  www.liv.ac.uk/tqsd/pol_strat_cop/cop_assess/cop_assess.doc, Section 8
  and

- on late submission
  www.liv.ac.uk/tqsd/pol_strat_cop/cop_assess/cop_assess.doc, Section 6

are applied to this assignment.