COMPX304-23A: Assignment 3

Individual Assignment
Submit on Moodle **Deadline: 23:00, Monday, 28th May, 2023 Weight: 12.5%**

Quantum Key Exchange

Symmetric-key cryptography is a popular method of securely communicating data between two entities. AES, ChaCha20, 3DES are examples of symmetric key encryption systems supported by TLS. While those are moderately complex algorithms a simple way to encrypt a message given a key is XOR encryption. This method repeatedly applies the XOR operation on the message using the key. The receiver performs the same operation to decipher the message.

For example, if the message is "0100110101100101" and the key is "001", the following processes take place:

```
Transmitter:
0100110101100101 (message)
0010010010010010 (key repeated)
0110100111110111 (XOR'ed cipher to be sent)

    Transmitter ---------------------------------> Receiver

Receiver:
0110100111110111 (incoming message)
0010010010010010 (key repeated)
0100110101100101 (decipher message)
```

Every type of symmetric key encryption, however, requires a shared key to be known between the two parties. But how does one transmit a key over an insecure channel? Sending it as plain text means it can be intercepted by a malicious eavesdropper and we can't encrypt it because it leads to a chicken-and-egg problem, since this operation would also require a key. There are many solutions proposed to solve this problem, each with its own assumptions and constraints. These solutions essentially build a secure channel within an insecure channel. A popular solution is the one used in TLS called the Diffie-Helman key-exchange algorithm.

Another innovative solution makes use of quantum computing. This solution is the subject of this assignment. Quantum computing manipulates quantum bits (qubits), instead of classical (regular) bits. Qubits are subject to quantum mechanical laws of physics. In this assignment, you'll implement and test the Quantum Key Exchange (QKE) algorithm. This algorithm is also known as the BB84 algorithm.

QKE assumes the existence of a quantum communication channel over which qubits can be transferred. Qubits are sent as photons and can be encoded via a photon's polarisation. For example, we can define counterclockwise polarization ↺ as 1, and clockwise polarisation ↻ as 0. However, this is not the only option, a photon could also be polarised in a linear fashion; thus, we can define an upwards polarization ↑ as 1, and a downwards polarization ↓ as 0. Because of quantum mechanics, the internal state of a qubit can only be measured by interacting with it through a polarization filter of a specific type.

The key quantum mechanical observation is that these two types of polarization (circular vs. linear) are orthogonal to each other. This means that if a photon is polarized in a circular manner, it has an equal 50-50 chance to be measured as ↑ or ↓ when measured linearly (and vice-versa). Consequently, in software and for the purposes of this assignment, a qubit can be modelled as instances of the following class:

Class Qubit
- int value
- int polarization **//0 for circular, 1 for linear**
+ new (value, polarization)
+ set (value, polarization)
+ measure (polarization): int **//Returns value if polarization matches.** **//Else, set the polarization to the new type. Set the value to 0 or 1 with 50/50 chance. Return the new value**

Based on these observations, the QKE algorithm works as follows (remember, the goal is to agree on a shared secret key):

1. The transmitter sends a stream of qubits. For each qubit it records the value and polarization type, which are both picked randomly with equal chances.

2. The receiver receives the stream of qubits. For each qubit it selects a random polarization type to measure it, and records the results

3. The transmitter and receiver exchange the polarization types used for the stream. The secret key is formed by the recorded qubit values where both happened to use the same polarization type. Thus, for these qubits both have recorded the same value but none but they know what that value actually is.

4. Remember, we are modeling a quantum computer. In a real system, the Qubits cannot be copied. The fact that qubits cannot be copied provides security in this scheme. An attacker in

the middle cannot read/measure the qubits and also transmit them again without changing the state of the qubits.

Here is an example QKE interaction.

Transmitter random polarizations (Linear, Circular)	LLC CLL CLCL
Transmitter random values	011 000 1110
Transmitter submitted qubits	↓↑↺ ↻↓↓ ↺↑↺↓
Receiver random polarizations	CCL CLC CLLC
Receiver measured values	110 001 1111
Polarization types exchanged Matching highlighted	LLC **CL**L **CL**CL CCL **CL**C **CL**LC
Both select the values only at the positions the polarization matched	011 **00**0 **11**10 110 **00**1 **11**11
Both find the same secret key!	0011

In this assignment, you need to do the following:
1. [15%] Implement and unit-test a qubit class
2. [15%] Implement and unit-test an XOR cipher/decipher class
3. [50%] Implement and unit-test an emulation of the QKE algorithm, followed by the secure exchange of a symmetrically encrypted message using the key produced by QKE. Experiment with three qubit stream lengths (16, 256 and 1024)
4. [20%] Create a report that documents your design and testing as well as the evaluation of your algorithms.

— End of Assignment 3 —