TP session 4 : Quantum Key Distribution simulation with Qiskit

Exercise 1: Design the key-encoding quantum circuit. For this, first create a 8-bit classical register and a 8-qubit quantum register and merge them. Hereinafter we will refer to this circuit as 'Alice quantum circuit'.

Exercise 2: Generate a random 8-bit quantum key. For this, first generate a random number using *'np.random.randint'* and then convert it into binary using *'np.binary_repr'*. Print the key both in decimal and binary forms. Finally, encode the key as qubits using the 'Alice quantum circuit'. For this, apply an X gate to the corresponding qubit when the digit in the key is equal to '1'. Plot the resulting circuit.

Exercise 3: Measure every qubit in the key according to the following basis [X,Z,Z,Z,X,Z,X,X] where X stands for the diagonal basis and Z for the computational basis. This corresponds to applying a H-gate to the qubit when measured in the diagonal basis. Print the resulting circuit. Next, propose a code generating a random measurement basis for every qubit.

Exercise 4: Define 'Bob's circuit' as a copy of Alice's circuit using the SendState funcion. Generate and apply random measurement basis to Bob's circuit as you did for Alice's. Measure all 8 qubits in Bob, run the simulation and plot the histogram. Is alice_key within the results of the histogram?

Exercise 5: Execute the proposed code. Once Alice and Bob's keys have been exchanged and qubits measured with different basis discarded, are the reduced keys equal (percentage of similarity = 1). Does this result depend on the number of bits in the key? How many bits needs your key to be able to send the key over long distances given the photon losses in optical fibers?

Exercise 6: Include a third circuit (Spy circuit), between Alice and Bob. The spy circuit measures using its own random basis. How can you detect that the key has been intercepted?