E91 Quantum Key Quantum key distribution

Jiarui Liu

1. Introduction.

Quantum cryptography is to use quantum uncertainty to construct a secure communication channel, so that any eavesdropping behavior on the channel cannot affect the communication itself, so as to achieve the purpose of eavesdropping failure to ensure the security of the channel.

The Ekert 91 protocol is a QKD protocol based on quantum entanglement proposed by Professor Ekert of Oxford University in 1991. In Ekert91 protocol, the distribution and measurement of entangled states are used to achieve key sharing, and the information obtained by Eve is estimated by Bell's inequality test.

2. The E91 protocol

Here is a brief introduction of E91. In E91 QDK Protocol, we used two groups of measurement basis to measure Qubit:

1. $Z-$basis: $B\text{ase}Z = \{|0\rangle, |1\rangle\}$;
2. $X-$basis: $B\text{ase}X = \{|+\rangle, |-\rangle\}$;

Each time Alice or Bob choose one of the two basis to measure.
The E91 protocol can be described in the following steps:
1. The source center prepare the quantum entangled pair as following:
$|\phi^+\rangle = 1/\sqrt{2}(|00\rangle + |11\rangle)$ The first qubit was sent to Alice and the second one sent to Bob.

2. Alice and Bob randomly choose a basis to measure their Qubit. They don't know the other one's choice until they finish measurement.
After measurement, they could discuss what basis they chose in classical channel.

3. Alice and Bob know the choice of both parties. They divide the measurement result into two groups: one is the decoy qubits Gd where they choose different measurement basis and another is the raw key qubits Gk where they choose the same measurement basis.

4. The group Gd is used to detect whether there is a eavesdropping. Gd is always entangled without eavesdropper in an ideal environment. If there is bit error in Gd, which means that there is also a eavesdropper, Alice and Bob think that the quantum channel is not safe and they will interrupt this communication and restart a new one.

5. If the quantum channel is safe, Gk can be used as the raw keys because Alice and Bob can receive the same measurements. Both Alice and Bob agree on that the measurement $|0\rangle|0\rangle$ represents the classical bit 0, while the measurement $|1\rangle|1\rangle$ represents the classical bit 1.

6. The E91 protocol ends successfully.

3. Discussion

As we can see from the prepared entangled pair above. The result of Alice's measurement and Bob's measurement should be exactly same if they use same measurement basis and no Eavesdropper exists. And they will use the measurement result to generate a key.

However, in the situation of existence of Eavesdropper. The Eavesdropper will steal the qubit which should be sent to Bob and measure it in a basis. After that he pass a new qubit to Bob. After discussion in classical channel, If Bob and Alice realized thy choose different basis to measure the pair, Bob would remeasure the qubit (which he actually received from Eavesdropper) in Alice's basis. They realize the existence of Eavesdropper if there is a chance that the measurement result of Alice and Bob differ. That is the way the Eavesdropper is "caught".

4. Implement in Q#

```
// Task 2.1. parpare the Entangled pair (Bell state);
operation CreateEntangledPair (qs : Qubit[]) : Unit is Adj {

    H(qs[0]);
    CNOT(qs[0],qs[1]);
}
```

```
// measure the qubit in the random base
operation MeasureQubit (qs : Qubit, bases : Bool) : Bool {
    if (bases == true) {
        H(qs);
    }
    return M(qs) == One;
}
```

```
operation E91_protocol() : Unit{

    let n = 5;
    let AliceArray = BuildRandomArray(n);
    let BobArray = BuildRandomArray(n);
    Message($"AliceArray  == {AliceArray}");
    Message($"BobArray  == {BobArray}");
    // let AliceArray =[true,false,true,false,true];
    // let BobArray = [true,false,true,false,true];
    mutable key = new Bool[0];
    for(i in 0..n-1){
        using(qs = Qubit[2]){
            CreateEntangledPair(qs);
            if(AliceArray[i] == BobArray[i]){
                let resultA = MeasureQubit(qs[0],AliceArray[i]);
                let resultB = MeasureQubit(qs[1],AliceArray[i]);
                Message($"{i}  A == {resultA}  B = {resultB}");
                set key+= [resultA];
            }
            ResetAll(qs);
        }
    }
    Message($"raw key == {key}");
}
```