Michał Kałmucki 151944

Laboratory 3 – BB84 Quantum Key Distribution Simulation

Introduction

The goal of this laboratory was to simulate the BB84 quantum key distribution protocol using a quantum circuit model. The protocol allows two parties (Alice and Bob) to establish a shared secret key using quantum states and random measurement bases.

BB84 Protocol Overview

In the BB84 protocol:

- Alice randomly selects a bit value `xA` and a basis `yA`
- Bob randomly selects a measurement basis `yB`
- Bits are kept only when `yA = yB` (key sifting)

The protocol was implemented and simulated using Qiskit and a quantum circuit model.

Quantum Circuit Implementation

Imports and Circuit Initialization

```
from qiskit import QuantumRegister, ClassicalRegister, QuantumCircuit
from qiskit_aer import Aer
from qiskit.compiler import transpile
import matplotlib.pyplot as plt

q = QuantumRegister(4)
c = ClassicalRegister(4)
qc = QuantumCircuit(q, c)
qc.reset(q)
```

Random Bit and Basis Generation (Alice)

```
qc.h(q[1])
qc.measure(q[1], c[1])

qc.h(q[2])
qc.measure(q[2], c[2])

qc.barrier()
```

Information Encoding (Alice)

```
qc.cx(q[1], q[0])
qc.ch(q[2], q[0])

qc.barrier()
```

Random Basis Selection and Decoding (Bob)

```
qc.h(q[3])
qc.measure(q[3], c[3])

qc.barrier()

qc.ch(q[3], q[0])
qc.measure(q[0], c[0])
```

Simulation and Data Collection

BB84 Simulation Function

```
backend = Aer.get_backend('qasm_simulator')

def run_bb84(shots):
    bits = []
    for _ in range(shots):
        compiled = transpile(qc, backend)
        job = backend.run(compiled, shots=1)
        result = job.result().get_counts()
        key = list(result.keys())[0]
        yB, yA, xA, xB = map(int, key)
        bits.append([xA, yA, yB, xB])
    return bits
```

Key Sifting

```
def sift_key(bits):
    kA = []
    kB = []
    for xA, yA, yB, xB in bits:
        if yA == yB:
            kA.append(xA)
            kB.append(xB)
    return kA, kB
```

Results

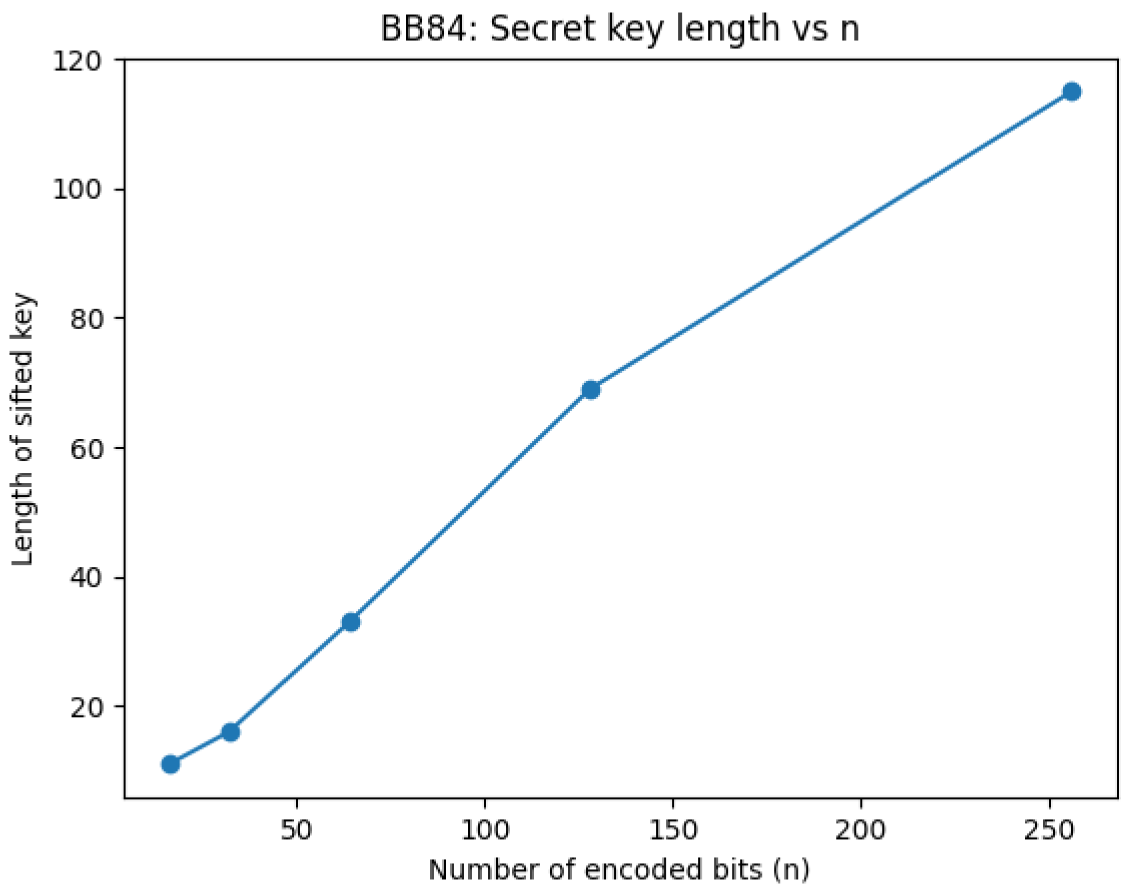
```
samples = [16, 32, 64, 128, 256]
key_len = []

for n in samples:
    bits = run_bb84(n)
    kA, kB = sift_key(bits)
    key_len.append(len(kA))
```

Output

n	key length
16	11
32	16
64	33
128	69
256	115

Graph



Discussion

The results show that the length of the sifted key increases approximately linearly with the number of encoded bits. On average, about 50% of the transmitted bits are discarded due to basis mismatch, which is consistent with the theoretical expectations of the BB84 protocol.

Conclusion

The BB84 protocol was successfully implemented and simulated using a quantum circuit model. The obtained results confirm the correctness of the protocol and demonstrate the key sifting process essential for secure quantum key distribution.