BB84

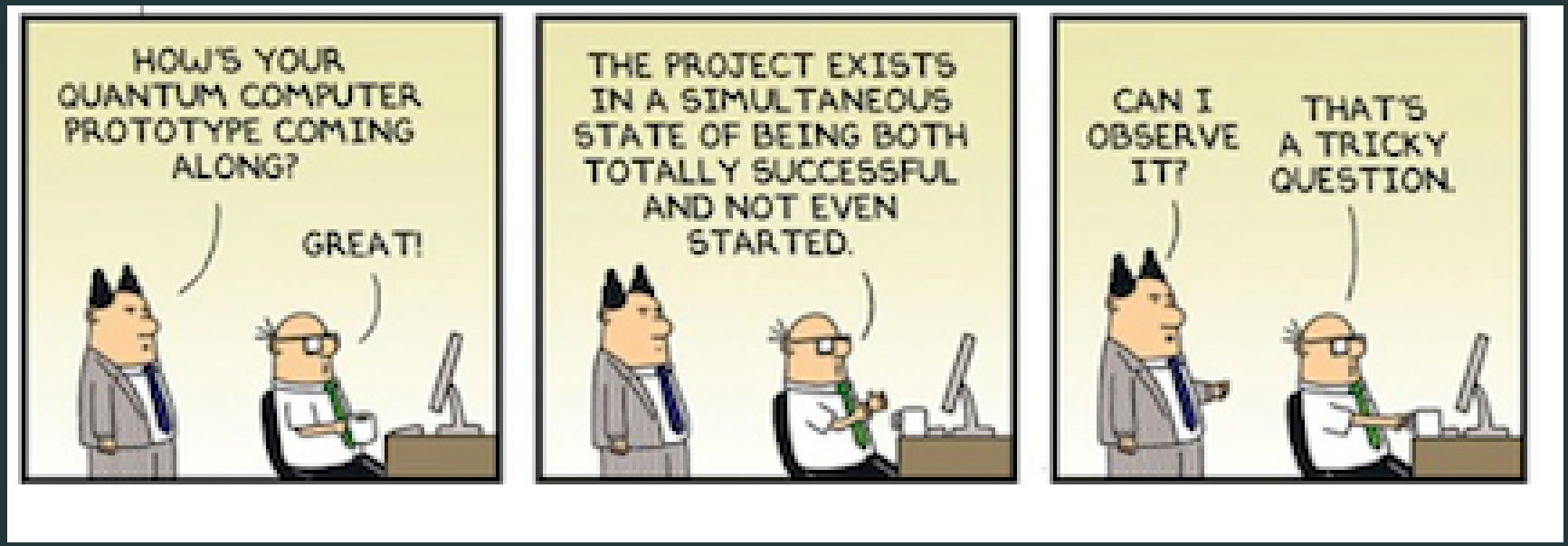
A Quantum Key Distribution Protocol

Lakshika, Shreya

July 2020

Overview

Simulation of BB84 Protocol


HOW'S YOUR QUANTUM COMPUTER PROTOTYPE COMING ALONG?
GREAT!
THE PROJECT EXISTS IN A SIMULTANEOUS STATE OF BEING BOTH TOTALLY SUCCESSFUL AND NOT EVEN STARTED.
CAN I OBSERVE IT?
THAT'S A TRICKY QUESTION.

Technology Stack

- Qiskit - An SDK for Quantum Computing
- IBM Q - IBM Quantum Experience to access IBM's quantum computers via the cloud
- Python Libraries

Quantum Key Distribution

- Key distribution is important in Cryptographic protocols

Quantum Key Distribution

- Key distribution is important in Cryptographic protocols
- Current protocols assume computational limitations

Quantum Key Distribution

- Key distribution is important in Cryptographic protocols
- Current protocols assume computational limitations
- What can be used...

Quantum Key Distribution

- Key distribution is important in Cryptographic protocols
- Current protocols assume computational limitations
- What can be used...
  - Quantum Cryptography

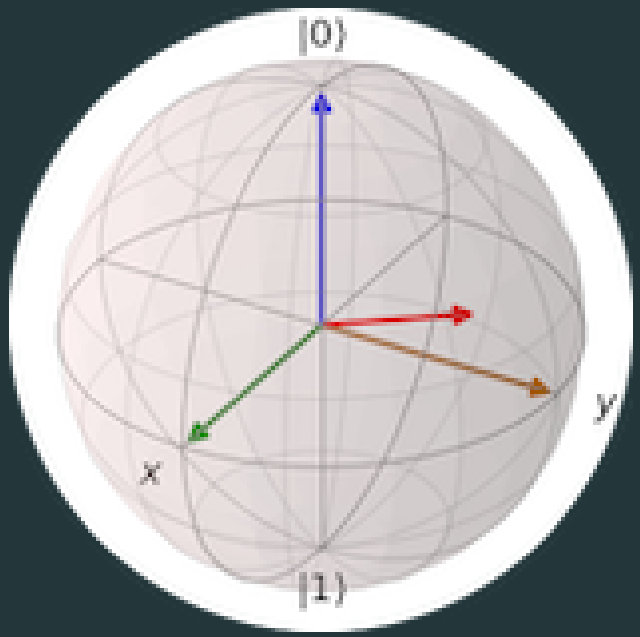
Quantum Key Distribution

- Key distribution is important in Cryptographic protocols
- Current protocols assume computational limitations
- What can be used...
  - Quantum Cryptography
  - Post-Quantum Cryptography

Quantum Key Distribution

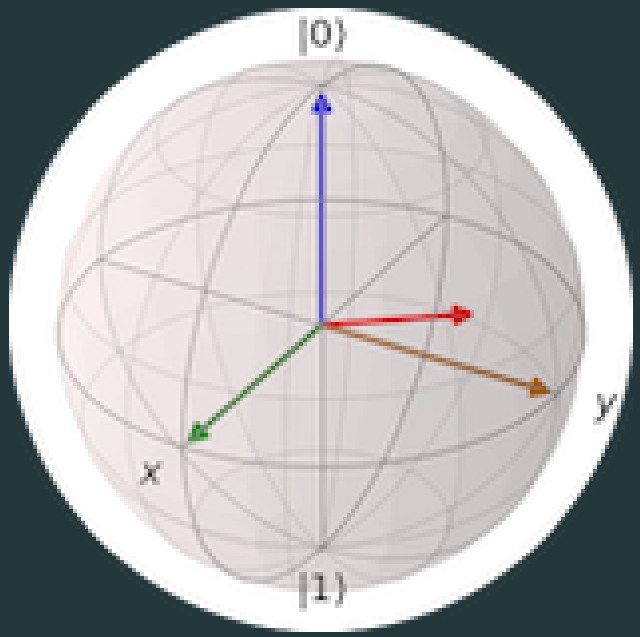
- Key distribution is important in Cryptographic protocols
- Current protocols assume computational limitations
- What can be used...
  - Quantum Cryptography
  - Post-Quantum Cryptography
- The first QKD protocol by Bennett and Brassard in 1984

The Quantum World



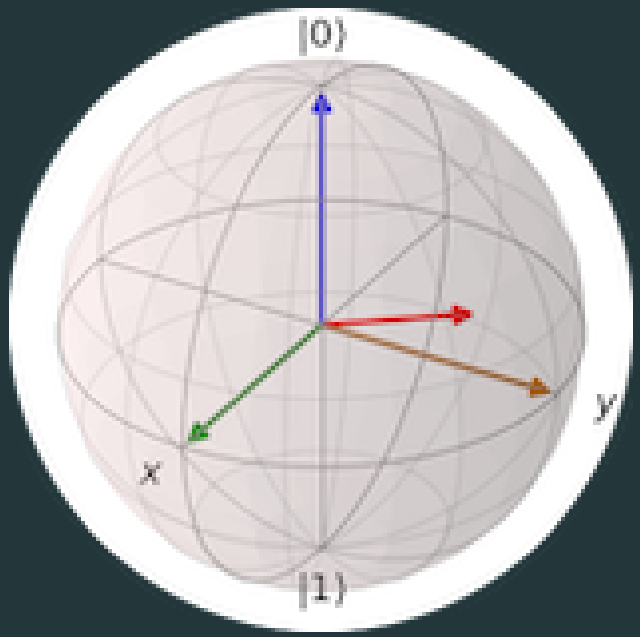
- **Qubit** - Basic unit of Quantum Information

The Quantum World



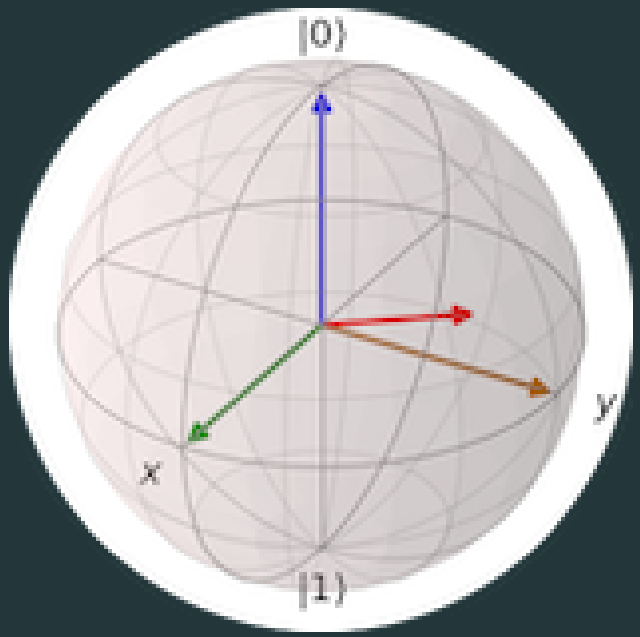
- **Qubit** - Basic unit of Quantum Information
- Bases and Measurement

The Quantum World



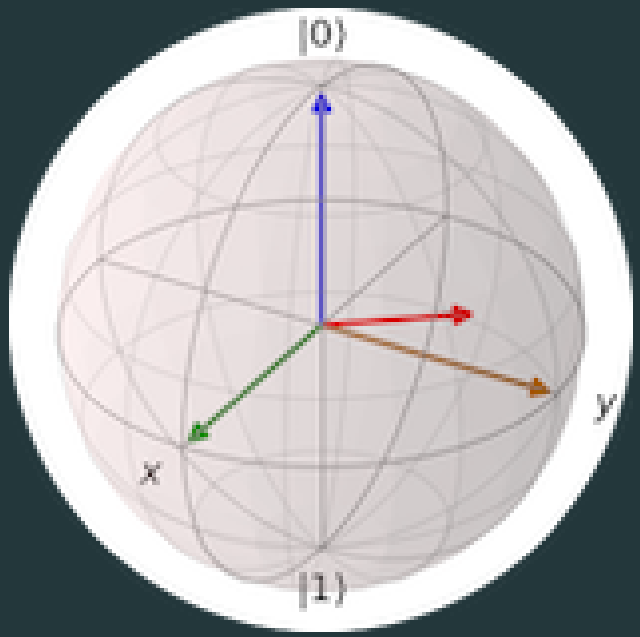
- **Qubit** - Basic unit of Quantum Information
- Bases and Measurement
- Communication channels

The Quantum World



- **Qubit** - Basic unit of Quantum Information
- Bases and Measurement
- Communication channels
  - A one-way physical Quantum channel

The Quantum World



- **Qubit** - Basic unit of Quantum Information
- Bases and Measurement
- Communication channels
  - A one-way physical Quantum channel
  - An authenticated two-way classical ideal channel

Protocol - Quantum Phase

- Generation of a random bitstring and bases

Protocol - Quantum Phase

- Generation of a random bitstring and bases
- Quantum transmission of encoded bits

Protocol - Quantum Phase

- Generation of a random bitstring and bases
- Quantum transmission of encoded bits
- Measurement by the receiver

Protocol - Quantum Phase

- Generation of a random bitstring and bases
- Quantum transmission of encoded bits
- Measurement by the receiver
- Public comparison of bases and sifting

Protocol - Information Reconciliation

The process of cleaning the keys from errors

- Error estimation

Protocol - Information Reconciliation

The process of cleaning the keys from errors

- Error estimation
- Low Density Parity Check codes based reconciliation

Protocol - Privacy Amplification

- Distillation of a highly secret key from a partially secure string by public discussion

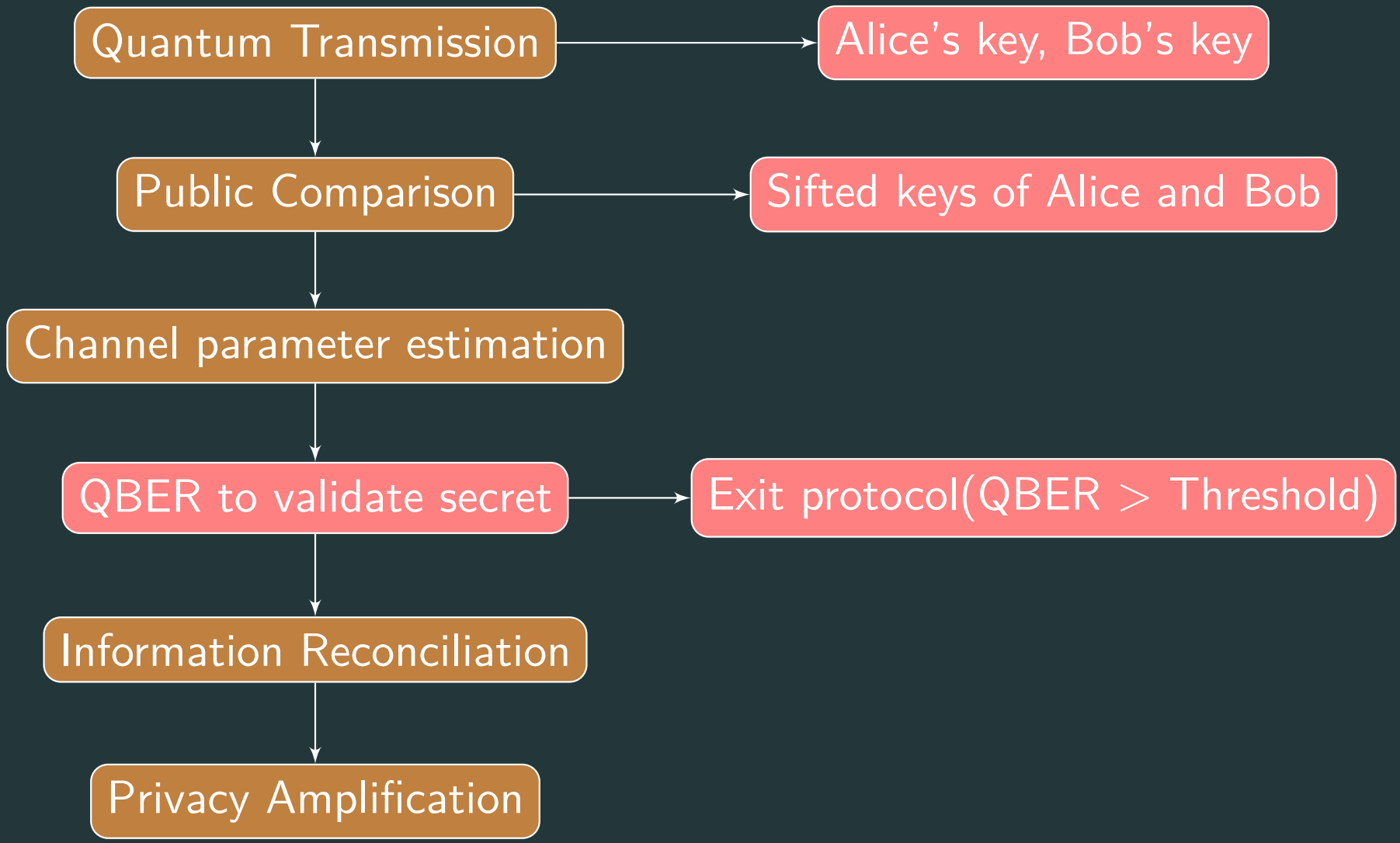
Protocol - Privacy Amplification

- Distillation of a highly secret key from a partially secure string by public discussion
- Shrinks the possible exposed information over the communication channels to almost zero

Protocol - Privacy Amplification

- Distillation of a highly secret key from a partially secure string by public discussion
- Shrinks the possible exposed information over the communication channels to almost zero
- 2-Universal hash functions are used

The Protocol at a glance !



Stats

- Implementation in three different cases
- Primary key lengths tested for : 50, 648, 1000
- Maximum error rate corrected : 1 %

```python
#Step 1 to Step 6
alice_bits, alice_bases, bob_bases, received_encode, received = gen
if not received :
    print('Abort : Eve is imitating Bob')

else:
    #Step 7 to Step 9
    bob_bits, agreed_base_indices = sifting(alice_bases, bob_bases,


    #Step 10 to Step 13
    error, S2T = qber(alice_bits, bob_bits, agreed_base_indices)
    print("QBER : ", error)

    #Step 14 - Alice and Bob check over a threshold for error befor
    if error > 0.03 :
        print ("Abort : Eavesdropping detected")

    else :
        #Step 15 - Alice and Bob generate their pseudo keys
        SminusT = select_bits(agreed_base_indices, S2T, 1)
        alice_pseudokey, bob_pseudokey = [], []
        for i in SminusT :
            alice_pseudokey.append(alice_bits[i])
            bob_pseudokey.append(bob_bits[i])

        if error != 0 :
            #Step 16 - Information Reconciliation
            if 648 <= len(alice_pseudokey) <= 700 :
                alice_pseudokey = alice_pseudokey[:648]
                bob_pseudokey = bob_pseudokey[:648]
            bob_corrected_key, success = reconciliation(alice_pseud

            if not success :
                print("Abort : Reconcilation not succeeded")

            else :
                #Step 17 - alice and bob perform privacy amplificat
                alice_key, bob_key, error = privacy_amplification(a
                print("Alice's key : ", alice_key)
                print("Bob's key : ", bob_key)
                print("Final error rate in Bob's key : ", error)
```

```python
            #Step 17 - alice and bob perform privacy amplification
            alice_key, bob_key, error = privacy_amplification(alice_pseudokey, bob_pseu
            print("Alice's key : ", alice_key)
            print("Bob's key : ", bob_key)
            print("Final error rate in Bob's key : ", error)
```

```
QBER :  0.005628517823639775
Alice's key :  [0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1,
0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0, 1, 1,
, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0
0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
, 0, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0
0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 1, 0,
, 0, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1
1, 0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1,
, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0
0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 0, 1,
, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1
0, 0, 1, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1,
, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0
1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1]
Bob's key :  [0, 1, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0
0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 0,
, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0
1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 0,
, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0
1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 0, 0,
, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1
0, 1, 1, 0, 1, 1, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0,
, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1
1, 0, 1, 0, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0,
, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0
0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0,
, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1
1, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1]
Final error rate in Bob's key :  0.0
```

Information	Alice	Eve	Bob
T	Y	Y	Y
Alice's test bits	Y	Y	Y
Bob's test bits	Y	Y	Y

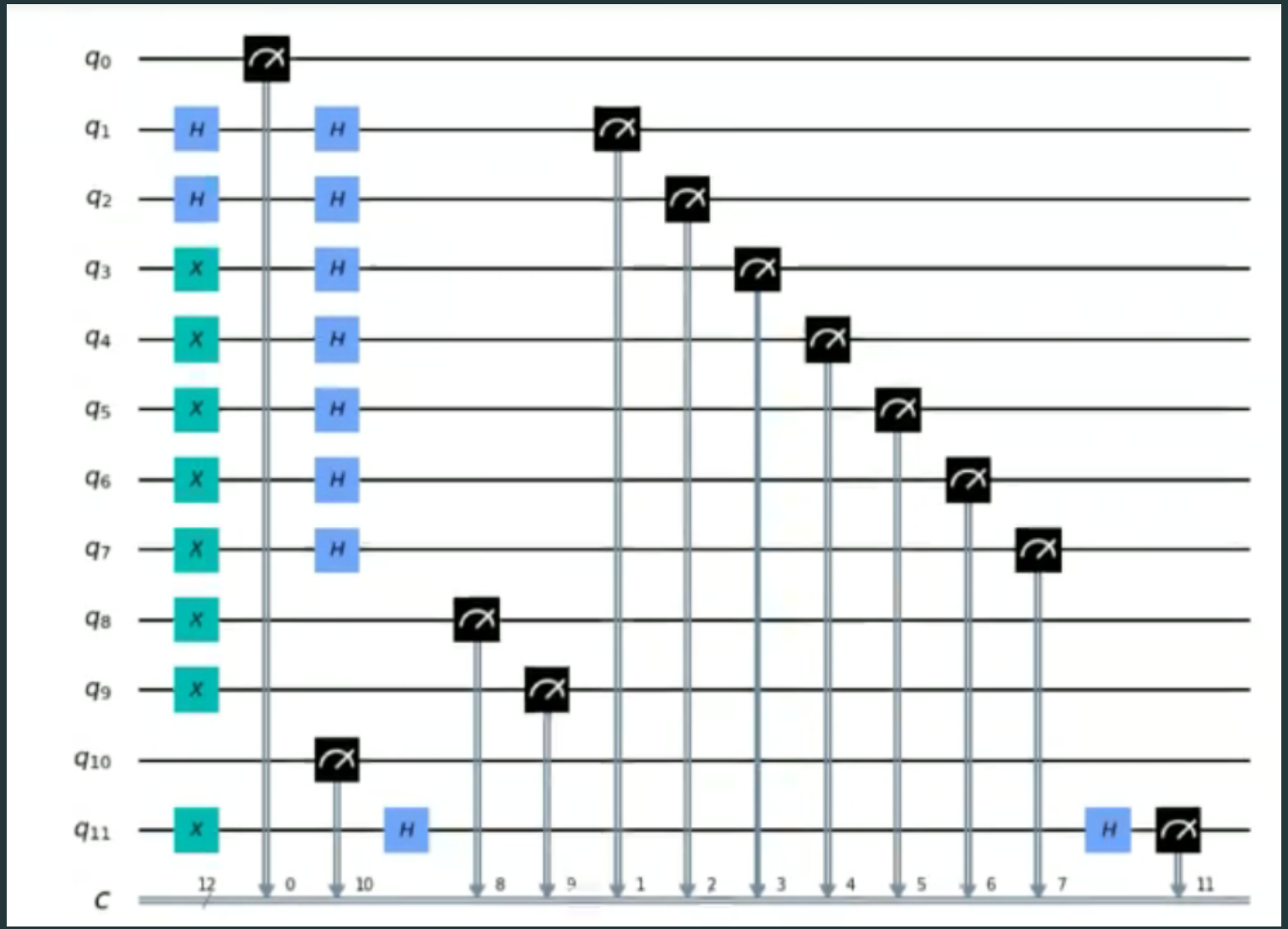
```python
def reconciliation(alice_bits, bob_bits, qber) :
    p = qber
    n = len(alice_bits)
    #Step1 : Produce parity check matrix(M1) of dimension m*n, n is length of bitstring
    H, m = parity_matrix(n, p)

    #Step2 : Alice produces the syndrome and hash values
    C = syndrome(H, alice_bits)

    #Step3 : Alice sends syndrome via CAC

    #Step4 : Bob produces the syndrome
    D = syndrome(H, bob_bits)

    #Step5 : Alice performs belief propagation algorithm
    y, success, i = belief_prop(C, D, bob_bits, MAX_ITERS, p, H)

    #Step6 : Bob sends success of reconciliation

    return y, success
```

```python
def privacy_amplification(alice_pseudokey, bob_pseudokey) :
    n, k = len(alice_pseudokey), len(alice_pseudokey) // 2
    seed = bit_string(n + k - 1)
    alice_key = toeplitz(n, k, alice_pseudokey, seed)
    bob_key = toeplitz(n, k, bob_pseudokey, seed)
    return alice_key, bob_key, error_rate(alice_key, bob_key)
```

```python
        else :
            #Step 17 - alice and bob perform privacy a
            alice_key, bob_key, error = privacy_amplif
            print("Alice's key : ", alice_key)
            print("Bob's key : ", bob_key)
            print("Final error rate in Bob's key : ",
```

```
Induces errors :  57
QBER :  0.008391608391608392
Alice's key :  [0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0,
1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0,
, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1, 1, 0
1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0
0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0,
, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0
1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0,
, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1
Bob's key :  [0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0
1, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 1, 0, 0, 0,
, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0
0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1,
, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 0, 0,
, 0, 1, 0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 0
1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1, 0,
, 1, 1, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 1
Final error rate in Bob's key :  0.0
```

Information
Pseudo key


q_0
q_1
q_2
q_3
q_4
q_5
q_6
q_7
q_8
q_9
q_{10}
q_{11}
c
H
X
12
0
10
8
9
1
2
3
4
5
6
7
11

References

The Protocol	Lecture Notes by *Univerity of Berkeley*
LDPC codes	*Alan Mink and Anastase Nakassis*, "Practical Strategies for QKD Key Production"
Standard Matrices	*P.Venkateshwari and M.Anbuselvi*, "Decoding performance of binary and non-binary LDPC codes"
	Mario Milicevic et al., "Key Reconciliation with LDPC Codes for Long-Distance Quantum Cryptography"
Privacy Amplification	*H. F. Chau et al.*, "Practical issues in quantum key-distribution post-processing"
Eavesdropping	*John Smolin et al.*, "Experimental Quantum Cryptography"

Difficulties faced

- Getting started with Quantum Computing
- In choosing a sub-domain
- Unsure about the language and tools to use
- Figuring out Information Reconciliation part

Learnings

- Explored the domain of Quantum Computing
- Attended the Qiskit Global Summer School
- Learnt the basics of Quantum Cryptography
- Reading research papers and trying to understand them
- Got introduced to Coding theory

Nature isn't classical, dammit, and if you want to make a simulation of nature, you'd better make it quantum mechanical...

- Richard Feynman