Analysis of Quantum Key Distribution

Divas Subedi
Department of Physics, Trinity College

(Dated: 16 May 2022)

With the rise of data science, particle physics, intensive drug research, and other problems with higher complexity, the computational power of current computer technology is becoming a bottleneck. This paper will shed light on a new paradigm of computing — Quantum Computing. The report will focus on the current state of quantum encryption and the threat to current encryption due to quantum computers. Finally, an analysis of Quantum Key Distribution Algorithms, specifically on BB84, is conducted.

I. INTRODUCTION

Richard Feynman, in his famous paper "Simulating Physics with Computer," demonstrated that to simulate physics at the quantum level, a classical computer needs exponentially growing storage and computation power to keep track of every possible evolution and its probability. In order to conduct a accurate simulation a quantum computer is required. In 1980, Paul Benioff described the first quantum mechanical model of quantum computing. He showed that computers could operate under the laws of quantum mechanics by describing the Turing machine in Schrödinger's equation form[1]. Since then, numerous effort have been put forward for realization of Quantum Computers.

With the rise of quantum computing technology, it is of utmost importance to consider the consequences of its realization. One of the most concerning topics with the scaling of quantum computers is the current state of cryptography. Current computer security systems are built upon one-way or trap-door functions. Such functions are computationally hard to solve, however, computationally easy to verify. One of the most used trap door functions in cryptography is the integer factorization problem. While it is hard to prime factorize a composite integer, it is easy to verify the factors once it is solved. Most encryption algorithms, including state of art RSA (Rivest-Shamir-Aldeman) encryption, rely on the computational difficulty of factorizing integers. Peter Shor demonstrated that this problem can be solved with relative ease with a sufficiently large quantum computer.[2]. This urges the security community to look to alternative encryption algorithms. While there is an appreciable ongoing effort to push post-quantum encryption algorithms using classical computers, this paper will be focusing on the encryption method that comes along with quantum computation and quantum communication.

II. QUANTUM COMPUTERS

A. Quantum Superposition

The properties of a quantum object are quantized, ie. there is a discrete set of values that a physical property or observable of a particle can be measured. For example, the spin of an electron can be measured to be either $\frac{1}{2}\hbar$ or $-\frac{1}{2}\hbar$, the energy level of an electron in a hydrogen atom can be measured to be one of the principal quantum levels. However, the quantum state of those properties can also be a combination of those discrete values. Two or more quantum states can be added together or "superposed" to form another valid quantum state. Every quantum state can be represented as a weighted sum of two or more distinct states.[3]. When an observable is measured, the superposition collapses and the outcome is one of the discrete values. Observation is not possible without interaction. A classical analogy could be a spinning coin in the dark. The only way to observe it is to smack a palm down on it, forcing it to be either heads or tails without knowing what the orientation was before. So, measuring particles forces the particle to take one of the quantum states.

B. Qubit

Any information can be represented as a string of bits as long as the interpretation of the bit string is known. In classical computers, the bits can be encoded in physical forms such as charge stored in a capacitor, flip-flop, or the magnetization of a tape. Similarly, quantum objects can be used to store information in form of bits. For instance, the polarization of a photon can be used to encode a bit — vertical polarization can be interpreted as 1 and horizontal as 0. However, the polarization of a photon can also be in a superposition of being vertically and horizontally polarized.

These states 0 and 1 can be represented as a set of orthonormal vectors ($|0\rangle$ and $|1\rangle$). Those orthonormal vectors form the basis of the vector space where all possible states can be defined. While in superposition, its state can be described as a linear combination of those basis vectors. Again, let $|0\rangle$ vector represent vertical polarization and $|1\rangle$ vector represent horizontal. Now, consider a superposition state such that probability of measuring it as $|0\rangle$ is $|\alpha|^2$ and measuring it as $|1\rangle$ is $|\beta|^2$. This superposition state can be described by the vector:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

Here, $\alpha, \beta \in \mathbb{C}$ and $|\psi\rangle \epsilon\ \mathbb{C}^2$ As per Born's rule, the coefficients associated with the states are ℓ^2 normalized, ie. $\alpha^2 + \beta^2 = 1$.

C. Bloch Sphere

To visualize the entirety of a qubit, a four-dimensional figure is required. However, the normalization constraint that

$\alpha^2 + \beta^2 = 1$ makes it feasible to represent the vector in a 3 -dimensional sphere. Each complex number can be written in their Euler form $\alpha = r_\alpha e^{i\phi_\alpha}$ and $\beta = r_\beta e^{i\phi_\beta}$.[4]

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$
$$|\psi\rangle = r_\alpha e^{i\phi_\alpha}|0\rangle + r_\beta e^{i\phi_\beta}|1\rangle$$
$$|\psi\rangle = e^{i\phi_\alpha}(r_\alpha|0\rangle + r_\beta e^{i(\phi_\beta - \phi_\alpha)}|1\rangle)$$

The global phase $e^{i\phi_\alpha}$ is irrelevant

$$|\psi\rangle = r_\alpha|0\rangle + r_\beta e^{i(\phi_\beta - \phi_\alpha)}|1\rangle$$

Now since $r_\alpha^2 + r_\beta^2 = 1$, they can be written as

$$r_\alpha = \cos\left(\frac{\theta}{2}\right) \text{ and } r_\beta = \sin\left(\frac{\theta}{2}\right)$$

Taking the relative phase: $\phi = \phi_\beta - \phi_\alpha$

$$|\psi\rangle = \cos\left(\frac{\theta}{2}\right)|0\rangle + \sin\left(\frac{\theta}{2}\right)e^{i\phi}|1\rangle$$

Now, this can be represented in a spherical co-ordinate system. This unit sphere is called Bloch Sphere. Figure 1 shows a Bloch sphere. In the figure, it seems like $|1\rangle$ is antipodal to $|0\rangle$. However, they are orthogonal in this space. The Z-axis in the Bloch sphere is regarded as the computational basis.

Axis	$\|\psi\rangle$
+X	$\frac{1}{\sqrt{2}}\|0\rangle + \frac{1}{\sqrt{2}}\|1\rangle$
−X	$\frac{1}{\sqrt{2}}\|0\rangle - \frac{1}{\sqrt{2}}\|1\rangle$
+Y	$\frac{1}{\sqrt{2}}\|0\rangle + \frac{i}{\sqrt{2}}\|1\rangle$
−Y	$\frac{1}{\sqrt{2}}\|0\rangle - \frac{i}{\sqrt{2}}\|1\rangle$

TABLE I. Axes and superposition

D. Quantum gates

Much like classical computer needs logical gates, quantum gates are need for quantum operations on a qubit. The operation of quantum gates are described by linear algebra rather than Boolean algebra.The vectors $|0\rangle$ and $|1\rangle$ can be written as $\begin{bmatrix}1\\0\end{bmatrix}$ and $\begin{bmatrix}0\\1\end{bmatrix}$ respectively as set of orthonormal vectors in $\mathbb{R}^2$. Quantum gates performs an operation on a qubit that produces a desired change the state of qubit. Quantum gates can be written as form of unitary matrices, and the interaction of the qubit with the gate can be given by matrix multiplication. For instance, if a quantun gate $\hat{U}$ is applied to a qubit $|\psi\rangle$, then the outcome is given by $\hat{U}|\psi\rangle$. The unitary nature of these matrix makes sure that the normalization criteria is satisfied for the the outcome. Some of the quantum gates are discussed below:

The **Pauli-X gate**, or simply X-gate, is similar to classical

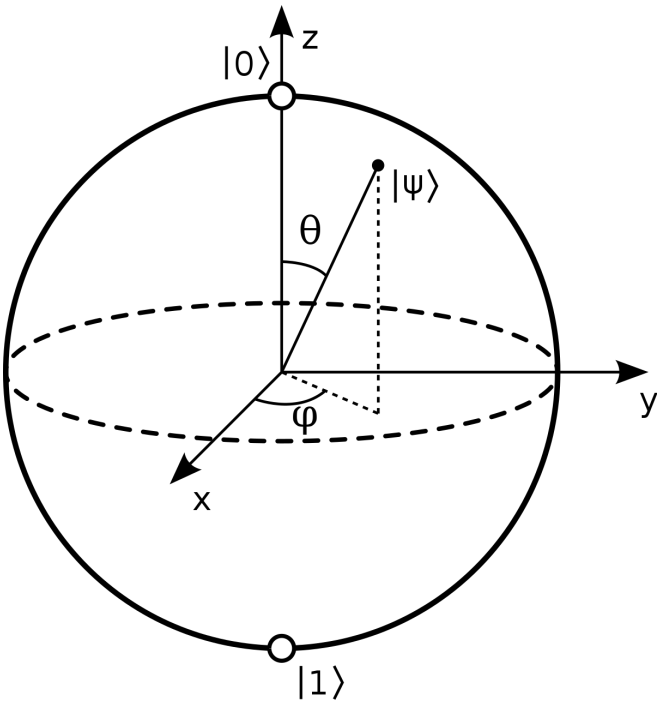

Fig. 1. A Bloch Sphere representing a qubit

NOT gate.

$$X = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}$$

When $|\psi\rangle = |0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$,

$$X|\psi\rangle = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}0\\1\end{bmatrix} = |1\rangle$$

.

If the quantum state is in superposition $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$.

$$X|\psi\rangle = \begin{bmatrix}0 & 1\\1 & 0\end{bmatrix}\begin{bmatrix}\alpha\\\beta\end{bmatrix} = \begin{bmatrix}\beta\\\alpha\end{bmatrix} = \beta|0\rangle + \alpha|1\rangle$$

So, X gate rotates the Bloch sphere by π radians about the X-axis.

Hadamard Gate(H) is another important gate which is represented by matrix:

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}$$

It rotates a qubit π radians about $\hat{x} + \hat{z}$ axis. When $|\psi\rangle = |0\rangle = \begin{bmatrix}1\\0\end{bmatrix}$,

$$H|\psi\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\1 & -1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}\frac{1}{\sqrt{2}}\\\frac{1}{\sqrt{2}}\end{bmatrix} = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$$

This vector points toward +X axis in the Bloch sphere, and it has a special name $|+\rangle$. Similarly, $H|1\rangle = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle = |-\rangle$. Both $|+\rangle$ and $|-\rangle$ have equal probability of being measured as either $|0\rangle$ or $|1\rangle$ when measured in computational basis.

E. Implementation of Quantum Computers

Even though quantum computing is still in its infancy, there are a lot of proposed theories backed by mathematics and physics on what a sophisticated computer can do like factorizing large prime numbers, creating truly randomized numbers, and simulating complex protein molecules. With the rise of data science and machine learning which requires a huge amount of computational power, classical computers might turn into a bottleneck. However, the implementation of quantum computers comes with a lot of difficulties. A quantum computer is very sensitive to perturbation and noise due to the environment. Quantum computers need to be in a very low temperature setting which requires expensive dilution refrigerators. The fault tolerance of quantum computing is slim. There are chances that a qubit carrier is lost, or information carried is lost.[5]. If the quantum gates are not faster than the decoherence time, information is lost. Hence, current quantum computers are not able to utilize a huge number of qubits to make a big impact in daily life.

1. IBM-Q programming

IBM-Q is the 16-bit quantum computer that is available for everyone to use for free. IBM-Q also supports Qiskit. Qiskit is an open-source framework for quantum computing. It provides tools for creating and manipulating quantum programs and running them on prototype quantum devices and simulators. This project will utilize the IBM-Q to perform some experiments regarding BB84 protocol.

III. ENCRYPTION

Encryption is a process of scrambling the data in such a manner that it is incomprehensible to anyone without the proper key and method of unscrambling it. In networks, messages and information are encrypted before sending so that any eavesdropper might not replicate, change or misuse the data. There are various ways of encrypting a piece of information. In computing and networking, a piece of information would generally mean any type of data in its binary form or its numerical equivalent. Encryption can broadly be classified into two categories: Symmetric and Asymmetric.

A. Symmetric Encryption

As the name suggests, the process of encrypting a piece of information and decrypting the cipher is symmetric. The same key is used for encrypting and decrypting. A simple example of the encryption of a message might be shifting every letter to the next letter. "HELLO" would be "JFFMP" and the intended receiver should have the key that is shifting the letter to the previous one. This makes symmetric encryption easy and fast to use. Symmetric encryption is extensively used in the current form of cybersecurity. Current techniques implemented for symmetric encryption are highly sophisticated and required an enormous amount of time and energy to break. Advanced Encryption Standard, the most widely used symmetric encryption, is deemed to be secure. The only way to get the key is to exhaustively try all possible keys. While there have been some known attacks on this mode of encryption, none of them have been efficiently able to crack the encryption.[6]

The two biggest drawbacks of symmetric encryption are key distribution and the number of keys required in a network. If a system involves three people (Ace, Becky, and Charlie), each pair needs to have their unique key. Ace and Becky share a key that is unknown to Charlie. Ace and Becky each have a shared key with Charlie unknown to the other. If Dave joins the system, now Dave needs to have a shared key with everyone. So, the number of keys keeps on increasing. For a system with n members or nodes, there needs to be $\binom{n}{2}$ keys. Again, since both parties need to know the encryption key, keys cant be shared through the network unless it is already made secure using another layer of encryption.

B. Asymmetric Encryption

As the name suggests, the process of encrypting a piece of information and decrypting the cipher is asymmetric. Different keys are used for encryption and decryption. This means every node in the network needs to have two keys: private and public. The private key is supposed to be kept secret by the individual and the public key is shared with everyone in the network. For a network of n nodes, the required amount of key is $2 \cdot n$. The specialty of asymmetric encryption is that they come in pairs and data encrypted by one key can be decrypted if and only if decrypted by its pair. A message encrypted by the public key can be decrypted by its private key only and vice-versa. Now, if Ace wants to send Becky a message, he will use Becky's public key to encrypt the message and sends it. He can be sure that no one except Becky can read the message as only Becky has the private key. Another example of its implementation: Evan wants to send a message to the entire group and wants to be sure that the message is unaltered, he can encrypt the message using his private key. Everyone with his public key would be able to decrypt it and can be certain that Evan sent it. An encrypted message can be further encrypted using a different key. For example, if Ace wanted to send a message to Becky and Becky wants to be entirely certain that it was Ace who sent the message, Ace can first encrypt the message using Becky's public key so no one except Becky can access it. After the first encryption, he can further encrypt the encrypted message using his private key. So, now Becky is entirely sure the message was from Ace and the communication is secure. This forms the basis of the digital signature that is used to

verify the authenticity of digital documents.

1. RSA Encryption

RSA (Rivest–Shamir–Adleman) is an example of an asymmetric encryption algorithm. This algorithm relies on the fact that factoring a number composed of two large primes, is computationally complex. Factoring integer falls under the category of NP (Non-Polynomial) problem. The time required to compute the problem grows exponentially with respect to the number of bits required to store the prime factors. Therefore, for a sufficiently large number, it would require an enormous amount of time to solve the problem. This can be used to create secure keys. Algorithm 1 shows the procedure to develop private and public keys.

Algorithm 1 RSA Algorithm

Problem: Generate public and private key

1: **choose** 2 random prime number p and q of approximately same size
2: $n \leftarrow p \times q$
3: $\phi(n) \leftarrow (p-1) \times (q-1)$ ▷ $\phi(n)$ is the Euler's totient function
4: **choose** e such that $\gcd(e, \phi(n)) = 1$
5: **find** d such that $d \times e \equiv 1 (\mathrm{mod} \phi(n))$

Now the (n,d) and (n,e) are private and public keys respectively.

Given that, p and q are a huge number, even if the public key and the entire algorithm are known to someone, it is almost impossible to figure out the private key. In real-life applications, p and q are huge numbers. They use 2048-bit numbers for p and q meaning each number is over at least $2^{1800} \approx 10^{541}$. The computation with numbers with hundreds of digits is very difficult and even the fastest classical computer might take millions of years to break one pair of asymmetric keys.

2. Shor's Algorithm

Shor's Algorithm was developed by Peter Shor in Bell's Lab. The algorithm is an approach to factorizing a large number into its prime factor. It consists of a certain number of stages – one of which requires a Quantum Computer. In algorithm 2, the solution to step 2 has modular periodicity. The algorithm uses quantum Fourier transform, which requires a sufficiently large computer, to determine this period.[7] [2].

Algorithm 2 Shor's Algorithm

Problem: find p and q such that $N = p \times q$

1: **pick** a such that $\gcd(a, N) = 1$
2: **find** r such that $a^r \equiv 1 (\mathrm{mod} N)$
3: **if** r is even **then**
4: $\quad x \leftarrow a^{\frac{r}{2}} \bmod N$
5: $\quad$ **if** $x + 1 \not\equiv 0 (\mathrm{mod} N)$ **then**
6: $\quad\quad \{p, q\} \leftarrow \{\gcd(x+1, N), \gcd(x-1, N)\}$
7: $\quad$ **end if**
8: **else**
9: $\quad$ **goto** step 1
10: **end if**

IV. QUANTUM KEY DISTRIBUTION

A. No Cloning Theorem

The no-cloning theorem states that it is impossible to create an independent and identical copy of an arbitrary unknown quantum state. Quantum measurements are destructive, in the sense that they alter the state of the system measured. When an observable of a system is measured, the superposition it was in collapses. Therefore, it is impossible to measure and copy a qubit via measuring. Alternatively, the evolution operator of the combined system (qubit that is to be copied and qubit that is used to copy the information), can be controlled such that after a finite time, the state of the qubit is replicated to another. However, no such evolution operator exists that can clone all the states. While the no-cloning theorem means that we cannot copy qubits in for the sake of error correction, this property can be advantageous in computer networking and data security.[8]

B. BB84

BB84 is a quantum key distribution protocol that uses a secure quantum channel to distribute shared keys. When qubits are measured consecutively in the same basis, they exhibit deterministic behavior. However when they are measured consecutively in different bases, that are not orthogonal to the previous one, they exhibit probabilistic behavior. BB84 uses this fact to generate secured keys. This protocol makes sure that there was no other eavesdropper listening in to the quantum channel during the key sharing process[9]. Polarization of photon can be used to implement qubit. In this case, polarizers kept in rectilinear and diagonal orientation are two sets of non-orthogonal bases. Fig. 2 shows the example of BB84 protocol in action. Let's say Ace and Becky want to share keys. Algorithm 3 describes the procedure to generate the keys.

Algorithm 3 BB84 Algorithm

Problem: Generate shared key through quantum channel

1: Ace prepares some random bits
2: Ace send the bit encoded in random bases
3: Becky measures the qubit in random bases
4: After all the qubit have been received, Ace and Becky publish their basis
5: Both crosses off the bit corresponding to mismatching bases

The remaining bits is the key

It is known that unless the qubit is interfered, the bit value for cases where they both share the same bases should be the same. However, there are cases where they might not be the same. One obvious case is the error introduced in the physical implementation of the quantum channel. Parties sharing keys should have knowledge of how good the quantum channel is, ie. the maximum error introduced by the quantum channel during transmission of bits. Another case is if the quantum

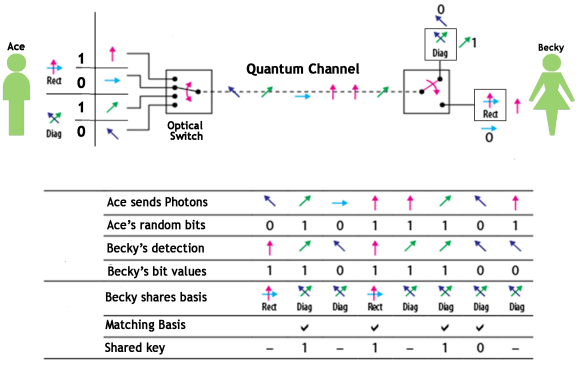

Fig. 2. Example of BB84 protocol in action

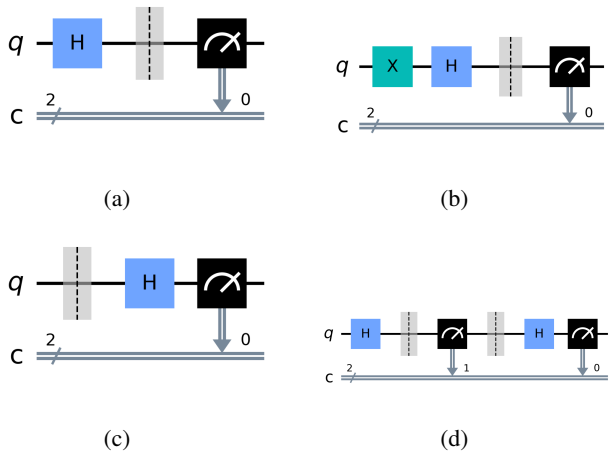

Fig. 3. Example circuits for BB84. (a) Ace sends 0 encoded in X basis and Becky measures in Z-basis. (b) Ace sends 1 encoded in X basis and Becky measures in Z-basis. (c) Ace sends 0 encoded in Z basis and Becky measures in X-basis. (d) This is case of interception. Alice send 0 encoded in X basis, Eve intercepts by measuring in Z-basis and finally Becky measures in X-basis

channel is compromised. If there was an eavesdropper, let's say Eve, listening to the quantum channel, the only way they could get information out of it by making a measurement, which inevitably changes the qubit. This fact allows Ace and Becky to determine if the channel was compromised. A selected sample of keys can be published over a public channel. The shared bits by Ace and Becky should share a certain portion of the bits, which is determined by the error rate introduced by the quantum channel.

C. Implementation and Analysis of BB84

For this paper, simulation of BB84 was done using Qiskit and IBM-Q. Due to the lack of a quantum channel, a quantum circuit was made to simulate a channel. Two possible scenarios were created: one without eavesdroppers and one with. Key was generated randomly and was encoded with random bases. The bases used are: Computational basis and Fourier basis, which correspond to Z-axis and X-axis respectively in Bloch Sphere. The interpretation of bit is given in Table II. The qubits are prepared as $|0\rangle$ by default. X gate is used to flip it to $|1\rangle$. Since Qiskit only works with the computational basis, encoding bit in Fourier basis was achieved by using a H(Hadamard) gate. The interception was implemented by making a measurement with chosen basis. Example circuit and their interpretation is given in Fig. 3

TABLE II. Bit interpretation of qubit in different bases

	Z-axis		X-axis	
Qubit State	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$
Bit interpretation	0	1	0	1

A raw key was generated of length 1000 by Ace and was encoded using random bases. Becky measures the qubit using random bases. For the intercepted case, Eve measures the qubits before Becky using random bases. After the transmission is complete, only the bits for which Ace and Becky share the same basis are kept. Out of those keys, 20% is sampled and used for validation. This experiment is repeated 1000 to create a Monte Carlo simulation. The histogram of the correlation between the sample keys is shown in Fig. 4. It can be seen that the correlation for the ideal case is always 1. This might not be the case in real life as imperfection in the quantum channel will sometime alter the qubit. However, there is a distinct difference between ideal and intercepted cases. In this case the mean of correlation for intercepted was 0.748 and the standard deviation was 0.0465. These values is strong suggestion of interception. If it is found that the correlation is lower than a certain threshold value, which is to be determined for each system, the channel is deemed to be compromised. Further transmissions are canceled until the problem with the quantum channel is dealt with. This allows us to securely generate keys and be sure that the information of the key was not compromised.

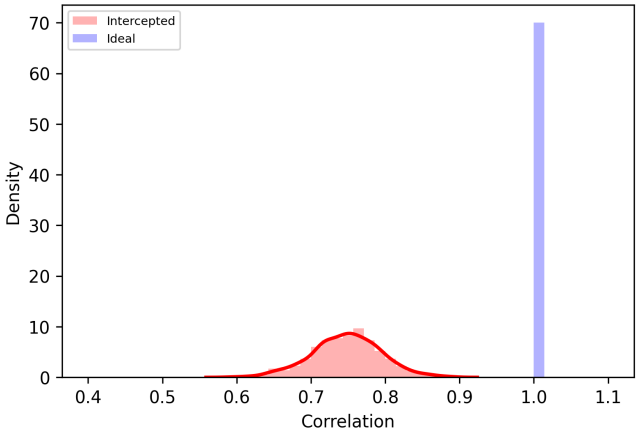

Fig. 4. Histogram showing the correlation between sampled keys for intercepted and ideal quantum channel.

V. CONCLUSION

Quantum Computer is currently a growing technology that aims to answer solve various different kinds of problems ranging from simulating complex proteins, integer factorization, and other BQP problems. The development of quantum computers might be a threat to the current security system as the current security system depends on the problems deemed to be hard to solve by classical computers. However, a quantum computer might be able to solve the problem with ease. One of the threats is to currently used asymmetric encryption algorithm: RSA, which is extensively used for sharing keys and user authentication. This relies on integer factorization which is an easy problem once a large enough quantum computer is built. Quantum Computing and communication, however, provide another way to securely share keys. One of the protocols BB84 is studied. Monte Carlo simulation was done to demonstrate how the BB84 protocol can securely generate keys as well as identify if the quantum channel used for sharing keys was secure.

[1] P. Benioff, en"The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines," Journal of Statistical Physics **22**, 563–591 (1980).

[2] P. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pp. 124–134.

[3] P. Dirac, *The Principle of Quantum Mechanics (3rd Edition)* (1947) p. 11.

[4] I. Glendenning, "The bloch sphere," QIA Meeting, TechGate, European Centre for Parallel Computing at Vienna.

[5] R. Van Meter, T. D. Ladd, A. G. Fowler, and Y. Yamamoto, "Distributed quantum computation architecture using semiconductor nanophotonics," **08**, 295–323, 0906.2686.

[6] G. Ou, en"Is encryption really crackable?" .

[7] E. Lucero, R. Barends, Y. Chen, J. Kelly, M. Mariantoni, A. Megrant, P. O'Malley, D. Sank, A. Vainsencher, J. Wenner, T. White, Y. Yin, A. N. Cleland, and J. M. Martinis, "Computing prime factors with a josephson phase qubit quantum processor," **8**, 719–723.

[8] W. K. Wootters and W. H. Zurek, en"A single quantum cannot be cloned," Nature **299**, 802–803 (1982).

[9] P. W. Shor and J. Preskill, "Simple proof of security of the BB84 quantum key distribution protocol," **85**, 441–444, quant-ph/0003004.