Attacking Quantum Key Distribution Protocols

Aritra Mukhopadhyay
National Institute of Science Education and Research
Bhubaneswar, Odisha 751005, India
4th year, Integrated M.Sc. Physics
Roll No.: 2011030
(Dated: May 9, 2024)

Classical cryptography is facing an unprecedented threat with the advent of quantum computing. Quantum algorithms can potentially break the security of many encryption algorithms currently in use, compromising the confidentiality and integrity of sensitive information. In this project, we demonstrate the implementation of two Quantum Key Distribution (QKD) protocols, BB84 and E91, which provide a secure solution for key distribution over an insecure channel. We simulate these quantum-based protocols over classical infrastructure, leveraging the internet to connect multiple computers, using Flask, a Python web framework. The quantum calculations are performed with the help of Qiskit, a popular open-source quantum development environment. By simulating the distribution of cryptographic keys over a real-world network, our project showcases the vulnerability of classical cryptography and highlights the importance of developing quantum-resistant cryptographic protocols that can ensure secure communication in a post-quantum world.

CONTENTS

I. INTRODUCTION

Cryptography, the practice of secure communication, has been a vital component of modern society. With the rapid growth of digital communication, the need for secure data transmission has become increasingly important. Classical cryptography, which relies on complex mathematical algorithms to ensure security, has been the cornerstone of secure communication for decades. However, with the advent of quantum computing, the security of classical cryptography is facing an unprecedented threat.

Classical cryptography can be broadly classified into two categories: symmetric and asymmetric cryptography. Symmetric cryptography employs the same secret key for both encryption and decryption, whereas asymmetric cryptography uses a pair of keys - a public key for encryption and a private key for decryption. While symmetric cryptography offers faster computation and lower computational overhead, it suffers from the problem of key distribution. In other words, securely distributing the shared secret key between parties has been a longstanding challenge.

Asymmetric cryptography, on the other hand, seemed to offer a solution to this problem. By using public-key cryptography, parties can exchange encrypted messages without having to share a secret key in advance. The security of asymmetric cryptography relies on complex mathematical problems, such as factorization and discrete logarithms, which are computationally infeasible for classical computers to solve. However, with the advent of quantum computing, these mathematical problems can be efficiently solved using quantum algorithms like Shor's algorithm.

The potential consequences of this development are far-reaching. If a large-scale quantum computer were to be built, it could potentially break the security of many encryption algorithms currently in use, compromising the confidentiality and integrity of sensitive information. This has sparked a growing interest in developing cryptographic protocols that can resist attacks from quantum computers.

However, in the face of this quantum threat, quantum computation itself comes to the rescue. The solution lies in revisiting the primitive symmetric cryptography method, which, although slower than asymmetric cryptography, offers a more secure approach. The main challenge in symmetric cryptography - securely distributing the shared secret key between parties - can be overcome using quantum algorithms.

Quantum Key Distribution (QKD) protocols, such as BB84 and E91, provide a secure way to distribute symmetric keys between parties. Although these algorithms do not prevent eavesdropping, they have a special mechanism by which both parties would know if eavesdropping has happened or not. If an eavesdropper, like Eve, attempts to measure the quantum keys during transmission, it will introduce errors, making it detectable by the legitimate parties, Bob and Alice.

The beauty of QKD protocols lies in their ability to detect any attempt by an eavesdropper to access the quantum keys. This is achieved through the no-cloning theorem, which states that an arbitrary quantum state cannot be copied exactly. Any attempt to measure or copy the quantum keys would disturb their state, introducing errors that can be detected by Bob and Alice during the classical post-processing steps.

If an eavesdropping attempt is detected, Bob and Alice can simply reject the generated keys and start over with a new key exchange. This ensures that the shared secret key remains secure, even in the presence of a powerful attacker with access to quantum computers.

In this report, we will explore the implementation of QKD protocols, specifically BB84 and E91, over a network using Flask and Qiskit. We will demonstrate how these protocols can provide a secure solution for key distribution, even in the face of quantum threats. The project aims to showcase the vulnerability of classical cryptography and highlight the importance of developing quantum-resistant cryptographic protocols.

II. CRYPTOGRAPHY

Cryptography, the practice of secure communication, has become an essential component of modern life. It is often misunderstood as a tool only necessary for military personnel or high-stakes organizations, but the reality is that cryptography plays a vital role in protecting the privacy and security of every individual.

In today's digital age, we generate and share vast amounts of sensitive information online, from personal identifiable data to financial transactions and confidential communications. This information is transmitted over insecure channels, making it vulnerable to interception and exploitation by malicious actors. Cryptography provides a solution to this problem by enabling secure communication over an insecure medium, ensuring that only authorized parties can access the encrypted information.

For instance, when you log in to your online banking account, cryptography ensures that your password and sensitive financial data are protected from prying eyes. Similarly, when you communicate with friends and family over messaging apps, cryptography safeguards your conversations from being intercepted or read by unauthorized entities. In essence, cryptography is the unsung hero that protects our digital lives and maintains the trust we have in online transactions. **Go to Appendix to see more about how a naive encryption might work.**

In this section, we will delve into the two fundamental types of cryptography: symmetric key cryptography and asymmetric key cryptography. We will explore their underlying principles, advantages, and limitations, setting the stage for understanding the role of quantum key distribution in secure communication.

A. Symmetric Key Cryptography

Symmetric key cryptography, also known as secret-key cryptography, is a type of encryption where the same secret key is used for both encrypting and decrypting the data. The basic idea behind symmetric key cryptography is that both the sender and the receiver share a common secret key, which is used to lock (encrypt) and unlock (decrypt) the message.

Some famous algorithms in symmetric key cryptography include AES (Advanced Encryption Standard), DES (Data Encryption Standard), and Blowfish. These algorithms use various techniques, such as substitution and permutation, to transform plaintext data into unreadable ciphertext.

The locking process involves encrypting the plaintext data using the shared secret key. The encryption algorithm takes the plaintext and the secret key as inputs, producing a ciphertext output that can be transmitted securely over an insecure channel. The unlocking process, decryption, is simply the reverse of encryption, where the receiver uses the same secret key to decrypt the ciphertext, recovering the original plaintext.

While symmetric key cryptography provides fast and efficient encryption, it has some significant drawbacks. One major limitation is the problem of key distribution: how do the sender and receiver establish a shared secret key without actually meeting in person? This is particularly challenging in scenarios where the parties are geographically distant or have no prior relationship. Another drawback is that if the secret key is compromised, the entire system becomes vulnerable to attacks.

B. Asymmetric Key Cryptography

Asymmetric key cryptography, also known as public-key cryptography, is a type of encryption where a pair of keys is used: one for encrypting and another for decrypting. The basic idea behind asymmetric key cryptography is that each user has a unique pair of keys: a public key for encryption and a private key for decryption. Some famous algorithms in asymmetric key cryptography include

RSA (Rivest-Shamir-Adleman), elliptic curve cryptography, and Diffie-Hellman key exchange. These algorithms use complex mathematical concepts, such as prime factorization and modular exponentiation, to ensure secure communication. The locking process involves encrypting the plaintext data using the receiver's public key. The encryption algorithm takes the plaintext and the public key as inputs, producing a ciphertext output that can be transmitted securely over an insecure channel. The unlocking process, decryption, is performed by the receiver using their private key, which is never shared with anyone. Asymmetric key cryptography provides several advantages over symmetric key cryptography, including simplified key distribution and management. Since each user's public key can be freely distributed, establishing a secure connection no longer requires a pre-shared secret key. Additionally, if an attacker obtains a user's public key, it does not compromise the security of the system. However, asymmetric key cryptography also has some drawbacks. One major limitation is that these algorithms are typically slower and more computationally intensive than symmetric key algorithms, making them less suitable for high-bandwidth or real-time applications. Another drawback is that the security of these systems relies on complex mathematical assumptions, which can be vulnerable to advances in computing power or new cryptanalytic techniques.

III. BREAKING RSA

One of the most widely used encryption systems in modern times is the RSA algorithm, a cornerstone of secure communication. However, the advent of quantum computers has posed a significant threat to its security. Shor's algorithm, a quantum algorithm, has been shown to be capable of breaking RSA encryption efficiently, rendering it vulnerable to attacks.

In this section, we will delve into the inner workings of the RSA algorithm and explore how it can be broken using Shor's algorithm. We will examine the most challenging step in breaking RSA, which surprisingly remains constant across breaking most other classical algorithms as well. To provide a comprehensive understanding, we will introduce the basics of the General Number Field Sieve (GNFS), currently the best classical algorithm for factoring large numbers, and compare it with Shor's algorithm in terms of time complexities.

A. RSA Encryption Algorithm

The RSA encryption algorithm, developed by Rivest, Shamir, and Adleman, is a widely used asymmetric encryption technique. The basic idea behind RSA is that Alice prepares a key pair, consisting of a public key and a private key, and shares the public key with others.

To encrypt a message, Bob uses Alice's public key to perform a locking operation, transforming the plaintext into an unreadable ciphertext. Conversely, to decrypt the ciphertext, Alice uses her private key to unlock the message, recovering the original plaintext.

The locking and unlocking operations can be mathematically represented as follows:

Let m be the plaintext message, (e, n) be Alice's public key, and (d, n) be Alice's private key. The locking operation (encryption) is given by:

$$c \equiv m^e \mod n$$

where c is the ciphertext.

The unlocking operation (decryption) is given by:

$$m \equiv c^d \mod n$$

In addition to ensuring confidentiality, RSA also provides authentication and non-repudiation features. To achieve this, it is recommended that Alice also "lock" the encrypted message with her own private key before sending it to Bob. This creates a digital signature, which allows Bob to verify that the message was indeed sent by Alice and not someone else. If the ciphertext can be decrypted using Alice's public key, it proves that she was the one who locked the message with her private key.

The RSA key pair generation algorithm is as follows:

1. Choose two large prime numbers, p and q, such that $p \neq q$.
2. Compute $n = p \times q$, the modulus.
3. Calculate $\phi(n) = (p - 1) \times (q - 1)$, Euler's totient function.
4. Choose an integer e, the public exponent, such that $1 < e < \phi(n)$ and $\gcd(e, \phi(n)) = 1$.
5. Compute d, the private exponent, such that $d \times e \equiv 1 \pmod{\phi(n)}$.
6. The public key is (e, n), and the private key is (d, n).

B. Breaking RSA

When it comes to breaking the RSA encryption algorithm, the values n and e are publicly available. The ultimate goal is to find the private exponent d, which can be achieved by first computing Euler's totient function $\phi(n)$. To do this, we need to know the prime factors p and q of n, which are not publicly known.

Given that n is a product of two large prime numbers, p and q, the only information available is that p and q are prime factors of n. Therefore, to break RSA, we need to factorize n into its prime factors p and q. Once we have these values, computing $\phi(n)$ is straightforward, and subsequently, finding d given e becomes a trivial task.

However, the catch lies in the fact that factorizing large numbers like n is an extremely time-consuming process. To put this into perspective, typical values of n lie in the range of 2^{2048} ($\approx 10^{600}$). For reference, in 1994, a decimal number with 129 digits (RSA-129) was factorized [1, 2] with the help of 1,600 computers collaborating over the internet and took 6 months. Later, in 2009, a binary number with 768 digits (232 decimal digits) was broken [3] using approximately 2,000 years of computational power (RSA-768).

It is not just RSA whose security relies upon the fact that prime factorization is tough, most other more advanced algorithms stand on the same pillars.

C. Prime Factorization Algorithms

The security of RSA relies heavily on the difficulty of prime factorization. In this section, we will discuss two prominent algorithms for prime factorization: the General Number Field Sieve (GNFS) and Shor's algorithm.

1. General Number Field Sieve (GNFS)

The GNFS is currently the best known classical algorithm for prime factorization. It is a variant of the number field sieve, which is a family of algorithms that exploit the properties of algebraic number fields to find non-trivial factors of large composite numbers.

In essence, the GNFS works by constructing an algebraic number field and then using it to find a non-trivial factor of the given composite number. The algorithm involves several intricate steps, including polynomial selection, sieving, and linear algebra. However, delving deeper into the workings of GNFS is beyond the scope of this text.

The time complexity of GNFS is approximately $L\left[a = \frac{1}{3}, c = \left(\frac{64}{9}\right)^{1/3}\right]$, where L denotes the L-notation, a measure of computational complexity. Specifically,

$$L(n, a, c) = \exp\left(c(\log n)^a (\log\log n)^{1-a}\right).$$

In this context, n represents the size of the input number.

2. Shor's Algorithm

On the quantum front, Shor's algorithm is a groundbreaking approach to prime factorization. This quantum algorithm, proposed by Peter Shor in 1994, exploits the principles of quantum parallelism and modular exponentiation to find non-trivial factors of large composite numbers.

In brief, Shor's algorithm works by using a quantum register to perform a Fourier transform on the function $f(x) = a^x \mod n$, where a is a random integer relatively prime to n. The resulting interference pattern is then analyzed to recover the prime factors of n.

The time complexity of Shor's algorithm is approximately $O(\log^2 n \cdot \log\log n)$, where n represents the number of bits in the input number. This exponential speedup over classical algorithms like GNFS has significant implications for the security of RSA and other cryptographic systems relying on prime factorization.

We can see in Figure 1 how these two time complexities compare against each other

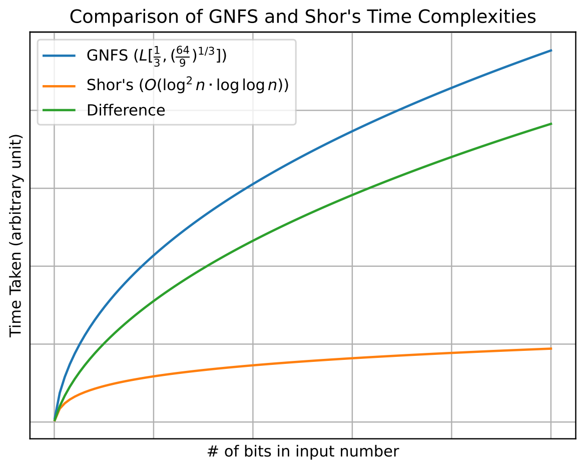

FIG. 1: Comparizon of GNFS and Shor's Time Complexities: The axes have not been labelled intentionally. This is because this is just for visualization how the two complexities compare against each other. We can see how the difference (green line) has a growing trend.

IV. BACKGROUND FOR QUANTUM KEY DISTRIBUTION

As we have seen, the advent of quantum computation has made it increasingly easy to break classical cryptographic algorithms. However, instead of rendering cryptography obsolete, quantum computing has come to its rescue by proposing the use of old and seemingly obsolete symmetric key distribution algorithms, such as AES, DES, and Blowfish, which do not rely on prime factorization and are much more robust, provided we can find a secure way to distribute the key. To ensure the secure distribution of keys, quantum computing has led to the development of various Quantum Key Distribution (QKD) algorithms.

In order to understand these QKD algorithms, it is es-

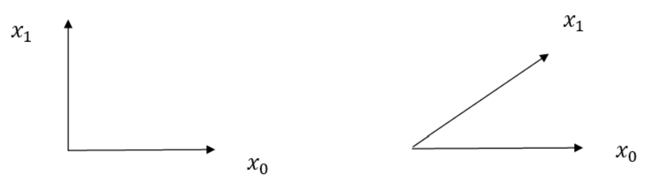

FIG. 2: Non-orthogonality Principle: in first case x_0 and x_1 are orthogonal, hence can be distinguished easily, while in the second case, x_0 and x_1 are non-orthogonal. Hence, there exists no measurement which can distinguish these two states.

sential to delve into some fundamental concepts in quantum mechanics that are thoroughly exploited in these protocols. This section will provide a background on the principles of quantum mechanics that form the basis of QKD, including the uncertainty principle, non-orthogonality principle, no-cloning theorem, and entanglement. By grasping these concepts, we will be well-equipped to explore the fascinating realm of Quantum Key Distribution in subsequent sections.

A. Uncertainty Principle

In quantum mechanics, the act of measurement has a profound impact on the system being measured. The uncertainty principle, formulated by Werner Karl Heisenberg, states that measuring one observable of a quantum system necessarily disturbs the other observables of the system. Consequently, it is impossible to simultaneously measure the values of two non-commuting observables with infinite precision.

In the context of quantum communication, this principle has significant implications. If Alice encodes information in quantum states and transmits them to Bob through a classical channel, any attempt by an eavesdropper to decode the message by performing a measurement will inevitably disturb the state, thereby introducing errors. This inherent property of quantum mechanics ensures that any unauthorized attempt to measure the quantum state will be detectable.

B. Non-Orthogonality Principle

Consider a two-state quantum system, where $|\psi_1\rangle$ and $|\psi_2\rangle$ are represented as:

$$|\psi_1\rangle = a|0\rangle + b|1\rangle \text{ where } |a|^2 + |b|^2 = 1$$

$$|\psi_2\rangle = c|0\rangle + d|1\rangle \text{ where } |c|^2 + |d|^2 = 1$$

The non-orthogonality principle states that if $|\psi_1\rangle$ and $|\psi_2\rangle$ are non-orthogonal, i.e., $\langle\psi_1|\psi_2\rangle \neq 0$, then it is impossible to perfectly distinguish between these two states. This is shown in Figure 2.

In the context of quantum communication, this principle has significant implications. If the encoded bits 0 and 1 are represented by non-orthogonal states, any attempt by an eavesdropper, Eve, to decode the message will introduce errors. This is because there exists no measurement that can perfectly distinguish between the non-orthogonal states $|\psi_1\rangle$ and $|\psi_2\rangle$. Therefore, the non-orthogonality principle provides a fundamental basis for secure quantum key distribution protocols.

C. No Cloning Theorem

The no-cloning theorem, proven by Wootters and Zurek in 1982, states that it is impossible to create a perfect copy of an arbitrary unknown quantum state. In other words, if we have a qubit in an unknown state $|\psi\rangle$, there exists no physical process that can transform the state into two identical copies, $|\psi\rangle \to |\psi\rangle \otimes |\psi\rangle$. This fundamental principle has significant implications for quantum key distribution (QKD). Specifically, it prevents any eavesdropper from creating exact copies of the quantum signal, thereby ensuring that Alice and Bob can detect any attempt to measure or copy their quantum communication. While an eavesdropper may be able to create approximate copies of the quantum state, gaining some information about the channel, Alice and Bob must eliminate this partial information to ensure a perfectly secure key.

D. Entanglement

Entanglement is a fascinating phenomenon in which multiple quantum particles interact in such a way that their individual properties cannot be described independently. Instead, they form a single, correlated system. When a particular property is measured on one particle, the opposite state is instantaneously observed on the other entangled particle, regardless of the distance between them. This effect, known as quantum non-locality, is demonstrated by the following equation:

$$|\psi\rangle_{AB} = \frac{1}{\sqrt{2}}(|0\rangle_A \otimes |1\rangle_B + |1\rangle_A \otimes |0\rangle_B)$$

where $|\psi\rangle_{AB}$ represents the entangled state of particles A and B. Measuring a property on particle A will instantaneously determine the corresponding property on particle B.

Entanglement forms the basis for various quantum information protocols, including the E91 protocol, which we will discuss in greater detail in subsequent sections.

The connection between entanglement and the CHSH inequality will also be explored in the context of QKD.

E. CHSH Inequality

The CHSH inequality is a fundamental concept in quantum mechanics that establishes a limit on the correlations that can exist between distant systems if they obey classical physics. Introduced by John Bell in 1964 and later refined by John Clauser, Michael Horne, Abner Shimony, and Richard Holt in 1969, this inequality has become a cornerstone of quantum theory. The CHSH inequality is typically expressed as:

$$\langle C\rangle = \langle M_A.M_B\rangle + \langle M'_A.M_B\rangle + \langle M_A.M_{B'}\rangle - \langle M_{A'}.M_{B'}\rangle \leq 2$$

where $\langle M_A.M_B\rangle$ represents the correlation between two systems measured in settings A and B, respectively. In classical physics, the correlations between distant systems are bounded by this inequality, with a maximum value of 2.

To understand why entangled states can violate this inequality, let's break down the expression for $\langle M_A.M_B\rangle$:

$$\langle M_A.M_B\rangle = \mathbb{P}(ab) - \mathbb{P}(a\bar{b}) - \mathbb{P}(\bar{a}b) + \mathbb{P}(\bar{a}\bar{b})$$

where $\bar{a}$ and $\bar{b}$ represent the complementary outcomes of a and b, respectively. Similarly, we can define $\langle M_A.M_{B'}\rangle$, $\langle M'_A.M_B\rangle$, and $\langle M_{A'}.M_{B'}\rangle$. Substituting these expressions into the CHSH inequality, we get:

$$\langle C\rangle = 2\left|\mathbb{P}(ab) + \mathbb{P}(a'b') - \mathbb{P}(a\bar{b}) - \mathbb{P}(\bar{a}b)\right|$$

Using the properties of probabilities, we can see that each term in the expression is bounded by $[-1,1]$. Therefore, the entire expression is bounded by $[-2,2]$, which proves the CHSH inequality.

However, when dealing with entangled quantum particles, such as maximally entangled EPR pairs, the correlation parameter $\langle C\rangle$ can exceed this bound. In fact, for these systems, we get $|\langle C\rangle| = 2\sqrt{2}$, which clearly violates the condition $|\langle C\rangle| \leq 2$. This violation of the CHSH inequality is a hallmark of quantum non-locality and has been experimentally confirmed in various systems.

V. QUANTUM KEY DISTRIBUTION PROTOCOLS

Quantum Key Distribution (QKD) is a method of secure communication that uses quantum mechanics to encode, transmit, and decode messages. QKD protocols can be broadly classified into two categories: Prepare and Measure protocols, where a sender prepares a quantum state and sends it to a receiver, and Entanglement-based protocols, where entangled particles are distributed between two parties. In this section, we will explore two specific QKD protocols: BB84 from the Prepare and Measure category and E91 from the Entanglement-based category. We will delve into the theoretical aspects of these protocols and discuss how they detect eavesdropping attempts, as well as our implementation using Qiskit and Flask libraries in Python.

A. BB84

The BB84 protocol, developed by Charles Bennett and Gilles Brassard in 1984, is a Quantum Key Distribution (QKD) protocol that leverages the quantum phenomenon of superposition. A quantum system can exist in multiple states simultaneously until measured in a superposition state.

In the BB84 protocol, Alice sends qubits to Bob through a quantum channel. If an eavesdropper, Eve, tries to intercept the communication, Alice and Bob have a mechanism to detect her presence in the channel. Upon detection, they discard the transmitted key and restart the process to evade Eve's attempt.

Here are the steps for the BB84 protocol:

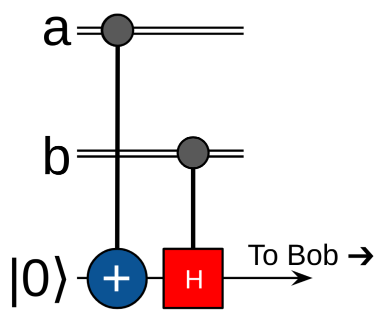

FIG. 3: Circuit Alice uses to prepare the qubits for Bob.

1. **Alice generates and sends:** Alice generates the key string to be sent, which we will call the binary string a. She also makes a choice of bases, either X or Z, corresponding to each bit in the key string. To represent her base choice, she prepares another binary string consisting of 0s and 1s, where 0 denotes the X basis and 1 denotes the Z basis. This results in two strings, a and b, of the same length, with one representing the key and the other representing the choice of bases. Then, she passes each bit pair, consisting of one element from a and one element from b, through the circuit shown in Figure 3. As a result, the prepared qubits look like this:

   Alice then sends these qubits through the quantum channel to bob.

2. **Bob performs measurements:** Bob, unaware of Alice's basis choices, prepares his own random choice of bases for each qubit. He then measures each qubit in his chosen basis. To accomplish this,

value a	value b	qubits
0	0	$\|0\rangle$
0	1	$\|+\rangle$
1	0	$\|1\rangle$
1	1	$\|-\rangle$

Bob performs a direct measurement when he wants to measure in the X basis, and applies a Hadamard gate followed by a measurement when he wants to measure in the Z basis. Due to his random choice of bases, Bob will correctly measure approximately 50% of the qubits in their original basis, while the remaining 50% will be measured in the incorrect basis. Unfortunately, the qubits measured in the wrong basis are useless and cannot be used to establish a secure key.

3. **Sharing of Bases:** Now that Bob has measured all the qubits, he doesn't know which ones were measured in the correct basis. However, Bob and Alice can now publicly share their basis choices with each other over an insecure classical channel. By doing so, they can identify which qubits were measured in the wrong basis. Both parties then discard these incorrect bits, leaving only the bits that were measured in the correct basis. The resulting string of remaining bits is called the "key".

1. *Detecting the Presence of Eve*

Although Alice and Bob have established a shared key, they still need to verify that the qubits were not eavesdropped on during transmission. By eavesdropping, we mean that Eve might have measured the qubits before Bob had a chance to do so. To perform measurements, one needs to know the bases, but Alice's basis choices are only made public after Bob confirms he has completed his measurements. As a result, Eve has no option but to measure the qubits in randomly chosen bases. However, as soon as Eve makes her measurements, the qubits will collapse to specific states, effectively creating a completely new set of qubits that differ from those originally prepared by Alice. It's as if Eve had created and sent an entirely new set of qubits to Bob.

Had Eve not eavesdropped, the corrected keys prepared by Alice and Bob on their respective sides would be identical, resulting in a 100% match. However, if Eve had eavesdropped, Bob would have measured something completely different from what Alice sent. Consequently, the match with Alice's key would be purely coincidental, yielding a mere 50% match.

Following the preparation of the corrected key, one party (say Alice) randomly selects a portion of the corrected key (in our implementation, 30% of the corrected key is used) and sends it to the other party through the classical channel. This transmission includes information about which bit corresponds to which position in the corrected key. The receiving party (Bob) then compares this sample with their own key. If the match approaches 100%, they can be confident that Eve did not eavesdrop. However, if the match is close to 50%, they will suspect eavesdropping and reject the keys. Upon successful verification, the 30% bits used for authentication are discarded, and the remaining bits constitute the final secure key.

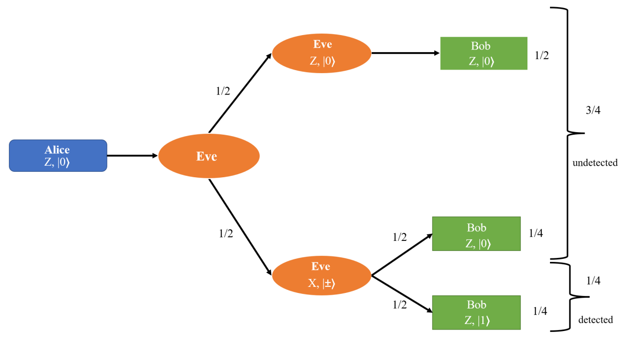

FIG. 4: Probability of detecting Eve

As depicted in Fig. 4, when Alice and Bob compare their bits to detect eavesdropping, there's a 75% chance that Eve remains undetected for each bit. This is because there's a 50% chance that Bob and Alice measure in the same basis, and if Eve also chooses this basis, she'll go unnoticed. Even if Eve picks the wrong basis, there's still a 50% chance that Bob randomly measures the correct value, despite measuring in a completely different basis. This results in a 75% probability of Eve evading detection for each individual bit.

When Alice and Bob reveal n bits of their shared secret key, the probability of Eve remaining undetected for all n bits is $(\frac{3}{4})^n$. Conversely, the probability of detecting Eve decreases exponentially with n, making it increasingly likely that eavesdropping will be detected as more bits are compared. For instance, if we reveal just 25 bits, the probability of not detecting Eve can be calculated as follows: $100 \times (\frac{3}{4})^{25}\% \approx 0.075\%$ effectively guaranteeing her detection.

B. E91 Protocol

Say, Alice wants to communicate with Bob. They assign the job of generating the entangled particles in a singlet state to Charlie.

$$|\psi_{sing}\rangle = \frac{1}{\sqrt{2}}(|\uparrow\downarrow\rangle - |\downarrow\uparrow\rangle)$$

Charlie generates and distributes them to Alice and Bob. When they both measure in the same basis, the probability that Alice measures +1 and Bob measures -1 is $\frac{1}{2}$, similarly the probability that Alice measures -1 and Bob measures +1 is also $\frac{1}{2}$. The probability that both of them measure the same is always zero.

They assign bit 0 to +1 result and bit 1 to -1 result. As long as they are measuring in the same basis, their results are exactly the opposite. They use this fact to establish the key. This is how the protocol works.

1. Alice and Bob chooses 3 bases each. They make sure both of them take two bases which are perpendicular to each other and one base midway between the two. The two perpendicular bases chosen by them won't be same. Also, among the 3 bases of each, two will be common in both and they would have another unique on their own. One example of their choice of bases is shown in Figure 5

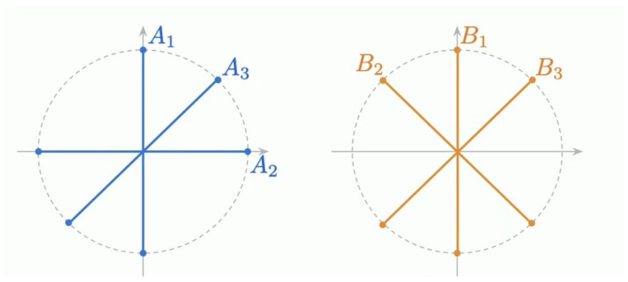

FIG. 5: Alice's basis choices in blue and Bob's choices in orange

2. After generating every entangled pair, charlie will send one qubit to Alice and another to Bob. They will choose one of their 3 bases and do the measurement in that basis.

3. After all the measurements are made, they exchange information about the basis they used through the public channel just as before in BB84.

4. Now that they have each other's bases, they divide their measurements into two groups: **Group 1** has all the bits which were measured by them in the same basis (which comes out to be about 1/3 part) and **Group 2 has all the bits which were measured in the wrong basis**. Now, Bob has measured the complementary bits of Alice, so he inverts his bits to get the final key.

1. Detecting the Presence of Eve

1. The particles are ideally assumed to be entangled as long as there was no eavesdropping. Once there is eavesdropping or interception by Eve on either of the particles of the entangled pair, the particles are no more entangled, and the wave function governing the entanglement has collapsed, which means the particles now have a definite spin. In other words, both particles have elements of physical reality.

2. Simply put, Eve's interception has introduced elements of physical reality into the system.

3. In order to finally establish the secret key, Alice and Bob should test whether the particles they have been receiving are indeed entangled as mentioned before, this helps them detect the interception by Eve.

4. Now is when the CHSH inequality comes in handy. Recall that only the correlations arising from the entangled particles violate the inequality while the correlation arising from the particles with elements of physical reality satisfies the inequality.

5. The group 2 measurement bases (A1, B3), (A1, B2), (A2, B2) and (A2, B3) are used to check the CHSH inequality, it is as follows:

$$|\langle C|C\rangle| = |\langle M_{A1}.M_{B2}|M_{A1}.M_{B2}\rangle + \langle M_{A1}.M_{B3}|M_{A1}.M_{B3}\rangle + \langle M_{A2}.M$$

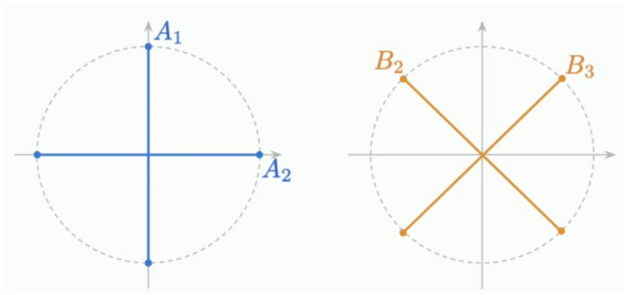

FIG. 6: Group 2 measurement bases

6. The left-hand side is called the test statistic, to calculate this value, the 4 correlations have to be first calculated. This means Alice and Bob have to publicly share the measurement results of the group 2 basis with each other and it is fine to do so, as these results are not being used to establish the key.

7. Once both of them obtain these results from each other, they calculate the correlations. For eg: To calculate $\langle M_{A1}.M_{B2}|M_{A1}.M_{B2}\rangle$, they count the number of measurements where Alice picked A1 and Bob picked B2, and then calculate the value $M_{A1}.M_{B2}$ of all these measurements. Finally, they add the $M_{A1}.M_{B2}$ value of all these measurements and divide by the number of measurements.

8. If the test statistic is lower than or equal to 2, they abort. If it is greater than 2, they proceed to use the established key in the conventional symmetric cryptography.

[1] D. Atkins, M. Graff, A. K. Lenstra, and P. C. Leyland, The magic words are squeamish ossifrage, in *Advances in Cryptology - ASIACRYPT '94, 4th International Conference on the Theory and Applications of Cryptology, Wollongong, Australia, November 28 - December 1, 1994, Proceedings*, Lecture Notes in Computer Science, Vol. 917 (Springer, 1994) pp. 263–277.

[2] enThe Magic Words are Squeamish Ossifrage (2023), page Version ID: 1181720658.

[3] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev, and P. Zimmermann, Factorization of a 768-bit rsa modulus, in *Advances in Cryptology – CRYPTO 2010*, edited by T. Rabin (Springer Berlin Heidelberg, Berlin, Heidelberg, 2010) pp. 333–350.