How to Securely Deploy Quantum Key Distribution

Greesan Gurumurthy[1], Eric Yu[1], Aadarsh Venugopal[1], Manisha Sri Suresh[1], Daniel Loran[1], and David Pham[1]

[1]University of California, Davis

June 13, 2022

1 Abstract

Quantum key distribution (QKD) is a "quantum-resistant" key exchange protocol that is promising as an alternative or replacement for conventional key exchange systems. QKD generates keys by encoding information into qubits with unknown states until measured. The no-cloning theorem guarantees that an eavesdropper cannot create perfect copies of unknown quantum states, so obtaining information about the qubits requires measuring them. An eavesdropper will reveal their presence because their measurements affect the measurements of the users of the QKD system. This property guarantees security of the quantum channel by ensuring that an eavesdropper will always be detectable. However, QKD also consists of a classical channel for authentication, which is generally not guaranteed the same security as the quantum channel. Our work focuses on addressing the potential vulnerabilities of QKD by exploring various modifications and additions to the standard QKD implementation. This report investigates previous works that address drawbacks of QKD such as vulnerability to denial-of-service attacks and attenuation on the quantum channel. Additionally, we implemented a QKD simulator that roughly replicates and explains the actions involving both the quantum and classical channels of a standard QKD system while also providing various methods to , also providing Qiskit interoperability.

2 Introduction

Quantum key distribution, shortened as QKD, is a secure communication method that implements a cryptographic protocol which involves components of quantum mechanics. More specifically, QKD uses quantum physics to securely exchange encrypted symmetric keys. QKD works to solve the problem of key distribution by allowing two remote parties to exchange cryptographic keys that satisfy the laws of physics. QKD can serve to establish keys between parties and the deployment of QKD works to secure communications over a combined quantum and classical channel.

With the increased threats to quantum computing there is particular concern on deployment of QKD. As such, this research will primarily focus on understanding the security concerns that exist and find possible solutions to deploy QKD securely with the integration of classical network key management and monitoring processes. Some particular areas that we will explore are how the eavesdropper can interrupt qubits, resulting in a collapse to the quantum state and vulnerabilities to denial of service attacks. In addition, we will look at the trade-offs that exist between other quantum-resistance measures. If there are existing solutions we will conduct security analysis and determine how it addresses National Security Agency (NSA) concerns.

3 Background

3.0.1 Classical key-exchange

Conventional key exchange protocols like RSA and Diffie-Hellman struggle to stay relevant or reliable in the face of quantum computers. Classical protocols rely on encryption schemes like prime factorization of large integers, which are secure against classical computers. With the rise of quantum computers and quantum algorithms (like Shor's algorithm and quantum Fourier transform), security may not be guaranteed for such systems. As an example, RSA is generally secure against brute-force attacks using a classical computer because so far, solving the prime factors for very large numbers is a computationally difficult task. However, quantum computers operating on Shor's algorithm can perform this task fairly quickly, potentially making classical encryption schemes obsolete in the near-future. This concern warrants the development of systems that are robust against these type of quantum attacks, with QKD being a notable candidate.

3.0.2 Quantum attacks

Quantum key distribution (QKD) has been improving over the past 25 years, and at this rate, we might see some private cloud networks supporting infrastructure for government, defense and health departments. The system needs to be cost-effective and guarantee solid security for real-time applications. Theoretical security proofs may be insufficient to evaluate the system accurately as the hardware to implement quantum mechanics is still improving. There could be some engineering constraints that may contradict the proofs; this leads to deviation in the theoretical and practical evaluation process. Some attacks can be demonstrated, such as detector blinding, while others can be coherent and abstract concepts. The standard methodology in classical cryptography called Common criteria is used to evaluate the attacks on multiple factors. Two practical methods to launch a saturation attack on the detector at its detection limit are evaluated using attack potential to QKD. The attack rating (attack potential) is calculated on many factors such as equipment, expertise, window of opportunity and knowledge of target of evaluation(TOE). Each factor will be assigned an appropriate numerical value and we calculate the overall attack potential by summation of all factors; low attack potential translates that it is easy to mount the attack successfully. [3]

The saturation attack takes advantage of the homodyne detector's linearity range and biases the excess noise estimation. It is also assumed that the QKD system is using Gaussian-modulated coherent states (GMCS) protocol. Below is the step-by-step analysis of this attack and the scheme is highlighted in Figure 1.

- Alice prepares coherent states of quadratures XA , PA, each quadrature is modulated according to a Gaussian distribution of variance VA, and sends the state to Bob through the quantum channel.
- Bob randomly measures one of the quadratures that he received from Alice using a balanced homodyne detector. This gives new quadrature measurements XB and PB, with variance VB. By measuring the fraction of sent and received data, users can estimate the channel transmittance T and then the excess noise ξ.
- The calibration of the detector before each protocol run will establish the linear range interval $[\alpha_1, \alpha_2](\alpha_1 < 0 < \alpha_2)$. If it receives any data that is out of this range, then the outer values of the range are considered at the receiver end.
- Eve (the attacker) will take this range limit as an advantage and try to saturate the detector using a simple intercept-resend attack. It intercepts the data from Alice and introduces Gaussian modulation of appropriate value; this value is added in quadratures(Xe, Pe) to exceed the linear limit of the detector.
- Alice and Bob calculate the transmittance and excess noise which are influenced by variance and Gaussian modulation. Eve optimizes the value such that excess noise drops to null and eavesdropping remains undetected.

The implementation of coherent strategy attacks might be complex and expensive, therefore an incoherent strategy attack is considered which uses an external laser to saturate the detector. This external laser is present with the coherent states such that their combined modulation will be enough to exploit the limited range of the detector as discussed above. This will improve the setup on the present side and give higher probability to keep the detector in a saturated state. This attack is similar to a blinding attack (continuous light on the detector) that takes advantage of a single-photon detector, this gives Eve full control of the detection outcomes.

For rating these two strategies, the knowledge of TOE is considered to be the same and high. The main difference occurs in expertise and setup; for coherent attacks you will need expertise on the coherent optical communications as we need to maintain the phase locked between Alice and Bob. Expertise is not required for incoherent attack as we don't need to bother about being phase locks; the simple implementation requires less setup compared to coherent strategy. Again due to phase locking, the window of opportunity for coherent strategy is difficult and moderate for incoherent. By considering all the factors the attack potential is calculated to be 26 and 14 for coherent and incoherent correspondingly. This implies that incoherent strategy attacks are easier to implement and have high probability to mount the attack successfully.

There are some ways to battle this attack, adding an additional hardware system after the detector to overcome the linear range problem. Hardware installation is not cost effective, so we can also go in the software route where an extra layer might be useful after the detector.

Furthermore, lack of authentication over the classical channel can lead to man in the middle attacks, where an eavesdropper imitates Bob when communicating to Alice, and the eavesdropper imitates Alice when communicating with Bob. While this requires higher capital to prepare the quantum equipment, it can lead an eavesdropper learning the secret key without Alice or Bob having any knowledge of the eavesdropping. To counter such acts, all communication across the classical channel must be authenticated.

4 Previous Work

4.1 QKD Implementations

4.1.1 BB84 Protocol

BB84 is the most prevalently taught QKD protocol. As a QKD protocol, BB84 uses a quantum channel to send qubits in the form of polarized photons from the sender to the receiver, and a classical channel to assist in communication of data indirectly correlated to key bits. It consists of multiple phases including sending encoded/polarized qubits, decoding/depolarizing qubits, announcing encoding and decoding bases, sifting, checkbit sequence comparison, and final key generation.[4]

1. Sending Encoded qubits: the initiator of the protocol, Alice, starts with photons all set to a certain state (either zero or one), based on Alice's beginning bit string (done by inserting X gates to switch zero-valued qubits to one-valued qubits, according to the bit string). She will encode photons through the use of rectilinear and diagonal filters. Each qubit passing through either filter will either be encoded in the 0 or 1 basis state (rectilinear) or the + or - basis state (diagonal). This will then be passed through the quantum channel to the party that Alice wants to share a private key with, Bob. The quantum channel can been either eavesdropped on (difficult if quantum channel is wired), or affected by noise. This noise and eavesdropping can potentially destroy the information held by qubits.

2. Decoding received qubits: the receiver of the qubits, Bob chooses a random basis to decode each qubit that Alice sends through the quantum channel. If Bob chooses the opposite decoding basis from Alice's encoding basis, this will result in the loss of the qubit's information, but if Bob chooses the right decoding basis, he will be able to decode the qubit value properly, unless that qubit is affected by eavesdropping or noise.

3. Sifting qubits based on basis: If Alice and Bob conflict in basis sequence chosen ($P = ((2^n - 1)/2^n)$), the extraneous qubits corresponding to mismatching basis selections must be sifted out from the key.

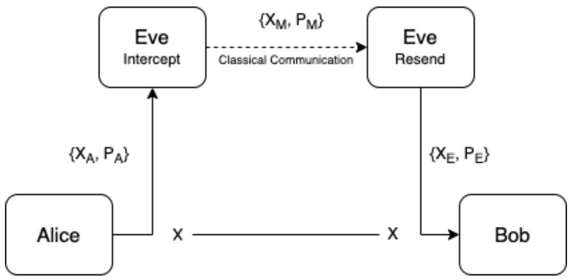

Figure 1: Scheme that represents the saturation attack

Both Alice and Bob announce their basis selections over the classical channel, and disregard the qubits associated with these basis selections.

4. Checkbit sequence comparison: This step purely exists as a way to monitor for noise and eavesdropping. First, Alice and Bob must decide over the classical channel a subsequence of bits to check the value of. After communicating the subsequence, Alice sends the values of the bitstring she started with corresponding to indices of the subsequence, while Bob sends the values of the qubits he measured corresponding to the indices of the subsequence. If these do not match, that indicates that the channel is either noisy or getting eavesdropped on. Both parties can decide a boundary ratio upon which to deem the current iteration of QKD a failure, and repeat QKD.

Single bit eavesdropping has a 75% chance of going undetected, as approximately half of the qubits will be sifted, and eavesdroppers will guess the correct basis half the time, not affecting the state of the qubits. Each qubit eavesdropped on will further decrease the chance of going undetected exponentially.

4.1.2 Ekert Protocol

The Ekert Protocol (E91) builds off the BB84 protocol; a quantum channel sits in between two users which qubits are sent in the form of polarized photons. However, it uses quantum entanglement to distribute qubits to both parties at the same time. Because entangled quantum particles reveal themselves once one of them is measured, a party can discretely communicate a key simultaneously. This property defeats eavesdropping as the attacker cannot both maintain entangled bits and read the key (Bell's Theorem). E91 uses 3 bases for measuring qubits instead of two, making more sensitive to noise and error. [1]

4.2 Information Reconciliation

4.2.1 Cascade Protocol

Cascade protocol looks to correct qubit errors received through iterative shuffling and parity checks. The receiver divides the check bit key into chunks and shuffles it into a new key, assigning a parity bit to each chunk to check how many bits in the chunk are wrong. By confirming the parity of the chunks with the sender, the receiver can figure out which chunks have an incorrect parity and try to correct the chunk. Bits corrected later in the iteration process also corrects bits in the past iterations, creating a "cascading affect". By continuously splicing and shifting the chunks, the receiver can eventually correct every bit. However, the user must not splice too thinly as information about the key may leak. This protocol communicates over the classical channel without the need to encrypt/hide any information conveyed during the correction process. [7]

4.3 Classical Authentication

Classical authentication requires the use of a digital signature to confirm the sender's identity. Typically, signatures are generated using the asymmetric Digital signature scheme, a scheme that uses two distinct prime numbers in the form of discrete logs. The message is encrypted by multiplying it with one of the prime numbers, the public key. Decrypting the message requires the receiver to use the other prime number, usually kept secret, to uncover the message. It is very difficult to guess which number (secret key) can solve the problem, but easy decrypt the message once found. Hence the identity of the user is nonfungible. Use of other schemes, such as RSA or AES can also authenticate users. However, asymmetric schemes are not quantum safe, as quantum algorithms such as Grover's and Shor's algorithm can solve these problems where factorization is required.

4.3.1 Poly1305-AES

While common authentication protocols like Diffie Hellman must be worked upon to become quantum resilient especially against Shors Algorithm, using Wegman Carter Authentication along with AES proves to already be effective against quantum attacks, and removes the possibility of key exhaustion[6]. Built upon the AES, Poly1305-AES adds an authentication factor to the symmetric key scheme. It uses a 128 bit AES key, random 128-bit key, and a 128 bit nonce for each unique message as a seed for generator. The protocol creates a 128-bit MAC can be used to authenticate the user. Because the scheme is symmetric, there is no hints to guess the key besides brute forcing. Given the randomness and uniqueness of the random seed, the key cannot be recreated from one of the sources either.

4.3.2 PQC

Post Quantum Cryptographic algorithms look to provide quantum-safe algorithms without the use of quantum mechanisms such as BB84. AES is considered quantum safe, as increasing the key size greatly increases the difficulty of decrypting, while still using the classical channel. Other PQC protocols look to using other NP-Hard problems to generate asymmetric keys with limitations to insufficient key size performance or computationally difficult decryption/encryption scheme. A good balance of effort, portability, and key resistance is needed when considering protocols for the quantum future. [5]

4.4 DOS mitigation

4.4.1 SDN

Denial-Of-Service attacks can happen in quantum channels, as the interference of qubit can cause it to destruct. With the use of multiple channels, we can circumvent Denial-Of-Service attacks by measuring response times and picking the channel that is not being attacked. Using client-side programs, we are able define the quantum network through software. [2]

5 Our Work

5.1 High-Level Simulator Details

The simulator is dashboard that allows general functions and processes that are featured in a standard QKD system. Users select the key size they wish to generate, and the backend handles the quantum circuit generation via the Qiskit library (including key and basis selection). A slider for a noise parameter is an option for simulating realistic scenarios where noise disrupts the bits of the key. This slider is co-linked with the distance and time sliders as we found functions in research that correlated one another for a particular qkd setup [**rapdos**].

5.2 BB84 Implementation

The BB84 protocol is implemented using Qiskit. The quantum channel / qubits are implemented as Qiskit quantum circuits. While the circuit printed in the terminal visual is only a visual representation, variable qc holds the actual quantum circuit. The measure gates will only be half as effective, as half the time, they will fail to change the state of the qubit. As a result it would be recommended to use double the decoherence given further usage of the actual quantum circuit.

5.3 AES Implementation

Our simulator uses the PyCryptodome library to implement the Advanced Encryption Standard (AES) cipher in the classical (authentication) channel. The purpose of using AES is to verify the users of the QKD system and ensure that messages transmitted during a session are not from an unauthorized third-party. The standard implementation requires a 16-byte secret that is pre-shared between the users. For the purposes of replicating a genuine QKD system as closely as possible, we added to the simulation a process for verifying tags and ciphertexts sent in a QKD session using the pre-shared key. In addition, we implemented a pre-shared key generator that does not require communication between the users. This is explained further in the following section (Time-based Syncing).

5.4 Time-based Syncing Implementation

The standard authentication protocol in QKD typically requires a pre-shared secret that is agreed on between the transmitter and receiver. In general, this pre-shared secret is communicated through a classical channel, which may possibly add another layer of security concern. To address this potential challenge, we devised a method to generate identical pre-shared secrets independently using the current time (i.e. the UNIX time during the beginning of a QKD session). In our Python implementation of a time-based random number generator, the users of the QKD system independently call a function that returns the current UNIX time. To account for the discrepancies in precision, the time value is modified by integer division based on delay between the transmitter and receiver. A hash function is applied to introduce "randomness" to the modified time value, which is then passed as a parameter to seed the random number generator. This system allows for two identical shared secrets to be generated independently, which eliminates the requirement for a separate channel to pre-share them. Keys generated based on time-syncing can also be used to verify the quantum source. In a point-to-point QKD system, that is, two users communicate directly with each other, it is possible that a third party initiates a session with the users (this can be considered a man-in-the-middle attack). To ensure that a user is receiving legitimate messages, the quantum channel could also transmit the pre-generated secret, which would be used to verify the quantum source.

6 Discussion

Quantum key distribution appears to be very promising in the field of cryptography because of potential security vulnerabilities with the advancements being made in quantum computers. There are, however, several criticisms that should be addressed for implementing such a system. QKD is a key exchange system (not for communication) and would likely not have many other applications. The keys are susceptible to noise, requiring many resources to generate potentially small final keys. Adversaries can take advantage of this by introducing high attenuation to the channel, rendering the system unusable. The quantum channel in QKD is generally considered to be secure in the sense that any attempts to obtain information about the key is detectable due to the underlying quantum mechanics encoding each qubit. However, the standard implementation of QKD still requires the use of a classical channel for authentication, and the ability to detect eavesdroppers does not protect against DDoS attacks. Our simulator attempts to recreate a simple QKD system that implements some of the proposed solutions to these drawbacks on top of the standard BB84 implementation.

References

[1] Artur K. Ekert. “Quantum cryptography based on Bell’s theorem”. In: *Phys. Rev. Lett.* 67 (6 Aug. 1991), pp. 661–663. DOI: `10.1103/PhysRevLett.67.661`. URL: `https://link.aps.org/doi/10.1103/PhysRevLett.67.661`.

[2] E. Hugues-Salas et al. “Experimental Demonstration of DDoS Mitigation over a Quantum Key Distribution (QKD) Network Using Software Defined Networking (SDN)”. In: *2018 Optical Fiber Communications Conference and Exposition (OFC)*. 2018, pp. 1–3.

[3] Rupesh Kumar et al. “Experimental vulnerability analysis of QKD based on attack ratings”. In: *Scientific Reports* 11 (2021).

[4] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition.* Cambridge University Press, 2011. ISBN: 9781107002173. URL: `https://www.amazon.com/Quantum-Computation-Information-10th-Anniversary/dp/1107002176?SubscriptionId=AKIAIOBINVZYXZQZ2U3A&tag=chimbori05-20&linkCode=xm2&camp=2025&creative=165953&creativeASIN=1107002176`.

[5] Ray A. Perlner and David A. Cooper. “Quantum Resistant Public Key Cryptography: A Survey”. In: *Proceedings of the 8th Symposium on Identity and Trust on the Internet.* IDtrust ’09. Gaithersburg, Maryland, USA: Association for Computing Machinery, 2009, pp. 85–93. ISBN: 9781605584744. DOI: `10.1145/1527017.1527028`. URL: `https://doi.org/10.1145/1527017.1527028`.

[6] Alasdair B. Price, John G. Rarity, and Chris Erven. *A quantum key distribution protocol for rapid denial of service detection.* 2017. DOI: `10.48550/ARXIV.1707.03331`. URL: `https://arxiv.org/abs/1707.03331`.

[7] Bruno Rijsman. *cascade-python.* `https://github.com/brunorijsman/cascade-python`. 2020.