Security Analysis for BB84 Key Distillation

Sara Nikula[1][a], Anssi Lintulampi[1][b] and Kimmo Halunen[2][c]
[1]*VTT Technical Research Center of Finland, Oulu, Finland*
[2]*University of Oulu, Oulu, Finland*
fi *fi* *fi*

Keywords: Quantum Key Distribution, BB84, Classical Post-processing, Key Distillation, Security Analysis.

Abstract: Key distillation, also referred to as classical post-processing, plays a pivotal role in Quantum Key Distribution (QKD) protocols. Key distillation encompasses numerous subroutines, making the analysis of its overall security implications potentially challenging for those outside the research community. In this paper, we elucidate the role of the key distillation phase in QKD from a security standpoint. We begin by analyzing the different components of the key distillation phase individually, followed by an examination of the process as a whole. We then calculate the bit strength of the produced key, assuming that an attacker is executing an intercept and resend attack. For our analysis, we employ a practical key distillation implementation linked to a decoy state BB84 protocol as a case study. Our findings suggest that the security of the final key, post the key distillation phase, hinges on several factors. These include the theoretical security of the implemented subroutines, the total information leakage throughout the process, and the choices of subroutine parameters. Given these assumptions, we can distill 287 secure bits for every 1000 bits that undergo the key distillation procedure.

1 INTRODUCTION

To facilitate confidential communication between two parties, an encryption key is required. Traditionally, this encryption key is agreed upon using asymmetric public-key cryptographic protocols, such as RSA (Moriarty et al., 2016) or elliptic curves (Nir et al., 2018). However, these algorithms are vulnerable to the algorithm presented by Shor (1994), which can be executed on a cryptographically relevant quantum computer. This implies that future key agreement protocols will need to be updated to be resistant to quantum computing.

One solution to this issue is quantum-safe public key cryptography (also known as post-quantum cryptography, PQC), which is based on mathematical problems that are not easily solvable even with quantum computing. Another quantum-safe method for establishing secret encryption keys between two parties is Quantum Key Distribution (QKD). The primary advantage of QKD over classical post-quantum key agreement schemes is that the security of QKD is grounded in the laws of quantum physics, rather than on the assumption that certain mathematical problems are computationally difficult to solve. The first QKD protocol, BB84, was introduced in 1984 by Bennett and Brassard (2014). The security of the BB84 protocol is based on encoding bit information in the polarization of a single photon, with a randomly selected basis. Measuring the photon will destroy its original state, ensuring that any potential eavesdropping will be detected as an increased error rate.

Therefore, in theory, QKD should provide unconditionally secure encryption keys, as it cannot be compromised by any computational algorithms, whether classical or quantum. However, in practice, the situation is more complex due to imperfect implementations and the necessity for the classical key distillation phase.

The BB84 protocol is composed of two distinct phases: the quantum phase and the classical phase. The quantum phase alone is insufficient for the derivation of an encryption key. To ensure the encryption key is both secure and error-free, a key distillation procedure must be executed over the classical channel. The classical key distillation process encompasses the following sub-procedures: authentication, bit error estimation, error correction, and privacy amplification. These procedures collectively ensure the

[a] https://orcid.org/0000-0002-2299-8030
[b] https://orcid.org/0009-0002-7361-054X
[c] https://orcid.org/0000-0003-1169-5920

Nikula, S., Lintulampi, A. and Halunen, K.
Security Analysis for BB84 Key Distillation.
DOI: 10.5220/0012717500003767

In *Proceedings of the 21st International Conference on Security and Cryptography (SECRYPT 2024)*, pages 407-415
ISBN: 978-989-758-709-2; ISSN: 2184-7711

integrity and security of the encryption key.

In this paper, we present a comprehensive analysis of the key distillation implementation of the BB84 protocol from an information security perspective. Several publications already exist that focus on specific aspects of the key distillation process. For instance, the works of Dupuis (2023), Shen et al. (2023), and Li et al. (2023) emphasize privacy amplification, while Lizama-Perez (2023) concentrates on error correction. The general vulnerabilities of practical quantum channel implementations have also been explored, as evidenced by the studies of Sun and Huang (2022) and Reutov et al. (2023). However, there is a noticeable lack of comprehensive security analyses of QKD key distillation implementations in the existing literature.

This paper addresses the considerations that must be taken into account when integrating theoretically secure subroutines of the classical key distillation phase into a QKD system. We propose a model for estimating the robustness of the key distillation phase, factoring in the imperfect realization of a quantum channel. We believe this contribution provides valuable insights that bridge the knowledge gap between the scientific community and technology providers, thereby enhancing the practical implementation of QKD systems.

The primary contributions of our paper are twofold. First, we provide a straightforward example model for estimating the security of a complete BB84 key distillation procedure. Second, we present an analysis of how the various components of this process influence the security of the final distilled key. These insights offer a comprehensive understanding of the key distillation process and its impact on the overall security of QKD systems.

The organization of the paper is as follows. In Section 2, we distinguish among the various security aspects of classical post-processing and make references to established QKD standards and tools. Section 3 is dedicated to presenting our findings. Here, we allocate specific subsections to detail each key distillation subroutine, culminating in a comprehensive security analysis. The implications of our analysis and the practicality of our proposed methodology are deliberated in Section 4. Finally, Section 5 encapsulates our conclusions drawn from the research.

2 METHODOLOGY

In this section, we introduce the prevailing standards and tools associated with QKD protocols and provide a synopsis of the BB84 key distillation process. Subsequently, we highlight crucial security perspectives pertinent to key distillation.

The recently introduced ISO/IEC (2023) standard on QKD delineates security requirements for both the quantum and classical phases of QKD, taking into consideration various types of threats. The International Telecommunication Union (ITU) has also issued recommendations concerning various aspects of QKD, such as the resilience of QKD networks (ITU, 2023) and key management (ITU, 2020). Furthermore, the European Telecommunications Standards Institute (ETSI) hosts a working group dedicated to QKD (ETSI, 2024b), which is actively preparing documents on various aspects of QKD, including the security of implementation (ETSI, 2024a).

2.1 General Overview of BB84 Key Distillation

In the process of key distillation, the classical channel is not intended to be encrypted. However, it requires authentication to confirm that the key is being shared with the correct party. The initial step, known as sifting, involves selecting only those photons that were prepared and measured in the same basis. As a result, the outcome of the measurement should be deterministic (Bennett and Brassard, 2014). Nonetheless, the sifted bit string is typically not perfect due to channel imperfections causing some bits to flip. To estimate the error rate, a subset of the sifted key bits is revealed and compared with the corresponding bit sequence of the other party.

Error correction is a process that eliminates bit errors induced by channel noise. During the error correction phase, parity checks are transmitted over a public channel. However, these checks inadvertently disclose some information about the secret key. The privacy amplification step is designed to counteract this leakage. It reduces the length of the key bit string to its final size.

2.2 Security Implications of Key Distillation

The security of the key distillation phase is considered to stem from the protocol's capacity to mitigate information leakage during the quantum phase, as well as its ability to limit its own information leakage, thereby ensuring the output of secure key material. Consequently, the combined imperfections of the quantum protocol and classical post-processing algorithms determine the theoretical security level of the final key.

The objective of the key distillation phase is to generate a secret key from the information exchanged during the quantum phase. Therefore, it is necessary to establish certain security assumptions for the quantum phase. This ensures that the key distillation process is sufficiently efficient to counter potential threats, such as eavesdropping.

On the one hand, if our focus is solely on the practical software security of the key distillation phase, we can be assured that the key distillation process itself will not introduce any new security threats. On the other hand, we will lack information about the security level of the keys produced by the protocol, as the security of the keys is dependent on the quantum phase.

The overall security of the final output from the key distillation process hinges on the assumption that the information leakage during both quantum and classical protocols can be accurately quantified. The key distillation protocol itself does not interact with any quantum phenomena. However, if we aim for it to produce practically secure keys, we must be cognizant of potential limitations in the quantum phase. For instance, key information can be obtained by tampering with the phase or wavelength of the signal (Sun et al., 2015; Li et al., 2011), through Trojan horse attacks (Jain et al., 2014), blinding attacks (Weier et al., 2011), photon number splitting attacks (Hong et al., 2016), or by exploiting information about the timing or duration of the signals (Huang et al., 2018; Sun and Huang, 2022). Countermeasures that mitigate or expose some of the attacks directed at the quantum phase have been proposed, for example, by Huang et al. (2018) and Yoshino et al. (2018).

In this paper, we focus solely on the intercept and resend attack model as a potential threat emanating from the quantum channel. This approach is justified, as the intercept and resend attack model can be detected or mitigated using key distillation subroutines. Moreover, a comprehensive discussion of all possible threat types falls outside the scope of this paper.

An intercept and resend attack involves an eavesdropper capturing each individual photon, measuring it, and then retransmitting it to the intended receiver. This type of attack can provide the eavesdropper with substantial information about the secret key, but it also introduces errors into the resulting key bit string. By calculating the probability of this attack, we derive a parameter that represents the likely number of bits eavesdropped during the quantum phase.

In more complex cases, this parameter could be a combination of effects of several attack types, covering all possible leaked bits during the quantum phase.

Each subprocess involved in the key distillation phase influences the security level of the protocol's final output. Therefore, it is crucial to ensure that the final output meets our criteria for a secure encryption key. This implies that merely proving the security of the individual subprocesses is insufficient. Instead, we need to have a comprehensive understanding of the entire process.

3 RESULTS

In this section, we scrutinize our key distillation implementation using the security aspects outlined in the previous section. Each step of the key distillation phase is examined individually: we first present the implementation of the specific subprocess, followed by an analysis of its security.

The key distillation software we implemented receives a bit-sifted noisy key from the quantum layer and executes steps that result in a distilled secure encryption key. Figure 1 provides an overview of the classical key distillation steps. It is important to note that while the messages on the classical channel are authenticated, they are not encrypted.

The key distillation software that has been implemented is connected to the control interface, the key management interface, and the quantum channel interface. The control interface signals when a new key exchange process is about to be initiated. The quantum channel transmits the bit-sifted raw key to the key distillation software, and the final product, a distilled secure encryption key, is then forwarded to the key management interface.

3.1 The Quantum Channel

Our QKD system implements the discrete-variable BB84 protocol (Bennett and Brassard, 2014) with phase modulation and decoy states. The photons are transmitted from the preparation device to the measurement device via an optical fiber.

The error rate of the implemented quantum channel is approximately 10% caused by the optical transmitters and receivers. This value is obtained via empirical tests on our quantum channel.

At both ends of the quantum channel, a laptop is connected to either the preparation device or the measurement device. The Alice side manages the preparation, while the Bob side handles the measurements. Once the quantum phase is complete, the laptops receive the raw bit strings from their respective quantum interfaces.

In our setup, the bit-sifting process is executed directly on the laptops that control the quantum proto-

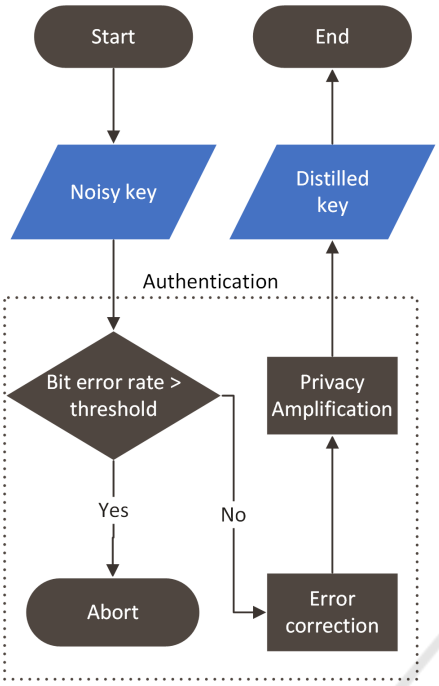

Figure 1: The diagram illustrates the steps involved in the key distillation software. The process commences when the noisy key is received from the quantum channel.

col. Consequently, the input received by our key distillation software is the bit-sifted raw key. Given that the basis choices of Alice and Bob are public, we exclude the bit-sifting phase from our security analysis. Each execution of the quantum phase results in 1000 raw bits that are prepared and measured in the same basis.

3.1.1 Security Analysis: Quantum Channel

The security of the BB84 protocol is predicated on the assumption that each pulse contains a single photon (Bennett and Brassard, 2014). Our quantum layer employs a decoy-state protocol to identify potential photon-number splitting attacks (Mi et al., 2022). Consequently, there are two potential sources of error: natural errors resulting from imperfections in the quantum channel, and errors induced by intercept-and-resend attacks. If an attacker intercepts a photon and subsequently retransmits it, the probability of inducing an error in that specific bit is $\frac{1}{4}$ (Bennett and Brassard, 2014). Given that each intercepted photon is independent of the others, the number of erroneous bits in this scenario follows a binomial distribution with $p = 0.25$.

In this regard, we have three possible alternatives:

1. assuming that every error is caused by eavesdropping,
2. assuming that every error is caused by errors in the quantum channel, or
3. trying to calculate the distribution between these two, relying on our knowledge about the quantum channel.

3.2 Authentication

Authenticated steps are delineated in Figure 1. We have implemented authentication using symmetric message authentication codes (MACs) and a post-quantum key encapsulation method to disseminate the symmetric authentication session key. Specifically, our method employs the CRYSTALS–Kyber key encapsulation algorithm (Bos et al., 2018) to generate a confidential authentication session key for Alice and Bob. We utilized the CRYSTALS–Kyber implementation and certificate management functions of the OpenSSL provider, developed by the Open Quantum Safe Project (Stebila and Mosca, 2016). Following the exchange of authentication keys, Alice and Bob can utilize this session key to generate message authentication codes using a specified MAC algorithm. Our implementation employs the HMAC algorithm (NIST, 2008) to generate secure message authentication codes using the symmetric key.

3.2.1 Security Analysis: Authentication

The security prerequisites for authentication methods are as follows: The final distilled encryption key remains uncompromised even if the authentication method fails after the completion of the QKD. Therefore, it is sufficient to employ a transiently secure authentication method. However, the authentication method in use should maintain its security throughout the key distillation process. In practice this means that to obtain more security the parties that are exchanging secret keys using QKD should renew their authentication session key from time to time.

The implemented authentication method seems like a viable option from the security perspective. The threat posed by Shor's algorithm (1994) excludes classical public key authentication methods and key encapsulation methods from being used in the authentication. The resilience against quantum computing is justified as QKD protocols are considered to be a secure key exchange method in the era of quantum computing.

The key encapsulation algorithm CRYSTALS-Kyber was chosen from the list of algorithms that National Institute of Standards and Technology (NIST) has chosen to be included in the post-quantum public key cryptography standard (Alagic et al., 2022).

HMAC is a secure and efficient keyed hashing algorithm and thus was selected.

CRYSTALS–Kyber is still going through standardization process. First version of the NIST's post-quantum public key cryptograhpy standard is expected in 2024 (NIST, 2017b). Future security of the authentication method is directly affected if vulnerabilities are found during the standardization. This affects also the security promise of QKD key distillation if vulnerable method is used for authentication.

Security level of the authentication method is given by security strength categories that NIST has defined for PQC algorithms in their call for proposals (NIST, 2017a). Standardized algorithms should fulfill the security definitions of these categories. Depending on the chosen parameter size, CRYSTALS–Kyber should correspond to at least the same cryptographic strength as AES128 against both classical and quantum attacks (CRYSTALS Team, 2023). We consider this to be sufficient minimum strength for QKD authentication method. Most security would be achieved using parameters for security level 5, which corresponds to security of AES256 (NIST, 2017a).

Security of any authentication method that uses public key cryptography depends on securely distributing the public keys of participants. In order to trust the security of the key encapsulation method we have to assume that a secure public key infrastructure exists.

3.3 Error Rate Estimation

In the error estimation phase, Alice randomly selects a 10% portion of the raw bit string. Bob compares these bits to his respective bits and calculates how many bits are changed. The error rate estimate is the ratio between flipped bits and all bits. This step is referred to as bit error rate estimation or sometimes as parameter estimation (Wolf, 2021). The sample used for error rate estimation is removed from the raw bit string to ensure it does not become part of the final secure key.

If the error rate is in acceptable limits, which in our case is $\leq 12\%$, Bob will send his error rate estimate to Alice. Otherwise, he only sends a signal to interrupt the distillation process and start the quantum phase again.

3.3.1 Security Analysis: Error Rate Estimation

Error rate estimate of the raw bit string is needed for two purposes:

1. High error rate refers to interference on the channel, which can be caused by intercept and resend attack. In this case, the parties abort the process and try again later. Errors can also occur due to natural factors like photon loss, detection and misalignment errors, background noise, and dark counts (Bennett et al., 1992).
2. Error rate estimate is needed when conducting CASCADE error correction protocol in the error correction phase.

Estimating the error rate consists of several steps: (1) one must choose acceptable threshold λ_{max} for the error rate of the bit string, (2) one must choose the proper size k for the subset of the noisy key that is revealed, (3) one must calculate the error rate Λ_k of this subset and (4) in order to continue the execution of the protocol, the subset must pass a check $\Lambda_k \leq \lambda_{max}$ (Wolf, 2021).

The size k of the subset has to be decided. The reliability of the estimate depends on the relative size of the subset: the smaller the subset, the more unreliable is the estimate (Lu et al., 2017). Let $\overline{X}_k$ denote the fraction of errors in the k-sized subset, with mean $\mu = 0.1$. Each X_i is a Bernoulli trial with $p = 0.1$ and $\overline{X}_k$ concentrates around $\mu = E(\overline{X}_k) = p = 0.1$ as k increases. The size of k can be estimated, e.g., by maximizing the probability $P(0.1 - \varepsilon \leq \overline{X}_k \leq 0.1 + \varepsilon)$.

Generally, the proportion of errors Λ_k in the sample S is $\Lambda_k = \frac{E}{k}$, where E is the hamming weight of the string $S_{Alice} \oplus S_{Bob}$ (i.e., the number of erroneous bits) and k is the size of the sample (Wolf, 2021). For example, if $E = 36$ and $k = 360$ it means that $\Lambda_k = 0.1$.

High estimate of Λ_k requires action: if $\Lambda_k > \lambda_{max}$ the key should be discarded. We define $\lambda_{max} = 0.1 + \varepsilon$, where ε is the maximum allowed deviation from the error rate of the implemented quantum channel.

When the condition $\Lambda_k \leq \lambda_{max}$ is satisfied, we input Λ_k as the estimated quantum bit error rate to the CASCADE error correction protocol.

All error rate estimation methods based on the sample error rate include some amount of uncertainty. However, during the error correction process, Bob will get the error rate Λ_n of the whole bit string as he will have a copy of the noisy key and the error corrected key. At this point, the protocol can still be aborted if $\Lambda_n > \Lambda_k$. Thus, the accuracy of Λ_k will be more of a efficiency than a security aspect: the eavesdropping attempt will be noticed after the error correction, but this approach will lower the secure key generation rate.

Defining an exact upper bound for the tolerable error rate after BB84 protocol is a challenging question. We must rely on the assumption that with a high enough probability, if an eavesdropper has been listening the quantum channel, either the protocol will abort or the eavesdropper will end up with a small

enough amount of information (Gottesman and Lo, 2001).

We treat the quantum channel here as a binary symmetric channel, which means that each bit is flipped with probability p, and the total amount of flipped bits follows a binomial distribution. We strive to set the highest tolerable error rate and raw key size so that it seems highly unlikely to get a key with this error rate in case that the eavesdropper would actually have eavesdropped each bit in our quantum phase.

As stated earlier the natural error rate in our quantum channel is 0.1 and the maximum accepted error rate is 0.12. In our setup, the amount of raw bits coming to the distillation process is 1000. The probability for getting a sample of 1000 bits with error rate of 0.12 or less from a binomial distribution with actual $p = 0.25$ is $\approx 9.445 \cdot 10^{-25}$, which we consider small enough to rely that the eavesdropper has not measured all bits in the quantum phase. On the other hand, probability for getting this error rate or less, i.e., probability of the raw bit string passing the error rate test when there is no eavesdropping activity, is 0.983.

Above we calculated the probability for each photon in the quantum phase being measured by an eavesdropper, so that $p = 0.25$. The other scenario is that only some portion of the quantum phase has been eavesdropped. If we know the error probability of the channel and we can compute the error in the received binary string, it is possible to estimate the number of bits that have been eavesdropped. Let p denote the probability of an error if a bit is eavesdropped and q the probability of an error for a bit that is not eavesdropped. We will assume $q < p$ and that the probability of error due to the channel and the probability induced by eavesdropping are independent of each other. We will also assume that the errors for each bit are independent events.

We denote by n the number of bits received and by k the number of eavesdropped bits. Now the observed error R is

$$R = \frac{k}{n} \times p + \frac{n-k}{n} \times q.$$

Solving for k gives

$$k = \frac{n(R-q)}{p-q}.$$

Now if we have the error probability after eavesdropping at $E_e = 0.25$ and the error for the channel at $E_c = 0.1$, we can compute p and q. Obviously, $q = E_c = 0.1$, but for p we have to note, that both eavesdropping AND channel error can happen independently and thus both possibilities need to be calculated. Thus $p = (1 - E_e) \times E_c + E_e \times (1 - E_c)$, which gives $p = 0.3$ with the values of E_e and E_c given above.

Now we can compute (under the assumptions given above) the value for k once R has been computed from the received binary string. If we take $R = 0.12$ as suggested by the earlier analysis and $n = 1000$ we get $k = 100$, which would mean that 100 bits were eavesdropped. Of course, this being a probabilistic system, this level of accuracy and certainty is not necessarily warranted. However, this gives some indication on the likeliest number of eavesdropped bits, which can then be adjusted with some error margin to make a good guess on the amount of eavesdropped bits.

3.4 Error Correction

For error correction, we utilized the Cascade-Python library (Rijsman Revision, 2020), a Python-based implementation of the CASCADE protocol (Brassard and Salvail, 1994). The CASCADE protocol is based on iterative parity checks: the party with the noisy version of the key shuffles the key bits and requests parity information for these blocks, while the party with the original version of the key provides the parity information. This process is repeated until all bit errors are corrected with a high probability. CASCADE is a probabilistic procedure, meaning it does guarantee the removal of 100% of the bit errors each time. However, based on our software tests, CASCADE performed very well when removing bit errors with an error rate of 0.1.

3.4.1 Security Analysis: Error Correction

The primary security aspect related to the error correction phase is our ability to quantify the amount of information leakage during this phase.

Due to the probabilistic nature of CASCADE, the actual amount of information leakage is only known after the protocol has concluded. In practice, a response to a single parity query leaks one bit of information (Martinez-Mateo et al., 2015; Mehic et al., 2020). Our error correction software tracks the number of leaked bits by counting the responses to parity queries.

After removing the 10% error rate estimation subset from the 1000 raw key bits, we are left with 900 bits in the error correction phase. Correcting errors from a 900-bit raw key with CASCADE leaks approximately 513 bits (averaged over 100 test runs using our software). This leaves us with 387 secret bits after error correction, assuming that the quantum phase has not been eavesdropped on.

3.5 Privacy Amplification

In our implementation, we conduct privacy amplification by multiplication with a Toeplitz matrix. We construct an $n \times m$ sized Toeplitz matrix (where n is the number of final key bits and m is the number of bits in the error corrected bit string) from a random seed and multiply the raw bit string with this matrix. The resulting bit string then becomes the final secure key. Due to the cyclic nature of the Toeplitz matrix, the length of the seed must be $n+m-1$.

3.5.1 Security Analysis: Privacy Amplification

The purpose of privacy amplification is to mitigate the information leaked during quantum or classical protocols. The main security aspects of the privacy amplification steps are: the seed of the hash function is selected randomly (Bennett et al., 1995; ISO/IEC, 2023); the length of the secure key after privacy amplification is at most $n-t$ (where $n =$ amount of all bits and $t =$ amount of bits known to an eavesdropper) (Bennett et al., 1995); and following from the previous point, we must have a good enough approximation about the maximum number of bits leaked to a potential eavesdropper. In practice, we will need to have some extra raw bits to cover situations where more than the average number of bits are leaked.

For choosing the seed of the Toeplitz matrix, we have used the function `os.urandom` in the Python standard library (Python Software Foundation, 2023). According to the documentation, this function should provide enough entropy for cryptographic purposes, so the randomness aspect should be satisfied. Bit leaks in the quantum phase and the error correction phase are addressed in the previous sections. The final point, the length of the resulting bit string, is addressed in the next section.

3.6 The Security Level of the Final Key

Let us denote the number of raw bits that enter the key distillation phase as X. Furthermore, let us denote the size of the error rate estimation subset as a, the amount of leaked information in the error correction phase as b, and the maximum number of potentially eavesdropped bits as c. After the privacy amplification step, as claimed by Bennett et al. (1995), the length of the distilled key should be at most:

$$\delta = X - a - b - c. \quad (1)$$

Earlier, we calculated that correcting errors from a bit string consisting of 900 bits leaks approximately 513 bits, which leaves us with 387 secret bits, assuming that the quantum phase has not been eavesdropped on. We also calculated that when using the maximum tolerable error rate of 0.12, the estimated number of eavesdropped bits in the quantum phase would be 100.

In our implementation, the final secure key was used as a 256-bit AES key (Dworkin, 2023). A length of 256 bits was chosen because it is expected to offer a security level of 128 bits in the era of computationally relevant quantum computers, taking into account the algorithm by Grover (1996), which achieves a quadratic speedup in unstructured search.

In our experiments, we obtained the following values for the maximum tolerable error rate: $X = 1000$, $a = 100$, $b = 513$ and $c = 100$. Now, following Equation 1:

$$\delta = 1000 - 100 - 513 - 100 = 287.$$

Thus, our key distillation process can distill 287 secure bits for every 1000 raw bits from the quantum layer. Because $287 > 256$, we have obtained enough secure bits to generate a 256-bit secret key. The excess bits serve as a security parameter which should cover unexpected bit leaks in the quantum or classical phase.

4 DISCUSSION

The conventional method of measuring the security of encryption algorithms is based on key length; different key lengths offer varying levels of security, typically such that more key bits equate to increased security against attacks. Consequently, the length of the distilled encryption key impacts the level of security it can provide during encryption.

Another factor to consider is the potential for eavesdropping during the key-establishment process, which could result in a potential attacker gaining some knowledge about the key. In our paper, we focus on the intercept and resend attack, estimate its probability using statistical means, and collect its probable effects into a single parameter when defining the security level of the key.

However, numerous other attacks may threaten the security of the quantum phase. Thus, in practical implementations, this parameter must be defined more comprehensively, taking into account all possible threats that have not been mitigated with effective countermeasures. Defining such a parameter, which collects information about the quantum phase as a whole, is a topic for further research. Our paper provides an example of defining and using such a parameter in conjunction with other parameters derived from the classical phase.

Measuring the exact security level of the produced key might be impossible because quantum protocols, such as BB84 used here, are not deterministic in nature but include some random aspects. For instance, we can calculate the average number of pulses that may be eavesdropped given the error rate of the bit string, but this is only an estimate and the actual number may differ. Thus, the only viable option, in the case of imperfect devices, appears to be to decide on a maximum number of eavesdropped bits that can be tolerated. Evidently, the more secure we want to be, the more raw bits are needed and the more time-consuming the key establishment phase will be.

Another important factor is the extent to which the security countermeasures during key distillation affect the key generation rate of the protocol. The security of the distilled keys can always be enhanced by demanding lower error rates and using more raw bits, but when implemented in real-life use cases, a key distillation process must meet certain performance requirements, and thus excessive security measures could render it practically useless.

5 CONCLUSIONS

We have conducted a comprehensive security analysis of a key distillation phase of a practical QKD based on the BB84 protocol. We have assumed that certain security limitations, such as the probability of a specific number of photons being eavesdropped, result from the protocol's quantum phase, and we have acknowledged these limitations in our analysis. To simplify the analysis, we have considered only the intercept and resend attack as a possible source of information leakage, although we note that many other attacks threaten practical QKD implementations.

With the help of existing standards and literature, we have examined every subroutine in the key distillation phase – authentication, error rate estimation, error correction, privacy amplification – and analyzed their effects on the security level of the final key.

Our analysis contributes to the collection of general frameworks that can be used to evaluate QKD key distillation implementations.

The security level of the final key is influenced by several parts of the system, and a single unambiguous measure for this security level should not be considered sufficient. Instead, due to the non-deterministic nature of the quantum phase, we propose that the security level should be considered as a combination of the bit length of the key and the probability that a certain portion of these bits is eavesdropped. The security level can be enhanced either by increasing the bit length or by adding safety procedures to the key distillation phase, such as using more raw bits to produce the final key. As the final use case of the secret key is left open, the final computational strength of the key must be determined separately for each case.

ACKNOWLEDGEMENTS

This work has been supported by the National Quantum Communication Infrastructure in Finland (NaQCI.fi) project (Grant Agreement 101091479), which is part of the European Union's The European Quantum Communication Infrastructure (EuroQCI) initiative.

REFERENCES

Alagic, G., Apon, D., Cooper, D., Dang, Q., Dang, T., Kelsey, J., Lichtinger, J., Miller, C., Moody, D., Peralta, R., et al. (2022). Status report on the third round of the NIST post-quantum cryptography standardization process. *US Department of Commerce, NIST*.

Bennett, C. and Brassard, G. (2014). Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, 560:7–11. Theoretical Aspects of Quantum Cryptography – celebrating 30 years of BB84.

Bennett, C. H., Bessette, F., Brassard, G., Salvail, L., and Smolin, J. (1992). Experimental quantum cryptography. *Journal of cryptology*, 5:3–28.

Bennett, C. H., Brassard, G., Crépeau, C., and Maurer, U. M. (1995). Generalized privacy amplification. *IEEE Transactions on Information theory*, 41(6):1915–1923.

Bos, J., Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schanck, J. M., Schwabe, P., Seiler, G., and Stehlé, D. (2018). CRYSTALS-Kyber: a CCA-secure module-lattice-based KEM. In *2018 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 353–367. IEEE.

Brassard, G. and Salvail, L. (1994). Secret-key reconciliation by public discussion. In Helleseth, T., editor, *Advances in Cryptology — EUROCRYPT '93*, pages 410–423, Berlin, Heidelberg. Springer Berlin Heidelberg.

CRYSTALS Team (2023). CRYSTALS–Kyber website. https://pq-crystals.org/kyber/. Last accessed 14.2.2024.

Dupuis, F. (2023). Privacy Amplification and Decoupling Without Smoothing. *IEEE Transactions on Information Theory*, 69(12):7784–7792.

Dworkin, M. J. (2023). Advanced Encryption Standard (AES). https://doi.org/10.6028/NIST.FIPS.197-upd1.

ETSI (2024a). ETSI GS QKD 016 V2.1.1 (2024-01).

ETSI (2024b). Industry Specification Group (ISG) on Quantum Key Distribution (QKD). https://www.etsi.org/committee/qkd. Last Accessed 25.1.2024.

Gottesman, D. and Lo, H.-K. (2001). Proof of security of quantum key distribution with two-way classical communications. *IEEE Trans. Inf. Theory*, 49:457–475.

Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219.

Hong, K., Foong, O.-M., and Low, T. (2016). Challenges in quantum key distribution: A review. In *ICINS '16: Proceedings of the 4th International Conference on Information and Network Security*, pages 29–33.

Huang, A., Sun, S.-H., Liu, Z., and Makarov, V. (2018). Quantum key distribution with distinguishable decoy states. *Phys. Rev. A*, 98:012330.

ISO/IEC (2023). ISO/IEC 23837-1:2023 - Information security — Security requirements, test and evaluation methods for quantum key distribution — Part 1: Requirements.

ITU (2020). ITU-T Recommendations/ ITU-T Y.3803 (12/2020). Last Accessed 25.1.2024.

ITU (2023). ITU-T Recommendations/ ITU-T Y.3815 (09/2023). Last Accessed 25.1.2024.

Jain, N., Anisimova, E., Khan, I., Makarov, V., Marquardt, C., and Leuchs, G. (2014). Trojan-horse attacks threaten the security of practical quantum cryptography. *New Journal of Physics*, 16(12):123030.

Li, H.-W., Wang, S., Huang, J.-Z., Chen, W., Yin, Z.-Q., Li, F.-Y., Zhou, Z., Liu, D., Zhang, Y., Guo, G.-C., Bao, W.-S., and Han, Z.-F. (2011). Attacking a practical quantum-key-distribution system with wavelength-dependent beam-splitter and multiwavelength sources. *Phys. Rev. A*, 84:062308.

Li, K., Yao, Y., and Hayashi, M. (2023). Tight exponential analysis for smoothing the max-relative entropy and for quantum privacy amplification. *IEEE Transactions on Information Theory*, 69(3):1680–1694.

Lizama-Perez, L. (2023). Reverse reconciliation for optimal error correction in quantum key distribution. *Symmetry*, 15(3).

Lu, Z., Shi, J.-H., and Li, F.-G. (2017). Error rate estimation in quantum key distribution with finite resources. *Communications in Theoretical Physics*, 67(4):360.

Martinez-Mateo, J., Pacher, C., Peev, M., Ciurana, A., and Martin, V. (2015). Demystifying the information reconciliation protocol cascade. *Quantum Info. Comput.*, 15:453–477.

Mehic, M., Niemiec, M., Siljak, H., and Voznak, M. (2020). Error reconciliation in quantum key distribution protocols. In Ulidowski, I., Lanese, I., Schultz, U. P., and Ferreira, C., editors, *Reversible Computation: Extending Horizons of Computing: Selected Results of the COST Action IC1405*, pages 222–236, Cham. Springer International Publishing.

Mi, S., Dong, S., Hou, Q., Wang, J., Yu, Y., Wei, Z., and Zhang, Z. (2022). Joint photon-number splitting attack on semi-quantum key distribution. *Frontiers in Physics*, 10.

Moriarty, K., Kaliski, B., Jonsson, J., and Rusch, A. (2016). PKCS #1: RSA Cryptography Specifications Version 2.2. RFC 8017.

Nir, Y., Josefsson, S., and Pégourié-Gonnard, M. (2018). Elliptic Curve Cryptography (ECC) Cipher Suites for Transport Layer Security (TLS) Versions 1.2 and Earlier. RFC 8422.

NIST (2008). The Keyed-Hash Message Authentication code (HMAC). *Federal Information Processing Standards Publication 198-1*.

NIST (2017a). Post-quantum cryptography: Call for proposals. https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf. Last accessed 14.2.2024.

NIST (2017b). Post-quantum cryptography: Workshops and timeline. https://csrc.nist.gov/Projects/post-quantum-cryptography/workshops-and-timeline. Last accessed 14.2.2024.

Python Software Foundation (2023). os, Miscellaneous operating system interfaces. https://docs.python.org/3/library/os.html. Last accessed 14.2.2024.

Reutov, A., Tayduganov, A., Mayboroda, V., and Fat'yanov, O. (2023). Security of the decoy-state BB84 protocol with imperfect state preparation. *Entropy*, 25(11):1556.

Rijsman Revision, B. (2020). The Cascade information reconciliation protocol. https://cascade-python.readthedocs.io/en/latest/protocol.html. Last accessed 14.2.2024.

Shen, Y.-C., Gao, L., and Cheng, H.-C. (2023). Privacy amplification against quantum side information via regular random binning. In *2023 59th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1–8.

Shor, P. W. (1994). Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th annual symposium on foundations of computer science*, pages 124–134.

Stebila, D. and Mosca, M. (2016). Post-quantum key exchange for the internet and the open quantum safe project. In *International Conference on Selected Areas in Cryptography*, pages 14–37. Springer. https://openquantumsafe.org/.

Sun, S. and Huang, A. (2022). A review of security evaluation of practical quantum key distribution system. *Entropy*, 24(2).

Sun, S.-H., Xu, F., Jiang, M.-S., Ma, X.-C., Lo, H.-K., and Liang, L.-M. (2015). Effect of source tampering in the security of quantum cryptography. *Phys. Rev. A*, 92:022304.

Weier, H., Krauss, H., Rau, M., Fürst, M., Nauerth, S., and Weinfurter, H. (2011). Quantum eavesdropping without interception: an attack exploiting the dead time of single-photon detectors. *New Journal of Physics*, 13(7):073024.

Wolf, R. (2021). *Quantum Key Distribution - An Introduction with Exercises*, volume 988 of *Lecture Notes in Physics*. Springer.

Yoshino, K.-i., Fujiwara, M., Nakata, K., Sumiya, T., Sasaki, T., Takeoka, M., Sasaki, M., Tajima, A., Koashi, M., and Tomita, A. (2018). Quantum key distribution with an efficient countermeasure against correlated intensity fluctuations in optical pulses. *npj Quantum Information*, 4(1).