IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received August 22, 2019, accepted September 12, 2019, date of publication September 24, 2019, date of current version October 9, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2943478

A Key Verification Protocol for Quantum Key Distribution

GUNES KARABULUT KURT[1], (Senior Member, IEEE), ENVER OZDEMIR[2], (Member, IEEE), NESLIHAN AYSEN OZKIRISCI[3], AND OZAN ALP TOPAL[1], (Student Member, IEEE)
[1]Department of Electronics and Communication Engineering, Istanbul Technical University, 34469 Istanbul, Turkey
[2]Informatics Institute, Istanbul Technical University, 34469 Istanbul, Turkey
[3]Department of Mathematics, Yildiz Technical University, 34220 Istanbul, Turkey

Corresponding author: Ozan Alp Topal (topalo@itu.edu.tr)

ABSTRACT Sharing a secret key between two physically separated nodes, Alice and Bob, is possible through the use of quantum key distribution (QKD) techniques. In the presence of an eavesdropper, Alice's key may not be identical with Bob's key, due to the characteristics of a quantum channel. To obtain identical keys at Alice and Bob, we propose a block-based key verification protocol that relies on Newton's polynomial interpolation. As the nodes solely share random numbers and indices of the removed blocks, no information is revealed about the secret message, at a cost of higher computational complexity. The error propagation through the key verification process is prevented by the characteristics of the proposed approach.

INDEX TERMS Key verification, polynomial interpolation, quantum key distribution, secret sharing.

I. INTRODUCTION

Quantum key distribution (QKD) enables sharing a secret message between two distant nodes, Alice and Bob. A secret message can be used as a cryptographic key to encrypt messages that are being transmitted over an insecure channel. Unlike the well-known key sharing algorithms [1], the security in QKD is established by the laws of quantum physics [2], [3]. The first QKD protocol is BB84, developed by Bennett [4]. A typical QKD protocol consists of two parts: a key agreement scheme and a key verification process. In the key agreement part of the protocol, BB84 encodes the information into randomly selected non-orthogonal quantum states, referred to as qubits. As Bennett and Brassard suggest, these qubits are commonly generated by the light polarization and are shared through fiber networks or free space optical links [5]–[7]. The uncertainty principle and the no-cloning principle guarantee that any eavesdropping attempt causes disruption of the quantum signal with a high probability and may introduce errors at the receiver node [2].

Following this work, QKD has expanded into an active area of research, both in theoretical and practical aspects. New key agreement protocols have been proposed [8]–[10], and their security proofs have been obtained [11]. Yet, BB84 protocol remains the most popular protocol, used within practical QKD systems.

The associate editor coordinating the review of this manuscript and approving it for publication was Tai-hoon Kim.

Assuming the presence of an eavesdropper (referred to as Eve) and a noisy communication channel in the key agreement procedure, the secret keys obtained by Alice and Bob may not be identical. Although a key verification process is carried out in the original paper of BB84, the formal problem statement and the associated key verification procedure are defined in the work of Brassard and Salvail [12]. While proposing a reconciliation protocol to verify the secret messages, the protocol causes some information leakage to the eavesdropper. The aim of this verification process is to correct possible errors in the distributed secret key with a minimum information leakage. In accordance with this aim, Brassard and Salvail present a reconciliation algorithm, which is later named as Cascade. Over the decades, some improvements are introduced to Cascade algorithm, but its main disadvantage, the communication load, is still an open issue [13], [14].

In 2003, the Winnow protocol is proposed for key reconciliation by Buttler *et al.* [15]. The performance of the Winnow protocol largely depends on the number of erroneous bits and their uniformity. As the main difference from Cascade, the Winnow protocol may introduce additional errors, if the error distribution is not uniform. However, the Winnow algorithm can decrease the number of transmissions between Alice and Bob. More recently, key reconciliation is studied by adapting the error correcting codes [16]–[18]. Error correcting codes are originated for reversing the disruptive effects caused by the communication channel. While the computational complexity varies according to the selected code,

This work is licensed under a Creative Commons Attribution 4.0 License. For more information, see http://creativecommons.org/licenses/by/4.0/

error propagation and information leakage are still open issues in these code-based approaches [19], [20]. The recent research activities about key reconciliation are discussed in Section II.C.

In this paper, we propose a key verification protocol with zero information leakage. The fundamental idea behind our key verification protocol depends on polynomial interpolation as Shamir's (k, n) thresholding scheme, where the protocol generates random y-axis values in a finite field and selects a certain number of key bits as x-axis values, then interpolates them to obtain a polynomial [21]. The rest of the y-axis values are determined by applying the remaining key bits to the constructed polynomial, which are utilized to compare the x-axis values. By sharing only the y-axis values of the polynomial, Alice and Bob detect erroneous bits and remove them to obtain identical secret keys. The shared information can not be used by the eavesdropper unless the eavesdropper knows a predetermined number of bit-streams through the key bits, which is limited by the nature of the QKD system [2]. Without revealing any useful information about the key bits to the eavesdropper, we aim to overcome the main paradigm of the key reconciliation protocols that some information regarding key is compromised in order to find and reconcile the erroneous bits.

The main contributions of this paper can be listed as below;

- We propose a key verification protocol for QKD, where Alice and Bob aim to eliminate the erroneous bits from the secret key obtained at Bob instead of correcting them. This paradigm difference in the verification protocol reduces the revealed information during the key verification to zero.
- We show that Alice and Bob obtain identical secret keys after the proposed key verification protocol. The error propagation through the key verification process is prevented by the inherent characteristics of the proposed approach. Therefore, privacy amplification becomes unnecessary for the QKD procedure.
- We eliminate all erroneous bits in the secret key, we show that our protocol also reduces the revealed information during the key agreement part to zero.

One drawback of our algorithm is the increased computational complexity in comparison to state-of-the-art key verification algorithms.

The remainder of this paper is structured as follows. In the following section, we give the foundations on polynomial interpolation and describe the QKD system model. In Section III, the proposed key verification algorithm is described. In Section IV, we provide the analysis of our algorithm. Finally, the concluding remarks are drawn and the future work is presented in Section V.

II. BACKGROUND AND RELATED WORK

A. POLYNOMIAL INTERPOLATION

We first present the basic notions that will be employed in the key agreement algorithm. An approximation method, polynomial interpolation, has been used for many problems in the computational sciences and cryptography [21]. Let $\mathbb{F}_q$ denote a finite field with q number of elements where q is a prime power. A polynomial $p(x)$ over $\mathbb{F}_q$ means that it has all coefficients in $\mathbb{F}_q$. Let the data points $(x_0, y_0), (x_1, y_1), \ldots, (x_r, y_r)$ be given such that $x_i, y_i \in \mathbb{F}_q$ and $x_i \neq x_j$ for $i \neq j$. The following theorem has been known since Newton and has been exploited for many different purposes [22].

Theorem 1: For the points $(x_0, y_0), \ldots, (x_r, y_r)$, there is a unique polynomial $p_r(x)$ of degree at most r such that $p_r(x_i) = y_i$ where $0 \leq i \leq r$ (Chapter 4, Theorem I of [22]).

The applications of polynomial interpolation is mainly due to the following corollary.

Corollary 2: Let $p_r(x)$ be a polynomial of degree r. Interpolating any $r + 1$ or more points on the graph of $y = p_r(x)$ results in $p_r(x)$ (Chapter 4, Theorem I of [22]).

Let $(x_0, y_0), \ldots, (x_r, y_r)$ be as above. In Newton's interpolation method, a polynomial of a degree r

$$p_r(x) = c_0 + \sum_{i=1}^{r} c_i \prod_{j=0}^{i-1} (x - x_j)$$

is constructed and the coefficients c_i sequentially obtained by $p_i(x_i) = y_i$ for $i = 1, 2, \ldots, r$. Note that, $p_r(x)$ is identical to $p(x)$.

A significant point for the polynomial interpolation is that at least $r + 1$ points must be known on the graph of the polynomial $p(x)$ to construct it. It is infeasible to obtain $p(x)$ if less than $r + 1$ points are known [21]. Note that the degree of $p(x)$ is r, and the finite field $\mathbb{F}_q$ is assumed to be large enough.

Polynomial interpolation was previously used for set reconciliation purpose [23]. In set reconciliation, each node has an individual set that might have different elements from other sets. Alice and Bob generate a polynomial from their set elements and share randomly selected reference points on the polynomial. Then, both nodes regenerate a polynomial by its roots in order to find both the erroneous and missing elements. In quantum key verification, the erroneous elements are located at Bob's side. Hence, Bob should figure out the erroneous key bits with help from Alice.

B. QKD SYSTEM MODEL

Figure 1 shows a QKD system consisting of two links: (i) a quantum channel for the key agreement process and (ii) an authenticated public communication channel for the key distillation process. In the key agreement process, Alice transforms the generated key bit string K_A of length $2L$ into qubits and shares them via the quantum channel. Bob measures these qubits and maps measurement results into his key bit string K_B of length $2L$. Due to the characteristics of the quantum channel and possible eavesdropping activity, Bob does not know the accuracy of his measurements and needs to estimate the disparities between the K_A and K_B. Key distillation is a post processing step that is used to obtain identical secret keys at Alice and Bob. The process

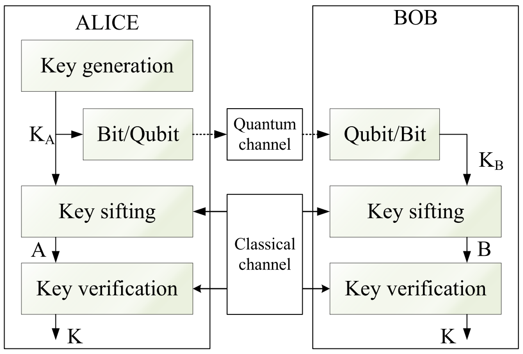


FIGURE 1. QKD system model.

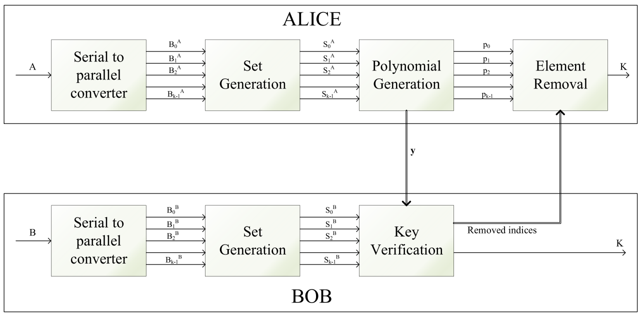


FIGURE 2. A block diagram of the proposed key verification protocol that takes place in the key verification block of Figure 1.

starts with key sifting. In key sifting, Alice and Bob share half of the randomly selected bits and estimate quantum bit error rate (QBER) denoted by ϵ. If QBER is above the key verification threshold, Γ, an error is reported and no key is constructed. If QBER is less than Γ, Alice and Bob discard only the shared bits in the key sifting process in order to eliminate any revealed information to Eve.

Let us denote the remaining bit string at Alice with A and the remaining bit string at Bob with B. Following a similar notation to [12], we denote the random variables by capital letters, the entropy of a random variable U with $H(U)$ and the conditional entropy of U and V is denoted by $H(U|V)$, if each bit from string A is randomly and independently selected,

$$H(A) = |A|,$$

and the conditional entropy is

$$H(A|B) = L \times H(\epsilon),$$

where $|A| = |B| = L$. At the end of the key verification, we define the proposed verification protocol with $V^{\epsilon}(A, B) = [\mathcal{S}, Y]$, where the final secret string $\mathcal{S}$ is produced from the correlated strings A and B by exchanging information string Y. The key verification protocol is detailed in the following section.

C. RELATED WORK

A detailed comparison of the existing information reconciliation schemes can be found in [24]. More recently, research efforts in the key reconciliation focus on designing error correction codes in a more efficient configuration. In [25], the author proposes discretized Gaussian modulation based continuous variable QKD to achieve a high reconciliation efficiency with a low complexity. The authors of [26] propose the usage of the Raptor codes for the first time in the QKD literature to obtain a better performance in the long distance QKD links. In [27], the authors present a polar coding structure for the key reconciliation and compare the efficiency and the complexity of their method with an LDPC based structure. LDPC based key reconciliation schemes have been compared and a symmetric blind information reconciliation method for QKD has been proposed in [28]. In [29], the authors propose a concatenated method for information reconciliation in QKD systems. The authors claim that their method outperforms other one-way information reconciliation schemes in terms of reduced computational requirements and communication delay. One of the main limitations of the aforementioned reconciliation schemes is the information leakage that is lower bounded by the Slepian-Wolf limit [30]. These methods do not guarantee identical secret keys after reconciliation and hence may lead to error propagation [24]. As an important difference from the existing information reconciliation literature, the proposed key verification algorithm does not leak any information during the process and guarantees identical secret keys at the end of the process, with a cost of an increased complexity.

III. KEY VERIFICATION PROTOCOL

The inputs of the key verification protocol are the sifted key bits of Alice and Bob, A and B, each of length L. The key verification process is applied to each block individually and the quantum channel between Alice and Bob is assumed to be a binary symmetric channel (BSC) with an error probability, ε [16]. As previously mentioned, any communication between Alice and Bob is carried out through a classical authenticated channel, which is also assumed to may be intercepted by an eavesdropper.

The proposed key verification protocol is depicted in Figure 2. The protocol starts by determining the parameters to be used in the following steps. Then, Alice divides her sifted key into k blocks using the serial to parallel converter block and transforms bit chunks of length n into decimal numbers in each block by using the set generation algorithm. At the following step, she determines whether there are sufficient number of bits can be used by the key verification process. Here κ denotes the minimum number of bits required to complete key verification. If sufficient number of bits are present, Alice continues with the polynomial generation algorithm and sends the y-axis values of the polynomial of a degree r to Bob. Note that, r is pre-shared between Alice and Bob. Based on y-axis values, Bob verifies his bit chunks and deletes unverified chunks, using the key verification block. In order to obtain identical keys, Bob shares the deleted block indices with Alice and then Alice removes the same blocks.

Algorithm 1: Set Generation Algorithm

```
    Input: $n, B_j^v$ for $j = 0, 1, \ldots, k-1$, $v \in \{A, B\}$
    Output: $S_j^v = \{a_{0,j}, a_{1,j}, \ldots, a_{s-1,j}\}$ from $B_j^v$
 1: for $j = 0 : k-1$ do
 2:     for $i = 0 : s-1$ do
 3:         $C_{i,j}^v = B_j^v(ni : n(i+1)-1)$ ;
 4:         $a_{i,j} = \texttt{Dec}(C_{i,j}^v)$ ;
 5:         $S_j^v(i) = a_{i,j}$
 6:     end
 7:     if $\mathcal{C}\{S_j^v\} < s$ then
 8:         send $j$ to other node ;
 9:         delete $S_j^v$;
10:     else
11:         save $S_j^v$;
12:     end
13: end
```

Algorithm 2: Polynomial Generation Algorithm

```
    Input: $S_j^A = \{a_{0,j}, \ldots, a_{s-1,j}\}$ for
           $j = \{0, 1, \ldots, k-1\}$, $r$
    Output: $y$, $p$ (Polynomial)
 1: for $j = 0 : k-1$ do
 2:     $x_A^j$ = randomly sample $r+1$ elements from $S_j^A$
 3:     $y^j$ = generate $r+1$ random numbers
 4:     $p = \texttt{Polynomial-Interpolate}(x_A^j, y^j)$;
 5:     for $i = r+1 : s-1$ do
 6:         $y^j(i) = p(S_j^A(i))$
 7:     end
 8: end
 9: $y = \{y^0, y^1, \ldots, y^{k-1}\}$
10: save $y$
```

The details of the algorithms of the proposed protocol are presented on the remainder of this section.

A. SET GENERATION

The proposed algorithm is summarized in Algorithm 1. $\mathcal{C}\{S\}$ denotes the cardinality of the set S. Let B_j^A be the j^{th} block with length m at Alice and B_j^B be the j^{th} block at Bob, where $j \in \{0, 1, \ldots, k-1\}$. The algorithm first splits the blocks B_j^A and B_j^B into chunks $C_{i,j}^A$ and $C_{i,j}^B$ such that each has n number of bits, where $i \in \{0, 1, \ldots, s-1\}$ and $s = m/n$ (Step 3 of Algorithm 1). In order to obtain distinct chunks in a block, s is selected to be less than 2^n. Note that, the size of blocks and chunks should be selected in a way that it is almost infeasible to have two chunks with the same bit strings.

Then, set generation algorithm on the block B_j^A continues by generating decimal numbers, $a_{i,j}$, from the chunks, $C_{i,j}^v$ by using $\texttt{Dec}(\cdot)$ function that converts the bit strings to decimal values (Step 4 of Algorithm 1) and transforming blocks into distinct set elements in $\mathbb{F}_q$ (Step 5 of Algorithm 1). The decimal equivalent of the chunks are referred to as set elements. Following this step, Alice checks whether each chunk in a block is transformed into distinct set elements (Steps 7-12 of Algorithm 1). Basically, we have

$$a_{g,j} \neq a_{h,j} \text{ if } g \neq h \text{ for } \quad 0 \leq g,\ h \leq s-1,$$

where $a_{g,j}$ denotes the g^{th} set element of the j^{th} block. If the number of set elements is less than s, Alice removes the corresponding block (Step 9 of Algorithm 1) and sends the index of the deleted block to Bob (Step 8). Otherwise, Alice saves the block (Step 11 of Algorithm 1) and continues to inspect following blocks (Steps 1-13 of Algorithm 1).

At the end of the set generation, the algorithm outputs ordered sets

$$S_j^A = a_{0,j}a_{1,j}\ldots a_{s-2,j}a_{s-1,j}, \text{ for } \quad j = 0, 1, \ldots, k-1$$

at Alice. In practice, it is more convenient not to use any field extension, therefore it can be assumed that q is itself a prime integer with more than 20 digits. We consider the elements $a_{i,j} \in \mathbb{F}_q$. Note that q should be larger than each $a_{i,j}$ for $i = 0, \ldots, s-1$.

As expressed above, Bob waits for an alert from Alice before starting the set generation. If the remaining number of bits is less than κ, Alice sends an alert to restart the key distribution. Otherwise, Bob generates the set of numbers

$$S_j^B = a'_{0,j}, \ldots, a'_{s-2,j}a'_{s-1,j}, \text{ for } \quad j = 0, 1, \ldots, k-1$$

in $\mathbb{F}_q$ from his received bits of the B_j^B.

B. POLYNOMIAL GENERATION

The polynomial generation algorithm is summarized in Algorithm 2. Let r be an integer, which will later be defined in terms on ε, and let

$$\vartheta : \{0, \ldots, s-1\} \to \{0, \ldots, s-1\}$$

be any permutation map on $\{0, \ldots, s-1\}$. First, Alice randomly selects $r+1$ elements from S_j^A by

$$x_A^j = \{a_{\vartheta(1),j}, \ldots, a_{\vartheta(r+1),j}\}$$

(Step 2 of Algorithm 2). Then, she also randomly selects $r+1$ number of $y_i^j \in \mathbb{F}_q$ (Step 3 of Algorithm 2). Then, she sets up $r+1$ pairs

$$(a_{\vartheta(1),j}, y_{\vartheta(1)}^j), (a_{\vartheta(2),j}, y_{\vartheta(2)}^j), \ldots, (a_{\vartheta(r+1),j}, y_{\vartheta(r+1)}^j).$$

By using $\texttt{Polynomial-Interpolate}$ function, Alice interpolates these points in order to produce a polynomial $p(x)$ of degree r (Step 4 of Algorithm 2). Note that the elements $a_{i,j}$ at Alice are produced by randomly selected bit-strings, and the numbers y_i^j are also randomly selected. Therefore, the $r+1$ pairs on the Euclidean space are random and the probability that the polynomial $p(x)$ is of degree less than r is negligible [21]. The degree problem can be rewritten as given below.

Theorem 3: Let $h(x)$ be a polynomial of degree $r-1$. The probability that a randomly selected point in the Euclidean space is on the graph of $h(x)$ is negligible.

Proof: Let $h(x)$ be a polynomial interpolating some points in the Euclidean space. Let $\mathfrak{a}$ be random point on the space. It is highly unlikely that the point $\mathfrak{a}$ lies on the graph of $h(x)$ [21]. The induction on r justifies the theorem. $\square$

Finally, Alice produces the remaining y vector via

$$p(a_{\vartheta(r+2),j}) = y^j_{\vartheta(r+2)}, \ldots, p(a_{\vartheta(s-1),j}) = y^j_{\vartheta(s-1)},$$

for $j = 0, 1, \ldots, k-1$ (Steps 5-7 of Algorithm 2).

At the end of the polynomial generation algorithm, j^{th} block at Alice's side has the following pairs.

$$\begin{tikzcd}
a_{0,j} \arrow[d, leftrightarrow] & a_{1,j} \arrow[d, leftrightarrow] & \cdots \arrow[d, leftrightarrow] & a_{r,j} \arrow[d, leftrightarrow] & a_{(r+1),j} \arrow[d, leftrightarrow] & \cdots \arrow[d, leftrightarrow] & a_{(s-1),j} \arrow[d, leftrightarrow] \\
y^j_0 & y^j_1 & \cdots & y^j_r & y^j_{r+1} & \cdots & y^j_{s-1}
\end{tikzcd}$$

C. KEY VERIFICATION

Bob generates the set of numbers S^B_j from the block B^B_j by using the set generation algorithm. The key verification algorithm is summarized in Algorithm 3. The steps between 2 and 21 show the key verification process and these steps are repeated for each block individually. First, Bob selects corresponding $r+2$ elements in x^j_B,

$$x^j_{B,tr} = a'_{\chi(1),j}, \ldots, a'_{\chi(r+2),j}$$

where tr denotes the trial index (Step 4 of Algorithm 3). Then, Bob selects the corresponding $r+2$ elements in y^j,

$$y^j_{tr} = y^j_{\chi(1)}, \ldots, y^j_{\chi(r+2)}$$

(Step 5 of Algorithm 3). Bob interpolates the following data set

$$(a'_{\chi(1),j}, y^j_{\chi(1)}), (a'_{\chi(2),j}, y^j_{\chi(2)}), \ldots, (a'_{\chi(r+2),j}, y^j_{\chi(r+2)}),$$

where χ is another permutation function on $\{0, \ldots, s-1\}$ (Step 6 of Algorithm 3). Bob first checks whether the degree of the generated polynomial is equal to r (Step 7 of Algorithm 3). As long as Bob obtains a polynomial $p'(x)$ of degree r then the set elements at Bob

$$a'_{\chi(1),j}, a'_{\chi(2),j}, \ldots, a'_{\chi(r+2),j}$$

should be the same as Alice's elements. In fact, Theorem 3 states that when $p(x^j)$ is a polynomial of degree r interpolating $r+1$ pairs then the probability that a random pair (x, y) on the graph of $p(x)$ is negligible.

Note that once Bob obtains the same polynomial as Alice, he can detect incorrect chunks in a block. For example, for the remaining y^j_c values at the y^j vector, Bob checks if

$$p'(a'_{c,j}) \overset{?}{=} y^j_c.$$

Once the equality holds for some c, it means that $a'_{c,j}$ was received correctly. It should be noted here that when Bob realizes $p(a'_{g,j}) \neq y^j_g$ for some y^j_g, then he concludes $a'_{g,j} \neq a_{g,j}$. This property ensures that all erroneous elements would be detected at Bob's side; thus, errors will not propagate through the key verification process. In addition, Bob may solve

$$p(x) = y^j_g$$

and obtain the correct data points sent by Alice, but the number r, which is the degree of $p(x)$, might be large and so finding a root $p(x)$ might not be feasible. Moreover, the equation $p(x) = y^j_g$ might have more than one roots so it requires additional communication to decide which ones match Alice's elements. Therefore, Bob keeps only $a'_{1,j}, \ldots, a'_{r+2,j}$ and $a'_{i,j}$ with $p(a'_i) = y^j_i$ and removes the other elements in S^B_j (Steps 11-14 of Algorithm 3). Note that, Bob sends the indices of the elements that he removes through key verification process to Alice (Step 15 of Algorithm 3).

Bob may receive at least one erroneous bit for some chunks, such that the number of matching set elements of his with Alice's is less than $r+2$. In this case, Bob will never be able to construct the same polynomial with Alice. Therefore, the algorithm should define a maximum number of trials for the receiver side. Let us denote this number by N_T and its selection is given in the practical considerations section. Whenever Bob tries more than N_T times to find

Algorithm 3: Key Verification Algorithm

```
    Input: $y^j$, $S^B_j = \{a_{0,j}, a_{1,j}, \ldots, a_{s-1,j}\}$ of length $m$,
    $j = \{0, 1, \ldots, k-1\}$, $r$, $N_T$
    Output: deleted $j$ indices
 1: for $j = 0 : k-1$ do
 2:     $tr = 1$;
 3:     while $tr < N_T$ do
 4:         $x^j_{B,tr}$ = randomly sample $r+2$ elements
            from $S^B_j$ ;
 5:         $y^j_{tr}$ = select corresponding $r+2$ elements
            from $y^j$ ;
 6:         $p_{tr}$ =
            Polynomial-Interpolate($x^j_{B,tr}, y^j_{tr}$);
 7:         if $degree(p_{tr}) = r$ then
 8:             $tr = N_T$ ;
 9:             $y^j_b = p_{tr}(S^B_j)$
10:             for $i = r+1 : s-1$ do
11:                 if $y^j_b(i) = y^j(i)$ then
12:                     keep $S^B_{b,j}(i)$
13:                 else
14:                     delete $S^B_{b,j}(i)$
15:                     save $j$
16:                 end
17:             end
18:         else
19:             $tr = tr + 1$ ;
20:         end
21:     end
22: end
```

polynomial $p(x)$ for a block then both Alice and Bob disregard the corresponding block completely in the process of key verification. The process continues until all remaining blocks are processed through the key verification algorithm. An illustrative example of the presented key verification protocol for a single block is given as the below. Note that, same process is repeated for every block.

Example:

- Let $\epsilon = 0.16$, $m = 12$, $n = 3$, $r = 2$ and the field be $\mathbb{R}$. The j^{th} block of the sifted keys are $B_j^A = \{010\,011\,110\,111\,001\}$ and $B_j^B = \{010\,011\,1\mathbf{01}\,111\,001\}$.
- Set generation: Alice forms a set from B_j^A as $S_j^A = \{2, 3, 6, 7, 1\}$ by $(010) \rightarrow 2, (011) \rightarrow 3, (110) \rightarrow 6, (111) \rightarrow 7, (001) \rightarrow 1$.
- Polynomial generation: Alice randomly selects 3 number of y_i^j as $y_j = \{1, 4, 2\}$. Then, she sets a polynomial with degree of 2, where $p_2(2) = 1, p_2(3) = 4, p_2(6) = 2$. The polynomial can be constructed as

$$p_2(x) = 1 + \frac{4-1}{3-2}(x-2) + \frac{\frac{4-1}{3-2} - \frac{2-4}{6-3}}{6-2}(x-2)(x-3)$$
$$= -\frac{11}{12}x^2 + \frac{91}{12}x - \frac{21}{2}.$$

  The corresponding y_i^j' of the remaining a_i^j' is found via $p_2(x = 7) = -7/3, p_2(x = 1) = -23/6$. Then, Alice shares $y_j = \{1, 4, 2, -7/3, -23/6\}$ with Bob.
- Bob forms a set from B_j^B as $S_j^B = \{2, 3, 5, 7, 1\}$ by $(010) \rightarrow 2, (011) \rightarrow 3, (1\mathbf{01}) \rightarrow \mathbf{5}, (111) \rightarrow 7, (001) \rightarrow 1$.
- Key verification: Bob tries to construct a polynomial with a degree of 2 with 4 reference points.

Trial	Selected Points $(a'_{i,j}, y_i^j)$	Formed Polynomial $p'(x)$
1	$\{(2,1); (3,4); (5,2); (7, -\frac{7}{3})\}$	$\frac{5}{24}x^3 - \frac{41}{12}x^2 + \frac{129}{8}x - \frac{81}{4}$
2	$\{(2,1); (3,4); (5,2); (1, -\frac{23}{6})\}$	$-\frac{169}{6}x^3 + \frac{1587}{6}x^2 - \frac{1275}{6}x + \frac{2799}{6}$
3	$\{(2,1); (3,4); (7, -\frac{7}{3}); (1, -\frac{23}{6})\}$	$-\frac{11}{12}x^2 + \frac{91}{12}x - \frac{21}{2}$

  At the 3rd trial, Bob finds the right degree of the polynomial. As stated in Theorem 3, since the degree of the polynomial is 2, Bob concludes $p_2'(x) = -\frac{11}{12}x^2 + \frac{91}{12}x - \frac{21}{2}$. Then, he verifies the remaining set elements by controlling $p_2'(x = 2) \overset{?}{=} 1, p_2'(x = 3) \overset{?}{=} 4, p_2'(x = 5) \overset{?}{=} 2, p_2'(x = 7) \overset{?}{=} -7/3, p_2'(x = 1) \overset{?}{=} -23/6$. Since $p_2'(x = 5) \neq 2$, Bob removes 5 from his set and sends its index i to Alice.
- Alice removes 5 from her set. Final keys in both Alice and Bob become $K = \{010\,011\,111\,001\}$.

In the above example, for the convenience of the operation, real number field is chosen to illustrate polynomial generation. In practice, however, multiplication operation is performed in the finite field $\mathbb{F}_q$, where q is a large prime integer.

IV. ANALYSIS ON THE KEY VERIFICATION PROTOCOL

In this section, analysis related to the presented key verification protocol is given in order to support the comparisons given in Section I.

A. INFORMATION LEAKAGE

Information leakage is the amount of disclosed information (in bits) during the QKD process. Let us denote the amount of total leaked information during QKD with I_{lk} and observed string at Eve with E. Then, the leaked information can be defined as

$$I_{lk} \triangleq I(A; E),$$

where $I(A; E)$ denotes the mutual information of random variables A and E. A QKD system consist of a quantum communication link and a classical public link between Alice and Bob. Since both media are open to interception from Eve, both media can leak information [31]. Then, we can state

$$I_{lk}(\mathcal{S}) = I_{lk}^q(\mathcal{S}) + I_{lk}^p(\mathcal{S}).$$

Here, $I_{lk}^q(\mathcal{S})$ is the amount of leaked information during the quantum key agreement part, where Alice and Bob use the quantum communication link. $I_{lk}^p(\mathcal{S})$ is the amount of leaked information during the key verification part, where Alice and Bob use the classical public link.

During the key agreement process, eavesdropping activity is assumed to introduce errors on the transmitted qubits [32]. Therefore, revealed information at quantum key agreement part is

$$I_{lk}^q(\mathcal{S}) = \epsilon \times n.$$

Here, $\epsilon \times n$ shows the number of erroneous bits in a block for an ϵ QBER and n-bit block. The key distillation medium is assumed to be an authenticated public channel, where any transmitted information during the key verification process is assumed to be obtained from the eavesdropper. During key reconciliation, Alice and Bob aim to correct erroneous bits at Bob by sharing side information about the secret key. Here the information of the final secret is upper bounded by $|\mathcal{S}| \leq H(A) = H(B)$. According to Slepian-Wolf limit [30], the revealed information for the key reconciliation processes with error control coding codes is lower bounded by

$$I_{lk}^p(\mathcal{S}) \geq H(A|B).$$

By intuition, Alice reveals the minimum amount of information if she knows Bob's information about the secret. This assures information leakage during the key reconciliation process as long as $\epsilon > 0$.

Contrary to the key reconciliation schemes, we aim to eliminate all erroneous bits at Bob. Hence, the information of the final secret key is upper bounded by $|\mathcal{S}| \leq I(A, B)$. Eliminating the error correction condition also helps us to change the information leakage limit.

Theorem 4: Revealed information during our proposed key verification process is zero for $\Gamma \leq \frac{r-1}{n}$.

Proof: During the proposed key verification scheme, the only shared information is the y axis values of the generated polynomial, $\mathbf{y}$. Let us denote the mutual information of $\mathbf{y}$ and the secret with $I(\mathcal{S}; Y)$. Then,

$$I_{lk}(\mathcal{S}) \triangleq I(\mathcal{S}; Y) = H(\mathcal{S}) - H(\mathcal{S}|Y).$$

For the selected $\Gamma \leq \frac{r-1}{n}$, Eve is assumed to obtained at most $(r-1)s$ consecutive bits from the Alice's key during the key agreement process. Let us rewrite the uncertainty about the secret at Eve as $H(\mathcal{S}|Y, A_{r-1})$, where A_{r-1} denotes any permutation of $(r-1)s$ number of bits from A and note that from the chain rule

$$H(\mathcal{S}|Y, A_{r-1}) \leq H(\mathcal{S}|Y).$$

As proved in Shamir's (k, n) thresholding scheme [21], $r-1$ revealed element pairs (a_i, y_i), does not reveal any information about polynomial of degree r, where every candidate secret S corresponds to a unique polynomial of degree $r-1$. From the construction of the polynomials, all their probabilities are equal. Thus, if

$$H(\mathcal{S}|Y, A_{r-1}) = H(\mathcal{S}|Y) = H(\mathcal{S})$$

then, $I(\mathcal{S}; Y) = 0$. □

Corollary 5: Revealed information in any QKD scheme using the proposed key verification scheme is zero for $\Gamma \leq \frac{r-1}{n}$.

Proof: Since eavesdropping at quantum channel causes error, we can assume that $I^q_{lk}(\mathcal{S})$ corresponds to erroneous elements of the secret key. Theorem 3 ensures that all erroneous elements (along with some correct elements) are deleted during the key verification algorithm at Bob. Hence, our algorithm guarantees that

$$I^q_{lk}(\mathcal{S}) = 0,$$

and consequently

$$I_{lk}(\mathcal{S}) = 0.$$

□

B. BLOCK VERIFICATION PROBABILITY

Under ε error probability BSC assumption, the exact QBER of an individual block and the distribution of erroneous elements through the chunks can not be determined. The probability of correct chunk and inherently a set element is

$$P(a'_{i,j} = a_{i,j}) = (1-\varepsilon)^n. \tag{IV.1}$$

The probability of obtaining exactly z number of correct chunks in a block j is

$$P_c(z) = P(\mathcal{C}\{S^A_j \cap S^B_j\} = z) = \binom{s}{z}(1-\varepsilon)^{nz}(1-(1-\varepsilon)^n)^{s-z}. \tag{IV.2}$$

Now let's assume that, Bob is not limited with N_T and tries every possible element combination for key verification. In this case, Bob needs to obtain at least $r+2$ correct set elements in order to verify a block. Then, the probability of block verification would be

$$P_c = \sum_{k=r+2}^{s} \binom{s}{k}(1-\varepsilon)^{nk}(1-(1-\varepsilon)^n)^{s-k}. \tag{IV.3}$$

Since Alice and Bob know the estimated QBER and ε at the beginning, the parameters n, r, s could be determined with respect to the requirements.

The following part of this paper shows the importance of the computational power of Bob. The key verification probability will be rewritten by limiting the trial number at Bob.

C. COMPUTATIONAL COMPLEXITY

Alice, in our quantum key verification method, interpolates $r+1$ pairs and obtains a polynomial of degree r for a block. This operation costs $O(r\log(r))$ bit operations. Alice also needs to compute the images of all remaining points under $p(x)$. Note that each block has s elements i.e. chunks. The cost of computing $p(a)$ is $O(r)$ for a single point a. As the number of remaining points is $s-r-1$, $O(r(s-r-1))$ bit operations requires to find the corresponding y-values for each block.

Alice forms the interpolating polynomial with $r+1$ pairs and she broadcast the y coordinates of these pairs. Bob pairs up his $r+2$ data points with broadcast y values and interpolates them. The resulting polynomial will be completely different even if a single data point of Bob does not match with Alice. The data points are formed with using bit strings, therefore a single bit error could cause a distinct polynomial at Bob.

Lemma 6: Let s be the number of elements in the block formed by Alice and $r+1$ be the number of pairs that are employed to construct interpolating polynomial $p(x)$. Assume that only e number of points of Bob do not match to Alice's such that $s-e \geq r+2$. If Bob randomly selects ψ number of data points and interpolates them then the probability that the polynomial is of degree r is

$$\frac{\binom{s-e}{\psi}}{\binom{s}{\psi}}$$

where $r+2 \leq \psi \leq s$.

Proof: There are s chunks which corresponds to a data point in each block on both sides. ψ number of pairs are employed to obtain the polynomial. The polynomial will match with Alice's once all the pairs match on both sides. As there are e disparities between Bob and Alice, selecting the correct ones out of all is

$$\frac{\binom{s-e}{\psi}}{\binom{s}{\psi}}$$

□

Lemma 6 suggests that $\psi \geq r+2$ should be as small as possible, therefore the best option is $\psi = r+2$ and that is why Bob employs $r+2$ pairs to construct polynomials.

D. PRACTICAL CONSIDERATIONS

The experiments on key distribution generally assume the initial key length $L = 10^4$ or $L = 10^5$ and the threshold $\Gamma = 10\%$. As it is indicated in the Section II, if the estimated error rate is above the threshold value ($\varepsilon > \Gamma$), the key verification process will be restarted. If not, according to the error rate and the number of set elements, the degree of the polynomial is determined and the process continues.

The minimum number of key length after verification, κ, is another important constraint. At the end of the key distribution process, the generated key should be durable to brute force attacks from quantum computers. In this case, κ should be longer than 256 bits. If the key verification algorithm deletes more than $L-\kappa$ bits, than the process is stopped and a new distribution is required.

1) THE MAXIMUM NUMBER OF TRIALS (N_T)

N_T is a critical parameter and creates a bottleneck for the running time and the key verification rate. As the number of trials increases, the algorithm takes a longer time but the probability of finding correct elements required for the correct polynomial generation increases.

The probability of selecting $r+2$ correct elements from a block containing z correct elements at the first trial is

$$P_1(x|z) = \frac{\binom{z}{r+2}}{\binom{s}{r+2}}, \tag{IV.4}$$

and the probability of selecting $r+2$ correct elements after l independent trials is

$$P_l(x|z) = (1 - P_1(x|z))^{l-1} P_1(x|z). \tag{IV.5}$$

Eventually, the probability of selecting $r+2$ correct elements after at most N_T trials becomes the following

$$\tilde{P}(x|z) = \sum_{l=1}^{N_T} (1 - P_1(x|z))^{l-1} P_1(x|z). \tag{IV.6}$$

Lemma 7: For block verification, the correct elements in a block is lower bounded by the degree of the polynomial, where $r+2 \leq z$. In this case, the probability of key verification after N_T trials for a block would be lower bounded by

$$P_c \geq \tilde{P}_c = \sum_{z=r+2}^{s} \tilde{P}(x|z) P_c(z). \tag{IV.7}$$

The final form of the lower bound for the block verification after N_T trials is

$$\tilde{P}_c = \sum_{z=r+2}^{s} \sum_{l=1}^{N_T} \left(1 - \frac{\binom{z}{r+2}}{\binom{s}{r+2}}\right)^{l-1} \frac{\binom{z}{r+2}}{\binom{s}{r+2}} \binom{s}{z} \times (1-\varepsilon)^{nz} (1 - (1-\varepsilon)^n)^{s-z}. \tag{IV.8}$$

In line with ε, n, $\tilde{P}_c$ and computational capabilities of Bob, we can select the N_T before the verification started and drop the block after N_T trials.

V. CONCLUDING REMARKS

In this paper, we propose a key verification protocol for quantum key distribution (QKD), where the legitimate sender and the legitimate receiver obtain identical secret keys as the protocol terminates. By solely sharing random numbers and deleted block indices, the revealed information to the eavesdropper is proven to be zero during the key verification process. Also, the protocol guarantees eliminating the revealed information to the eavesdropper during the quantum key agreement process. For future work, we aim to correct erroneous bits at the cost of a higher complexity.

REFERENCES

[1] W. Diffie and M. E. Hellman, "New directions in cryptography," *IEEE Trans. Inf. Theory*, vol. IT-22, no. 6, pp. 644–654, Nov. 1976.

[2] W. K. Wootters and W. H. Zurek, "A single quantum cannot be cloned," *Nature*, vol. 299, pp. 802–803, Oct. 1982.

[3] A. K. Ekert, "Quantum cryptography based on Bell's theorem," *Phys. Rev. Lett.*, vol. 67, no. 6, pp. 661–663, Aug. 1991.

[4] C. H. Bennett, "Quantum cryptography using any two nonorthogonal states," *Phys. Rev. Lett.*, vol. 68, no. 21, pp. 3121–3124, May 1992.

[5] G. Cañas, N. Vera, J. Cariñe, P. González, J. Cardenas, P. W. R. Connolly, A. Przysiezna, E. S. Gómez, M. Figueroa, G. Vallone, P. Villoresi, T. F. da Silva, G. B. Xavier, and G. Lima, "High-dimensional decoy-state quantum key distribution over multicore telecommunication fibers," *Phys. Rev. A, Gen. Phys.*, vol. 96, Aug. 2017, Art. no. 022317.

[6] B. Fröhlich, M. Lucamarini, J. F. Dynes, L. C. Comandar, W. W.-S. Tam, A. Plews, A. W. Sharpe, Z. Yuan, and A. J. Shields, "Long-distance quantum key distribution secure against coherent attacks," *Optica*, vol. 4, no. 1, pp. 163–167, Jan. 2017.

[7] P. V. Trinh and A. T. Pham, "Design and secrecy performance of novel two-way free-space QKD protocol using standard FSO systems," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.

[8] B. Huttner, N. Imoto, N. Gisin, and T. Mor, "Quantum cryptography with coherent states," *Phys. Rev. A, Gen. Phys.*, vol. 51, no. 3, p. 1863, 1995.

[9] D. Bruß, "Optimal eavesdropping in quantum cryptography with six states," *Phys. Rev. Lett.*, vol. 81, no. 14, p. 3018, 1998.

[10] W. Liu, Y.-B. Wang, and W.-Q. Fan, "An novel protocol for the quantum secure multi-party summation based on two-particle bell states," *Int. J. Theor. Phys.*, vol. 56, no. 9, pp. 2783–2791, 2017.

[11] H.-K. Lo and H. F. Chau, "Unconditional security of quantum key distribution over arbitrarily long distances," *Science*, vol. 283, no. 5410, pp. 2050–2056, Mar. 1999.

[12] G. Brassard and L. Salvail, "Secret-key reconciliation by public discussion," in *Advances in Cryptology EUROCRYPT*. 1993, pp. 410–423.

[13] J. Martinez-Mateo, C. Pacher, M. Peev, A. Ciurana, and V. Martin, "Demystifying the information reconciliation protocol cascade," 2014, *arXiv:1407.3257*. [Online]. Available: https://arxiv.org/abs/1407.3257

[14] W. Ma and G. Zeng, "An improvement on 'Cascade' protocol in quantum key distribution," *Acta Sinica Quantum Optica*, vol. 16, no. 4, pp. 271–275, 2010.

[15] W. T. Buttler, S. K. Lamoreaux, J. R. Torgerson, G. H. Nickel, C. H. Donahue, and C. G. Peterson, "Fast, efficient error reconciliation for quantum cryptography," *Phys. Rev. A, Gen. Phys.*, vol. 67, May 2003, Art. no. 052303.

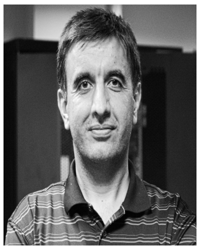
[16] D. Elkouss, A. Leverrier, R. Alleaume, and J. J. Boutros, "Efficient reconciliation protocol for discrete-variable quantum key distribution," in *Proc. IEEE Int. Symp. Inf. Theory*, Jun./Jul. 2009, pp. 1879–1883.

[17] J. M. Renes, D. Sutter, F. Dupuis, and R. Renner, "Efficient quantum polar codes requiring no Preshared entanglement," *IEEE Trans. Inf. Theory*, vol. 61, no. 11, pp. 6395–6414, Nov. 2015.

[18] D. Elkouss, J. Martinez, D. Lancho, and V. Martin, "Rate compatible protocol for information reconciliation: An application to QKD," in *Proc. IEEE Inf. Theory Workshop Inf. Theory (ITW)*, Jan. 2010, pp. 1–5.

[19] J. S. Johnson, M. R. Grimaila, J. W. Humphries, and G. B. Baumgartner, "An analysis of error reconciliation protocols used in Quantum Key Distribution systems," *J. Defense Model. Simul. Appl. Methodol. Technol.*, vol. 12, no. 3, pp. 217–227, 2015.
[20] D. Elkouss, J. Martinez-Mateo, and V. Martin, "Information reconciliation for quantum key distribution," 2010, *arXiv:1007.1616*. [Online]. Available: https://arxiv.org/abs/1007.1616
[21] A. Shamir, "How to share a secret," *Commun. ACM*, vol. 22, no. 11, pp. 612–613, Nov. 1979.
[22] D. Kincaid and W. Cheney, *Numerical Analysis: Mathematics of Scientific Computing*. Pacific Grove, CA, USA: Brooks/Cole Publishing, 1991.
[23] Y. Minsky, A. Trachtenberg, and R. Zippel, "Set reconciliation with nearly optimal communication complexity," *IEEE Trans. Inf. Theory*, vol. 49, no. 9, pp. 2213–2218, Sep. 2003.
[24] C. Huth, R. Guillaume, T. Strohm, P. Duplys, I. A. Samuel, and T. Güneysu, "Information reconciliation schemes in physical-layer security: A survey," *Comput. Netw.*, vol. 109, pp. 84–104, Nov. 2016.
[25] I. B. Djordjevic, "On the discretized Gaussian modulation (DGM)-based continuous variable-QKD," *IEEE Access*, vol. 7, pp. 65342–65346, 2019.
[26] M. B. Asfaw, X.-Q. Jiang, M. Zhang, J. Hou, and W. Duan, "Performance analysis of raptor code for reconciliation in continuous variable quantum key distribution," in *Proc. Int. Conf. Comput. Netw. Commun. (ICNC)*, Feb. 2019, pp. 463–467.
[27] S. Lee, J. Park, and J. Heo, "Improved reconciliation with polar codes in quantum key distribution," 2018, *arXiv:1805.05046*. [Online]. Available: https://arxiv.org/abs/1805.05046
[28] E. O. Kiktenko, A. S. Trushechkin, C. C. W. Lim, Y. V. Kurochkin, and A. K. Fedorov, "Symmetric blind information reconciliation for quantum key distribution," *Phys. Rev. Appl.*, vol. 8, no. 4, Oct. 2017, Art. no. 044017.
[29] L. Yang, H. Dong, and Z. Li, "One-way information reconciliation schemes of quantum key distribution," *Cybersecurity*, vol. 2, no. 1, p. 16, May 2019.
[30] D. Slepian and J. K. Wolf, "Noiseless coding of correlated information sources," *IEEE Trans. Inf. Theory*, vol. IT-19, no. 4, pp. 471–480, Jul. 1973.
[31] V. Scarani, H. Bechmann-Pasquinucci, N. J. Cerf, M. Dušek, N. Lütkenhaus, and M. Peev, "The security of practical quantum key distribution," *Rev. Mod. Phys.*, vol. 81, pp. 1301–1350, Sep. 2009.
[32] J. Martinez-Mateo, D. Elkouss, and V. Martin, "Key reconciliation for high performance quantum key distribution," *Sci. Rep.*, vol. 3, p. 1576, Apr. 2013.

GUNES KARABULUT KURT (S'00–M'06–SM'15) received the B.S. degree (Hons.) in electronics and electrical engineering from Bogazici University, Istanbul, Turkey, in 2000, and the M.A.Sc. and Ph.D. degrees in electrical engineering from the University of Ottawa, ON, Canada, in 2002 and 2006, respectively. From 2000 to 2005, she was a Research Assistant with the CASP Group, University of Ottawa. From 2005 to 2006, she was with TenXc Wireless, Canada. From 2006 to 2008, she was with Edgewater Computer Systems Inc., Canada. From 2008 to 2010, she was with Turkcell Research and Development Applied Research and Technology, Istanbul. Since 2010, she has been with Istanbul Technical University, where she is currently a Professor. Her research interests include wireless network design, physical layer security, and network coding. She is a Marie Curie Fellow.



ENVER OZDEMIR received the Ph.D. degree in mathematics from the University of Maryland College Park, in 2009. He was a member of the Coding Theory and Cryptography Research Group (CCRG), Nanyang Technological University, Singapore, from 2010 to 2014. He is currently an Associate Professor with the Informatics Institute, Istanbul Technical University. His research interests include cryptography, computational number theory, and network security.

NESLIHAN AYSEN OZKIRISCI received the Ph.D. degree in mathematics from Yildiz Technical University, in 2014, where she is currently a Postdoctoral Researcher with the Mathematics Department. Her research interests include commutative algebra, ring and module theory, number theory, and cryptography.

OZAN ALP TOPAL received the B.S. and M.S. degrees in electronics and communication engineering from Istanbul Technical University, Istanbul, Turkey, in 2017 and 2019, respectively. He is currently pursuing the Ph.D. degree in telecommunication engineering. He has been serving as a Research Assistant with the Department of Electronics and Communications Engineering, Istanbul Technical University, since 2018. His research interests include physical layer security and quantum key distribution.