Title	Quantum key distribution with delayed privacy amplification and its application to the security proof of a two-way deterministic protocol
Author(s)	Fung, CHF; Ma, X; Chau, HF; Cai, QY
Citation	Physical Review A - Atomic, Molecular, And Optical Physics, 2012, v. 85 n. 3
Issued Date	2012
URL	http://hdl.handle.net/10722/145929
Rights	Creative Commons: Attribution 3.0 Hong Kong License

Quantum key distribution with delayed privacy amplification and its application to the security proof of a two-way deterministic protocol

Chi-Hang Fred Fung,[1,*] Xiongfeng Ma,[2,†] H. F. Chau,[1,‡] and Qing-yu Cai[3,§]

[1]*Department of Physics and Center of Computational and Theoretical Physics, University of Hong Kong, Pokfulam Road, Hong Kong*

[2]*Center for Quantum Information and Quantum Control, Department of Physics and Department of Electrical & Computer Engineering, University of Toronto, Toronto, Ontario, Canada*

[3]*State Key Laboratory of Magnetics Resonances and Atomic and Molecular Physics, Wuhan Institute of Physics and Mathematics, Chinese Academy of Sciences, Wuhan 430071, People's Republic of China*

(Received 18 November 2011; published 9 March 2012)

Privacy amplification (PA) is an essential postprocessing step in quantum key distribution (QKD) for removing any information an eavesdropper may have on the final secret key. In this paper, we consider delaying PA of the final key after its use in one-time pad encryption and prove its security. We prove that the security and the key generation rate are not affected by delaying PA. Delaying PA has two applications: it serves as a tool for significantly simplifying the security proof of QKD with a two-way quantum channel, and also it is useful in QKD networks with trusted relays. To illustrate the power of the delayed PA idea, we use it to prove the security of a qubit-based two-way deterministic QKD protocol which uses four states and four encoding operations.

DOI: 10.1103/PhysRevA.85.032308 PACS number(s): 03.67.Dd, 03.67.Hk, 03.67.Ac

I. INTRODUCTION

Quantum key distribution (QKD) [1,2] allows two parties, Alice and Bob, to share a secret key by exchanging quantum particles. The final secret key is secure against any eavesdropper, called Eve, with unlimited computational power. Initial security proofs of QKD mostly focus on infinite key size and perfect equipment [3–10]. More recent security proofs take into consideration device imperfection [11–16]; while the effect of finite key size is explicitly considered in Refs. [17–23].

QKD protocols usually involve two postprocessing steps after the quantum state transmission step: error correction (EC) [24] to make sure Alice's key is the same as Bob's and privacy amplification (PA) [25,26] to ensure Eve does not have any nontrivial information on the final secret key. The final secret key generated can then be used in a subsequent cryptographic application such as the one-time pad (OTP) [27,28].

In this paper, we consider running QKD without immediately running EC and PA. Assuming that the OTP will be used as the next step, we delay the application of EC and PA until after the OTP, in effect performing a weakly secure OTP. Delaying EC is trivial and requires no extra attention since errors in the original key simply translate into the same errors in the OTP-encrypted message, and bit errors do not affect the security of the message. On the other hand, delaying PA is nontrivial since normal PA ensures a key becomes secure first before being used, and now we use the insecure key first before making it secure. These two operations do not appear to be commuting, but we will prove that they do when we choose an appropriate PA scheme. By commuting, we mean that delayed PA is secure with the same security level achieved by the same PA function used to make the original raw key secure. In summary, we prove that delaying PA after the OTP does not affect the security and the key generation rate. Delayed PA is the focus of the paper.

At first glance, delaying PA does not appear to be of much use. However, after a more thorough thought, we find that it is useful on at least two occasions. First, it is useful in the secret key sharing between nodes in a QKD network where the nodes do not have a direct quantum link with each other but are separately connected to a common trusted relay. QKD is run between each node and the intermediate trusted relay, without running the full QKD postprocessing. Some postprocessing such as EC and PA may be delayed[1] until two nodes decide to share a key together, in which case these postprocessing steps are run only between them. This is particularly useful when the classical communication, computation, and/or energy costs associated with the trusted relay are high, for example, as in satellite-based QKD [29]. Thus, delaying some costly postprocessing parts can be beneficial. In this paper, we do not discuss the trusted relay scenario but only the validity of delaying PA in a general manner. Delaying EC is more trivial since any two parties each holding a bit string can remove errors between their strings by exchanging error syndromes with each other.

The second situation where delayed PA is useful is that we can use it to construct a two-way deterministic QKD protocol (DQKD) [30–38], whose security against general attacks is fully proved in this paper. A two-way deterministic QKD protocol is a prepare-and-measure protocol in which each signal (in our case, a qubit) makes a round trip from Bob to Alice and back to Bob. In contrast to conventional qubit-based QKD such as the BB84 protocol [1], the correct measurement basis is always used in DQKD because the signals are both prepared and measured by Bob. To encode a key bit, Alice simply applies some operation (based on her key bit value) on

*chffung@hku.hk
†xfma@iqc.ca
‡hfchau@hku.hk
§qycai@wipm.ac.cn

[1]Investigation of the possibility of delaying basis reconciliation is beyond the scope of this paper.

the qubit sent by Bob and then returns it back to Bob. Bob can decode Alice's key bit by a measurement in the same basis as what he used to prepare the initial qubit. We remark that two-way DQKD with continuous variables has been shown to have the potential for enhancing the security threshold [39]. Delayed PA may serve as a tool for proving the security of two-way continuous-variable QKD. In this paper, we focus on qubit-based two-way DQKD.

The security of qubit-based two-way DQKD had been a long-standing problem until our recent security proof of it [40]. There, we directly compute the overall density matrix of Alice, Bob, and Eve for one particular two-way DQKD protocol in which Alice uses two operators for the encoding of her bit. In this paper, we consider a different qubit-based two-way DQKD protocol in which Alice uses four encoding operators and prove its security using the delayed PA idea. We show that this particular protocol resembles the integration of the BB84 protocol and the OTP. Because of this, our analysis is significantly simplified since the security of the DQKD protocol against general attacks will then directly derive from that of the BB84 protocol [6–10,12] and the OTP [28]. We simply rely on the security results of the latter. Our proof idea is to convert the integrated scheme to the DQKD protocol through a series of equivalent protocols. We remark that the idea of integrating QKD with the OTP has been proposed before by Deng and Long [35] without a rigorous security analysis. The scheme of Deng and Long runs in a batch-after-batch manner where a batch of qubits received by Alice on the BB84 channel is stored in quantum memory first before they are used as a batch for the OTP encryption, in contrast to our scheme in which each qubit is returned to Bob immediately after reception by Alice.

The organization of the paper is as follows: After reviewing some preliminaries in Sec. II, we first prove the security of delayed PA in Sec. III. This will be an important tool that we will use in the conversion to the DQKD protocol. The DQKD protocol is described in Sec. IV, and the detailed discussion of its security proof based on the conversion argument is explained in Sec. V. To begin the conversion, we outline the initial protocols of the conversion process in Sec. V A. Then we discuss the conversion process in Sec. V B. We conclude in Sec. VI.

II. PRELIMINARIES

A. Notations

Bit strings are represented as vectors with elements in GF(2), where GF(q) is the Galois field with q elements. We use $\vec{k}$ to denote such a vector and $k[i]$ to denote the ith bit. We define the projector function $P(|\phi\rangle) \equiv |\phi\rangle\langle\phi|$. We denote the Pauli matrices by

$$I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad Z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad Y = iXZ,$$

and the corresponding eigenstates by $\{|0_w\rangle, |1_w\rangle\}$, where $w \in \{x,y,z\}$.

B. Security measure

We adopt the universal composability definition of security first proposed by Canetti [41]. This definition quantifies the security of a cryptographic primitive in terms of its deviation from the ideal functionality. The notion of universal composability has been extended to the QKD setting [42,43].

Definition 1 [10,44,45]. A classical random variable K (representing the key) drawn from the set $\mathcal{K}$ is said to be ϵ-secure with respect to an eavesdropper holding a quantum system E if

$$\tfrac{1}{2}\mathrm{Tr}|\rho_{KE} - \rho_U \otimes \rho_E| \leqslant \epsilon, \tag{1}$$

where $\rho_{KE} = \sum_{k\in\mathcal{K}} P_K(k)|k\rangle\langle k| \otimes \rho_{E|K=k}$ is the state of the systems K and E, $P_K(k)$ is the probability of having $K = k$, $\rho_U = \sum_{k\in\mathcal{K}} |k\rangle\langle k|/|\mathcal{K}|$ represents an ideal key taking values uniformly over $\mathcal{K}$, and $|\mathcal{K}|$ is the size of $\mathcal{K}$. Here, $\mathrm{Tr}|A| = \sum_i |\lambda_i|$, where λ_i's are the eigenvalues of A.

QKD expands a shorter secret key to a longer one. When one round of QKD that is ϵ_1-secure expands on a key generated by a previous round of QKD that is ϵ_2-secure, the composition of the two rounds is $(\epsilon_1 + \epsilon_2)$-secure [43].

C. Additive functions

In this paper, we consider PA functions that are additive. A function $f : \mathrm{GF}(q)^N \to \mathrm{GF}(q)^{N_{\mathrm{PA}}}$ is said to be additive if $f(\vec{a} + \vec{b}) = f(\vec{a}) + f(\vec{b})$ for all $\vec{a},\vec{b} \in \mathrm{GF}(q)^N$ and the addition operators are defined in the respective fields. For prime q, the function f is additive if and only if it can be expressed in the form of matrix multiplication $f(\vec{a}) = A\vec{a}$, where addition and multiplication are defined in GF(q). The "if" part of this statement is obvious. To prove the "only if" part, note that additivity implies linearity when q is prime since every $\vec{a} \in \mathrm{GF}(q)^N$ can be expressed as a pure summation of unweighted basis vectors of some basis (e.g., a weighting of 2 is broken down into a sum of two terms as in $\vec{a} = 2\vec{v} = \vec{v} + \vec{v}$, with $\vec{v}$ being a basis vector) and $f(\vec{a}) = \sum_{i=1}^N f(a[i]\vec{v}_i) = \sum_i a[i] f(\vec{v}_i)$, where $\vec{v}_i$'s are the basis vectors. Thus, the columns of A are $f(\vec{v}_i)$'s.

We work in GF(2) throughout the paper and so $q = 2$ and we can always consider PA functions of the form $f(\vec{a}) = A\vec{a}$. For $q = p^r$ being a power of a prime, we can also restrict additive f to be of the same form in the following sense. We may view GF(q) as a vector space over its subfield GF(p), since any element of GF(q) can be written in the form $\sum_{i=1}^r \gamma_i\beta_i$, where $\gamma_i \in \mathrm{GF}(p)$, $\beta_i \in \mathrm{GF}(q)$, and β_i cannot be expressed in the form $\gamma\beta_j$ for $j \neq i$ and some $\gamma \in \mathrm{GF}(p)$. This means that an element of GF(q) can be represented as a length-r vector with elements γ_i. Thus, when we express the PA function in the prime field so that $f : \mathrm{GF}(p)^{rN} \to \mathrm{GF}(p)^{rN_{\mathrm{PA}}}$, the statement that f is additive if and only if $f(\vec{a}) = A\vec{a}$ for all $\vec{a} \in \mathrm{GF}(p)^{rN}$ also holds. Of course, this does not mean that f can be expressed as $A\vec{a}$ for all $\vec{a} \in \mathrm{GF}(p^r)^N$.

Note that the number of additive PA functions grows exponentially as N increases. This makes Eve's job to attack a key distribution scheme more difficult with larger N as she has to make guesses on the PA function to be used by Alice and Bob in order to customize her attack. Also, additivity is not a very strong constraint and additive functions are commonly used. For example, in Toeplitz-matrix-based PA [22,23,46,47], the PA function $f(\vec{a}) = A\vec{a}$ is additive where A is a Toeplitz matrix.

A property of an additive function f is that any image of f is a translation of the kernel by some offset. This means that the number of elements in every image is the same. We use this property in the proof of Theorem 1.

III. DELAYED PRIVACY AMPLIFICATION

Suppose Alice has an N-bit raw key $\vec{a}$ on which Eve has some information. The raw key, which may not be completely secure, can be turned into a shorter secure final key by applying PA. Bob initially holds an N-bit raw key $\vec{b}$ which is a noisy version of $\vec{a}$ and, for the current discussion of delayed PA, we assume that Bob can correct all errors so that he also holds $\vec{a}$. We denote the function chosen by Alice and Bob for PA as f (mapping N bits to $N_{\mathrm{PA}} < N$ bits) and the secure key shared between Alice and Bob as $f(\vec{a})$. Normally, to encrypt an N_{PA}-bit message $\vec{m}'$, Alice computes $f(\vec{a}) \oplus \vec{m}'$ and sends it to Bob. Bob can recover the original message $\vec{m}'$ by XORing[2] the encrypted message with the shared key $f(\vec{a})$ [see Fig. 3]. Eve, in the middle of Alice and Bob, can see the encrypted message $f(\vec{a}) \oplus \vec{m}'$, but cannot get information on the original message $\vec{m}'$ because she does not know $f(\vec{a})$.

In a delayed PA scheme, Alice expands the original message $\vec{m}'$ to $\vec{m}$ for encryption with the pre-PA key $\vec{a}$. We call the expanded message $\vec{m}$ the PA inverse of $\vec{m}'$. To do this securely, as we show below, Alice should choose $\vec{m}$ (an N-bit string) uniformly among all strings that satisfy $\vec{m}' = f(\vec{m})$. Alice then sends $\vec{a} \oplus \vec{m}$ to Bob [see Fig. 4]. We demand that f be additive. Thus, anyone who receives this string can apply f to get $f(\vec{a} \oplus \vec{m}) = f(\vec{a}) \oplus \vec{m}'$, which is the encrypted message sent in the normal OTP situation. In particular, Bob can recover the original message by applying f and XORing with the shared key $f(\vec{a})$. Alternatively, Bob can recover the original message by XORing the received data with the pre-PA key $\vec{a}$ and applying f. The security of the original message $\vec{m}'$ is not obvious, as Eve sees $\vec{a} \oplus \vec{m}$ in this delayed PA scheme, not $f(\vec{a} \oplus \vec{m})$ in the normal OTP scheme. Nevertheless, we show in the following theorem that the security of the delayed PA scheme is identical to that of the normal OTP scheme.

Theorem 1 (Security of delayed PA). Given an additive function f that maps an N-bit string to an N_{PA}-bit string and that the final key $f(\vec{a})$ for some $\vec{a}$ is ϵ-secure against Eve according to security Definition 1, then, for some N_{PA}-bit message $\vec{m}'$ chosen independently of $\vec{a}$, $\vec{m}'$ is ϵ-secure against Eve (i.e., with the same security level) when she sees $\vec{a} \oplus \vec{m}$, where $\vec{m}$ is uniformly chosen among all strings that satisfy $\vec{m}' = f(\vec{m})$.

Proof. Due to the security of the OTP, $\vec{m}'$ is secure when Eve sees $f(\vec{a}) \oplus \vec{m}'$. Starting with this condition, we convert it to the final condition claimed in the theorem. Let $f^{-1}[\vec{\alpha}]$ be the inverse image of $\vec{\alpha}$ under f. Thus, Eve seeing $f(\vec{a}) \oplus \vec{m}'$ is effectively the same as Eve seeing $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ since Eve knows f and thus can compute the former given the latter and vice versa. First note that $f^{-1}[\vec{\alpha}]$ has $2^{N-N_{\mathrm{PA}}}$ elements irrespective of $\vec{\alpha}$. This is because f is additive and has the form $f(\vec{a}) = A\vec{a}$, where A is an $N_{\mathrm{PA}} \times N$ matrix. So, every inverse image is in fact an affine subspace that can be translated to the kernel of f by an offset. Hence, all inverse images $f^{-1}[\vec{\alpha}]$ for all $\vec{\alpha}$ have the same number of elements.[3] This is important since this allows us to use a random variable v independent of $\vec{\alpha}$ to select an element in the set $f^{-1}[\vec{\alpha}]$. The variable v has a fixed range and is drawn uniformly.

As the final part of the argument, note that giving Eve a random element of $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ chosen according to v is equivalent to Eve seeing all the elements of $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ and v. Since v is independent of the elements of $f^{-1}[f(\vec{a}) \oplus \vec{m}']$, knowing v gives Eve no extra information about $f(\vec{a})$ or $\vec{m}'$ over what knowing $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ gives. Thus, from Eve's point of view, seeing a random element of the set $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ is equivalent to seeing the whole set. Choosing $\vec{s}$ uniformly in $f^{-1}[f(\vec{a}) \oplus \vec{m}']$ means choosing $\vec{s}$ uniformly such that $f(\vec{s}) = f(\vec{a}) \oplus \vec{m}'$ or $f(\vec{s} \oplus \vec{a}) = \vec{m}'$. By defining $\vec{m} = \vec{s} \oplus \vec{a}$, we arrive at the claim of the theorem. ■

Remark 1. We note that delaying PA does not affect the security and the key generation length. The reason is as follows. Theorem 1 proves that the same security level is achieved by delaying PA with the same PA function. Since the PA function defines the final key length, the key generation length is not affected.

A. Special messages

We note the following special cases.

(i) Random message. If the original message $\vec{m}'$ is also uniformly chosen (acting as a key), $\vec{m}$ can be uniformly chosen without regard to the condition $\vec{m}' = f(\vec{m})$.

(ii) Imperfect key as message. If the original message $\vec{m}'$ is an imperfect key, we can delay the PA of it together with $\vec{a}$. For instance, suppose that $\vec{m}' = g(\vec{a}')$ is secure after applying the PA function g to the insecure key $\vec{a}'$. Then, the encrypting party can send $\vec{a} \oplus \vec{m}$, where $\vec{m}$ is uniformly chosen among all strings that satisfy $g(\vec{a}') = f(\vec{m})$.

B. Computation of the PA inverse

To apply the delayed PA scheme, given the original message $\vec{m}'$, Alice needs to compute its inverse by choosing $\vec{m}$ uniformly among all strings that satisfy $\vec{m}' = f(\vec{m})$. Here, we offer a method that Alice can use to compute the PA inverse $\vec{m}$. Note that this is one possible method, there may be other methods with different efficiencies to perform the same task.

Our method goes as follows. Since f is imposed to be additive, it can be represented by a matrix multiplication in GF(2), the finite field of two elements:

$$\vec{m}' = f(\vec{m}) = A\vec{m}, \tag{2}$$

where A is an $N_{\mathrm{PA}} \times N$ matrix with entries in GF(2). Multiplication of two elements is AND,[4] while addition is XOR.

[2]Exclusive OR (XOR), denoted by $\oplus$, is an operation on two bits, such that $i \oplus j = 0$ if both $i = j$ and $i \oplus j = 1$ otherwise. XOR can be extended to become an operation on two bit-strings by XORing each bit pair independently.

[3]Sec. III B describes the computation of $f^{-1}[\vec{\alpha}]$ and shows explicitly how to find the $2^{N-N_{\mathrm{PA}}}$ elements for a given $\vec{\alpha}$.

[4]AND is an operation on two bits such that i AND $j = 1$ only when $i = j = 1$.

We assume that the rows of A are linearly independent. Thus, we can apply row operations (XOR of two rows) based on Gaussian elimination to express A in upper triangular form:

$$A = R\begin{bmatrix} * & * & \cdots & \cdots & * \\ 0 & * & \cdots & \cdots & * \\ & & \ddots & & \\ 0 & \cdots & 0 & * \cdots & * \\ 0 & \cdots & 0 & 0 \; * \cdots & * \end{bmatrix}, \tag{3}$$

where the last row has $N_{\mathrm{PA}} - 1$ zeros at the beginning and R is an $N_{\mathrm{PA}} \times N_{\mathrm{PA}}$ matrix representing the row operations with $RR = I$. Thus, given $R\vec{m}'$, we can find $\vec{m}$ by randomly choosing the last $N - N_{\mathrm{PA}}$ elements of $\vec{m}$ and successively determining the remaining elements of $\vec{m}$ by using the triangular structure.

C. Example usage: A simple relay

Suppose Bob and Charlie want to establish a secret key, but they do not have a direct quantum link with each other. Instead, Bob has a quantum link with Alice (a relay) who has already shared a huge supply (denoted as pool $\mathcal{P}$) of perfectly secure[5] key bits with Charlie. Normally, Alice and Bob would run BB84 to generate from an N-bit raw key $\vec{a}$ an $N_{\mathrm{PA}} < N$-bit final key $f(\vec{a})$. With this key, Alice can OTP-encrypt N_{PA} bits from pool $\mathcal{P}$ and send the cipher text $f(\vec{a}) \oplus \vec{m}'$ to Bob, where $\vec{m}'$ denotes the bits from the pool. Bob then recovers the confidential message $\vec{m}'$ that Charlie knows and this completes the task of sharing a key between Bob and Charlie through Alice. If the key $f(\vec{a})$ is ϵ-secure against Eve according to the universally composable definition in Definition 1, the originally perfectly secure key $\vec{m}'$ now becomes ϵ-secure.

Suppose that the costs of classical communication, computation, and/or energy of Alice are high. PA can be costly in these aspects. In particular, performing PA requires one party to transmit the full specification of the PA function $f(\cdot)$ to the other party. For example, Toeplitz-matrix-based PA needs $N + N_{\mathrm{PA}} - 1$ bits to specify [22,23,46,47], which can be a big number when the block size is large. Also, performing Toeplitz-matrix-based PA requires large matrix multiplication, which translates to large computation and energy needs. These can be costly for a satellite relay, for example. In order to reduce these costs, Alice and Bob can delay PA and turn it over to Bob and Charlie. To illustrate the idea of delayed PA, we assume for simplicity that bit and phase error testing and error correction are performed between Alice and Bob as normal. Based on the phase error rate, Bob as in the normal situation decides a particular PA function f. But instead of telling Alice about f, Bob tells Charlie about it. Now, with delayed PA, Alice can take N bits from pool $\mathcal{P}$ and directly OTP-encrypt them with the raw key $\vec{a}$. This N-bit cipher text $\vec{a} \oplus \vec{m}$ is transmitted to Bob, where $\vec{m}$ denotes the N bits from the pool. Bob recovers $\vec{m}$, which Charlie knows. Both Bob and Charlie apply PA f to share a final secret key $f(\vec{m})$, which has length N_{PA} and is ϵ-secure according to Theorem 1. This generates the same key as in the normal situation without delayed PA. Note that, in this example, we sacrifice more key bits between Alice and Charlie to save the communication, computation, and/or energy costs of Alice.

[5]We assume for simplicity that the perfectly secure key is established by face-to-face key exchange.

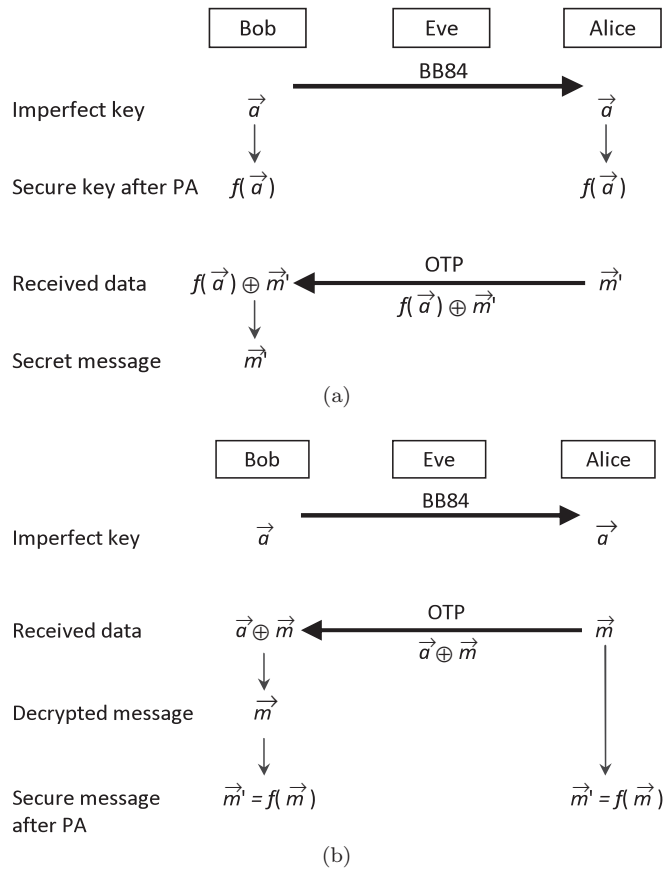

FIG. 1. Overview of delayed PA. The message $\vec{m}'$ is secure in both situations, even though Eve sees different strings in the OTP channel. We show in Theorem 1 that security is not affected by whether Eve sees $f(\vec{a}) \oplus \vec{m}'$ or $\vec{a} \oplus \vec{m}$. Here, f is the PA function which shortens its input and the arrow of the BB84 channel indicates the direction of the qubits. (a)BB84 with normal PA and OTP. (b)BB84 with PA delayed after the OTP. $\vec{m}$ can be regarded as the expanded version of the secret message $\vec{m}'$.

IV. TWO-WAY DQKD PROTOCOL

Figure 2 illustrates the two-way deterministic QKD protocol we consider in this paper, which we call Protocol DQKD. The steps of Protocol DQKD are as follows. Note that here and in the rest of paper, we present protocols in the context of Koashi's security analysis [12] in which preshared secret keys are used for encrypted communications of error correction information. However, paradigms of other security analyses [6–9] are applicable as well.

(1) Qubit transmission. Bob sends qubits to Alice taken in $\{|0_z\rangle, |1_z\rangle, |0_x\rangle, |1_x\rangle\}$ on line B-to-A.

(2) Encoding. For each qubit received by Alice, she either measures it with a random basis (check mode) or applies a random operation to it before returning it to Bob via line A-to-B (encoding mode). We call the qubit in the check mode a test bit and in the encoding mode a code bit. The operation

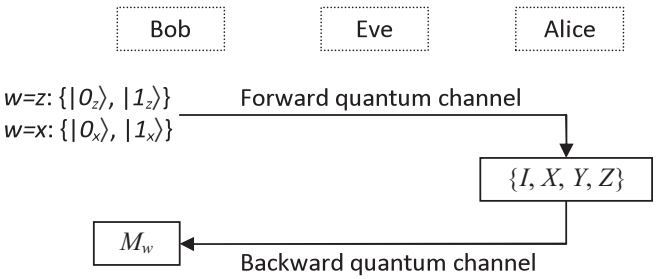

FIG. 2. Protocol DQKD. Bob chooses one of the four qubit states to send to Alice on the forward channel. She applies one of the four operations to encode her key bit and returns the qubit to Bob on the backward channel. Bob measures the received qubit in the basis he originally used for the forward qubit. The value of Alice's key bit depends on Bob's basis (see Table I).

she applies in the encoding mode is I, X, Y, or Z chosen with uniform probabilities. It does not matter whether she returns a qubit to Bob via line A-to-B in the check mode.

(3) Measurement by Bob. Bob measures each qubit received on line A-to-B in the same basis as the one he used for the state he sent to Alice on line B-to-A in Step 1.

(4) Channel estimation. After transmission of all qubits, Alice and Bob estimate the bit error rate e_b and phase error rate e_p of line B-to-A using the test bits measured in check mode in Step 2. They can do this by comparing their bit values of those qubits that Alice measured with consistent bases.

(5) Key reconciliation. Bob announces to Alice the basis used for each code bit. Alice constructs her key bit value based on the basis (see Table I): when the basis is x, the key bit is 0 (1) if she applied I or X (Z or Y); when the basis is z, the key bit is 0 (1) if she applied I or Z (X or Y). Bob uses the same rule to decide the key bit value. Note that Alice and Bob do not discard any code bit. There is no basis reconciliation step.

(6) Key bit error testing. Alice and Bob test for the error rate $e_b^{\leftrightarrows}$ in the key bits by comparing a subset of them. The remaining key bits form their raw keys, $\vec{a}$ for Alice and $\vec{b}$ for Bob. We denote the length of them by N.

(7) Final key generation. Alice and Bob choose a PA function $f(\cdot)$ that is additive and maps N bits to $N_{\rm PA} = N[1 - h(e_p)]$ bits. Alice applies PA to her raw key $\vec{a}$ to obtain the final key $\vec{k} = f(\vec{a})$. She sends Bob $Nh(e_b^{\leftrightarrows})$ bits of error correction information encrypted with preshared secret bits. This allows Bob to correct his raw key $\vec{b}$ to match Alice's $\vec{a}$. Bob then applies PA to get the same final key $\vec{k}$.

TABLE I. Key bit value dependence on the basis used by Bob (x or z) and Alice's encoding operation (I, X, Y, or Z). For example, when Bob uses basis z, bit 1 is encoded by Alice if she applies X or Y on the qubit sent by Bob.

	Bit value	
Basis	0	1
x	$\{I,X\}$	$\{Z,Y\}$
z	$\{I,Z\}$	$\{X,Y\}$

The net key expansion length is

$$N_{\rm key,two-way} = N[1 - h(e_b^{\leftrightarrows}) - h(e_p)]. \tag{4}$$

In Sec. V B, we show that the newly generated key with length $N_{\rm PA}$ is secure, and thus the net key gain given in Eq. (4) is achievable. We prove this by combining the BB84 protocol and the OTP protocol and then successively converting the OTP protocol to finally form the two-way DQKD protocol given here.

An interesting feature of two-way DQKD protocol is that every code bit encoded by Alice in the encoding mode is used for the final key generation without being wasted due to measurement basis mismatch. There is no basis reconciliation for the key bits and this is why the protocol is called *deterministic*.[6] This is in contrast to the original BB84 protocol where half of the code bits are discarded. On the other hand, the efficient BB84 protocol [48] allows all code bits to be used as well, but only asymptotically. Therefore, in finite-length situations, two-way DQKD is still more efficient in using the code bits.

Note that the test bits in the check mode of two-way DQKD are measured by Alice with a random basis and thus are subject to discarding due to basis mismatch. Thus, the check mode performances are the same in two-way DQKD and BB84.

A disadvantage of two-way DQKD is that the quantum signals emitted by Bob suffer from twice the channel loss compared to BB84.

V. SECURITY PROOF OF TWO-WAY DQKD

The security proof of the two-way DQKD protocol is based on arguing for the equivalence of the protocol with an integrated scheme, and thus the security of the former directly follows from that of the latter. The integrated scheme consists of the BB84 protocol on the forward line and the one-time pad on the backward line. The security of both are well established [6–10,12,28]. Starting with the integrated scheme in Sec. V A, we convert it to the two-way DQKD protocol in Sec. V B.

A. Original protocols for constructing two-way DQKD

Here, we outline the steps of the BB84 protocol and the OTP, which serve as the starting point of the conversion process.

1. Protocol 1 on line B-to-A: BB84 with quantum memory

We can view the line from Bob to Alice as a BB84 key distillation step. The steps of BB84 are shown below, where

[6]Note that the term deterministic was first introduced in Ref. [30] to mean that, when Alice wants to send 0 (or 1) to Bob, she can encode her bit definitely. This makes sense in quantum direct communication, but not in QKD. We apply this term to the QKD setting but only use it to mean that every code bit will be used to generate the final key, instead of that every code bit is the final key bit. This is because Alice and Bob need to run PA which is determined only after Alice has encoded all the raw key bits. Privacy amplification will then turn Alice's raw key bits into a new bit string that is different from what she initially sent.

we assume for simplicity the use of quantum memory to avoid the step of discarding bits measured with inconsistent bases.

(1) Bob sends $N + N_{\text{test}}$ qubits to Alice taken in $\{|0_z\rangle, |1_z\rangle, |0_x\rangle, |1_x\rangle\}$.

(2) Alice stores all $N + N_{\text{test}}$ received qubits in quantum memory.

(3) Bob announces the basis of each qubit, and Alice measures her qubits in the corresponding bases.

(4) Alice and Bob randomly select N_{test} test bits to find out the bit error rate e_b and phase error rate e_p for this line B-to-A.[7] Alice and Bob choose a PA function $f(\cdot)$ that is additive and maps N bits to $N_{\text{PA}} = N[1 - h(e_p)]$ bits.

(5) The final secret key is derived from Alice's raw key as an N_{PA}-bit string $\vec{k} = f(\vec{a})$. To allow Bob to obtain the same final key, Alice sends Bob $N_{\text{EC}} = Nh(e_b)$ bits of error correction information encrypted with preshared secret bits of the same size so that Bob can correct his raw key $\vec{b}$ to become $\vec{a}$. He then applies the same PA function $f(\cdot)$ to get the final key $\vec{k}$.

Now, according to Koashi's security proof of the BB84 protocol [12] (see also other proofs [6–9]), the final key $\vec{k}$ is secure against Eve with the the net key expansion length as

$$N_{\text{key}} = N_{\text{PA}} - N_{\text{EC}} = N[1 - h(e_b) - h(e_p)]. \tag{5}$$

2. *Protocol 2 on line A-to-B: One-time pad*

We can view the line from Alice to Bob as a one-time pad encryption step.

(6) Alice encrypts an N_{PA}-bit message $f(\vec{m})$ with the secret key $\vec{k}$ with the OTP and sends the encrypted message $f(\vec{m}) \oplus \vec{k}$ to Bob over a classical channel. (As we will see later, $\vec{m}$ will be chosen randomly with uniform probabilities.)

(7) Bob decrypts his received data with key $\vec{k}$ to get the secret message $f(\vec{m})$.

Here, Eve sees $f(\vec{m}) \oplus \vec{k}$ on line A-to-B and the message $f(\vec{m})$ is secure against her because of the security of the OTP [28].

B. Conversion from original protocol

We successively convert the original Protocol 2 to new Protocols 2b, 2c, and 2d, while maintaining the same security in each step to finally arrive at the two-way DQKD protocol. Figure 3 shows the equivalent protocols and they are described in more detail in the following.

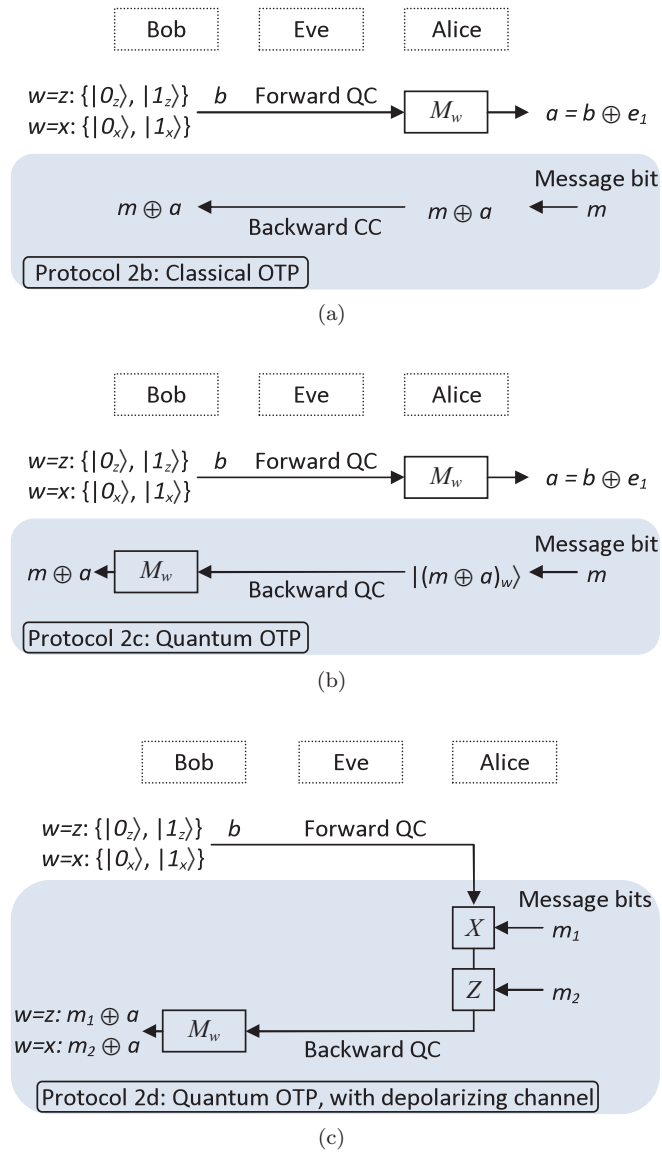

FIG. 3. (Color online) Protocols 2b, 2c, and 2d are equivalent. The equivalence between Protocols 2c and 2d can be understood intuitively by noting that both a measurement and a depolarizing operation disentangle line B-to-A and line A-to-B. This is rigorously shown by comparing their density matrices in Eq. (6) of Protocol 2c and Eqs. (8) and (9) of Protocol 2d. We identify the message bit m_1 (m_2) of Protocol 2d with m of Protocol 2c when $w = z$ ($w = x$). Thus, Protocol 2d requires Bob to inform his basis w to Alice so that she knows whether m_1 or m_2 is used for the final key generation. In all cases, privacy amplification f is delayed after the OTP to generate the final secret key $f(\vec{m})$, and the top part of each figure is Protocol 1. Here, we assume for simplicity that the backward line A-to-B is noiseless, but noise can be incorporated easily (see Sec. V C). QC, quantum channel; CC, classical channel.

1. *Protocol 2b on line A-to-B: One-time pad with delayed PA*

(6) Alice encrypts an N-bit random message $\vec{m}$ with her raw key $\vec{a}$ with one-time pad and sends the encrypted message $\vec{m} \oplus \vec{a}$ to Bob over a classical channel line A-to-B.

(7) Bob recovers the secret message $f(\vec{m})$ as follows. He applies PA to his received bits to get $f(\vec{m} \oplus \vec{a})$. Due to

[7] The average quantum bit and phase error rates e_b and e_p are related to the classical bit error rates in the x basis test bits and the z basis test bits, denoted as e_x and e_z, respectively. Asymptotically, the quantum bit error rate and the phase error rate for the remaining x (z) basis bits are e_x and e_z, respectively (e_z and e_x, respectively). Thus, the average quantum bit error rate is $e_b = (e_x + e_z)/2$ and the average quantum phase error rate is $e_b = (e_z + e_x)/2$. Even though they are the same, we use separate symbols for them to emphasize their meanings in secret key distillation.

the additivity of $f(\cdot)$, his received data are $f(\vec{m}) \oplus f(\vec{a}) = f(\vec{m}) \oplus \vec{k}$ which he can decrypt with the same key $\vec{k}$ to recover the message $f(\vec{m})$. Alternatively, he can XOR his received string $\vec{m} \oplus \vec{a}$ with the raw key $\vec{a}$ and apply PA f to recover the message $f(\vec{m})$.

Here, Eve sees $\vec{m} \oplus \vec{a}$ and this is different from the situation in Protocol 2. Nevertheless, as we have shown in Theorem 1, the security of $f(\vec{m})$ is the same as that in Protocol 2, meaning that Eve cannot get any information about $f(\vec{m})$.

2. Protocol 2c on line A-to-B: One-time pad on quantum channel, with measurement

Line A-to-B is now regarded as a quantum channel, even though we use it for the communication of the classical OTP-encrypted message. We encode the OTP-encrypted classical message in Step 6 of Protocol 2b $\vec{m} \oplus \vec{a}$ in a quantum state so that it can be carried by the quantum channel. This is easily done by encoding each bit in the eigenstate of some basis. Here, we assume that the basis used is the same basis $w \in \{x,z\}$ Bob used to encode his qubit on line B-to-A. Also, we assume for simplicity that Alice knows w for each bit. Thus, the N-qubit state Alice sends is $|(\vec{m} \oplus \vec{a})_{\vec{w}}\rangle \equiv \bigotimes_i |(m[i] \oplus a[i])_{w[i]}\rangle$, where the index i denotes the ith bit for the message, key, and basis. The modified steps are as follows.

(6) Alice encrypts an N-bit random message $\vec{m}$ with the raw key $\vec{a}$ with the one-time pad and sends the encrypted message $|(\vec{m} \oplus \vec{a})_{\vec{w}}\rangle$ to Bob over the quantum channel line A-to-B.

(7) Bob measures the N qubits received from line A-to-B in basis $\vec{w}$ to recover the OTP-encrypted message $\vec{m} \oplus \vec{a}$. He recovers the secret message $f(\vec{m})$ as in Step 7 of Protocol 2b.

Overall density matrix. We first consider the state $|\Psi[i]\rangle_{A\bar{A}}$ for the ith bit shared after Alice received her N qubits from line B-to-A (where system A is the ith qubit received by Alice on line B-to-A and system $\bar{A}$ includes all the remaining systems including Eve's and Bob's states for the N transmissions and Alice's remaining $N-1$ qubits). To simplify notation, we drop the index i from all symbols (including $|\Psi[i]\rangle$, $w[i]$, and $a[i]$) in the following since we always deal with the ith qubit. This state $|\Psi\rangle$ is the state before Alice decides to send anything on line A-to-B. In Protocol 2c, Alice measures her state of A in basis $w \in \{x,z\}$ using the projection $\{|0_w\rangle\langle 0_w|, |1_w\rangle\langle 1_w|\}$. So the overall state becomes $\sum_{a=0,1} |a_w\rangle_A |\Psi(a,w)\rangle_{\bar{A}} |a\rangle_{A'}$, where $|\Psi(a,w)\rangle_{\bar{A}} \equiv \langle a_w|_A \otimes I_{\bar{A}} |\Psi\rangle_{A\bar{A}}$ and system A' is the ancilla for storing the measurement result. We specifically isolate the raw key bit a in system A' so that we can use it to perform the OTP with the message bit m. Next, Alice prepares a random message $2^{-1}\sum_{m=0,1} |m\rangle_M\langle m|$ and runs controlled-Z (if $w = x$) or controlled-X (if $w = z$) on systems M (as control) and A. This is equivalent to the OTP encryption resulting in the overall density matrix

$$\rho_{MA\bar{A}A'} = \frac{1}{2}\sum_{m=0,1} |m\rangle_M\langle m| \otimes P\left(\sum_{a=0,1} |(m\oplus a)_w\rangle_A |\Psi(a,w)\rangle_{\bar{A}} |a\rangle_{A'}\right),$$

in which system A is sent by Alice on line A-to-B to Bob and system M is her message bit.

After the OTP encryption, the raw key bit a is no longer needed. Thus, we trace over system A' to get the overall state

$$\rho_{MA\bar{A}} = \frac{1}{2}\sum_{\substack{a=0,1\\ m=0,1}} P(|m\rangle_M |(m\oplus a)_w\rangle_A |\Psi(a,w)\rangle_{\bar{A}}). \tag{6}$$

Note that tracing over system A' which contains the raw key bit does not mean giving the raw key bit to Eve. Eve's state is contained in system $\bar{A}$. The state in Eq. (6) is important for our discussion since it contains all the relevant systems in the protocol. In fact, Protocol 2d in the next section will be shown to be equivalent to Protocol 2c here by showing that the corresponding states there are the same as those in Eq. (6).

3. Protocol 2d on line A-to-B: One-time pad on quantum channel, without measurement

In the previous Protocol 2c, the measurement by Alice disentangles line B-to-A and line A-to-B. Therefore, to come up with an equivalent protocol without a measurement, we need to reproduce this disentanglement feature and at the same time achieve the same overall state in Eq. (6). One way to do this is by replacing the measurement by a depolarizing channel. Starting with the same initial state $|\Psi\rangle_{A\bar{A}}$ for the ith bit as in the previous subsection, Alice performs randomly with uniform probabilities the operations I, X, Y, or Z on each of her N qubits independently. We express this random operation as Alice using a mixed state $4^{-1}\sum_{m_1,m_2=0,1} |m_1m_2\rangle_{M_1M_2}\langle m_1m_2|$ to control the four operations on system A:

$$|\Psi\rangle_{A\bar{A}}\langle\Psi| \to \frac{1}{4}\sum_{m_1,m_2=0,1} |m_1m_2\rangle_{M_1M_2}\langle m_1m_2| \otimes (X^{m_1}Z^{m_2})_A |\Psi\rangle_{A\bar{A}}\langle\Psi| (Z^{m_2}X^{m_1})_A \tag{7}$$

Here, we assume Alice holds the purification of this mixed state. Tracing the right-hand side over M_1 or M_2, simple calculations (see the Appendix) lead to

$$\rho_{M_1A\bar{A}} = \frac{1}{2}\sum_{\substack{a=0,1\\ m_1=0,1}} P(|m_1\rangle_{M_1} |(m_1\oplus a)_z\rangle_A |\Psi(a,z)\rangle_{\bar{A}}), \tag{8}$$

$$\rho_{M_2A\bar{A}} = \frac{1}{2}\sum_{\substack{a=0,1\\ m_2=0,1}} P(|m_2\rangle_{M_2} |(m_2\oplus a)_x\rangle_A |\Psi(a,x)\rangle_{\bar{A}}). \tag{9}$$

Note that Eqs. (8) and (9) are expressed in terms of bases z and x, respectively, regardless of the actual basis w used by Bob to encode system A.

We argue that Protocol 2c and Protocol 2d are the same as follows. Equations (8) and (9) represent the final overall state of Protocol 2d and we compare them to that of Protocol 2c in Eq. (6). We can see that when $w = z$ ($w = x$), we can identify m_1 in Eq. (8) [Eq. (9)] with m in Eq. (6). Thus, when $w = z$ ($w = x$), we can regard that Alice's message bit is in m_1 (m_2). Once the basis w is publicly announced by Bob, all of Alice, Bob, and Eve will know whether m_1 or m_2 will be used by Alice; in other words, they will know which of Eqs. (8) and (9) describes the situation. Therefore, Protocol 2c and Protocol 2d are the same from Eve's and Bob's points of view.

In Protocol 2d, we need a step for Bob to inform Alice about w so that she knows whether m_1 or m_2 is her message bit. Note that when Alice performs one of the four operations of the depolarizing channel, she does not know what the message bit value is (unless $m_1 = m_2$). After Bob receives his qubit, he announces to Alice his basis choice w and only then does Alice know the value of her own message bit.

The modified steps of Protocol 2d are as follows.

(6) Alice chooses two N-bit random messages, $\vec{m}_1$ and $\vec{m}_2$. For each qubit received from line B-to-A, she applies Z if $m_2 = 1$ and applies X if $m_1 = 1$.[8] The qubit is then forwarded back to Bob via line A-to-B.

(7) Bob measures the N qubits received from line A-to-B in basis w which he has used in Step 1.

(8) Bob announces to Alice the basis for each qubit. If the basis is z (x), Alice's message bit is m_1 (m_2). So in the previous step, Bob's measured qubit corresponds to Alice's OTP-encrypted message bit $m_1 \oplus a$ (for $w = z$) or $m_2 \oplus a$ (for $w = x$). He recovers the secret message $f(\vec{m})$ as in Step 7 of Protocol 2b, with the appropriate substitution $m_1 \to m$ or $m_2 \to m$ for each bit. Alice also constructs the secret message $f(\vec{m})$ with the same substitution.

Therefore, when line A-to-B is noiseless, Alice and Bob will share the message bit m_1 (for $w = z$) or m_2 (for $w = x$). When line A-to-B is noisy, we can add further error testing and error correction for $\vec{m}$, which we have omitted for simplicity of discussion. Finally, we note that combining Protocol 1 and Protocol 2d essentially gives Protocol DQKD given in Sec. IV.[9] Thus, we have proved the security of Protocol DQKD.

We remark that it makes sense that Alice's message bit depends on the basis used by Bob. Because when Bob initially sends, for example, a z eigenstate to Alice via line B-to-A, only Alice's X operation (controlled by m_1) will bit flip the state, and so m_1 should become her message bit.

C. Key generation rates

In Protocols 2, 2b, 2c, and 2d, we assume that the final key is derived from applying PA to Alice's raw key: $\vec{k} = f(\vec{a})$. Bob is responsible for correcting his raw key to match Alice's. To ensure security, Alice's message $\vec{m}$ is shortened to $N_{\rm PA} = N[1 - h(e_p)]$ bits of secure message $f(\vec{m})$, where e_p is obtained from Step 4 of Protocol 1. In the discussion so far, we have not considered errors on line A-to-B. Errors on line B-to-A cause Alice's raw key to be different from Bob's raw key such that $\vec{b} = \vec{a} \oplus \vec{e}_1$, where $\vec{e}_1$ is the error pattern with an error rate of e_b (cf. Step 4 of Protocol 1). Errors on line A-to-B cause Bob to receive $\vec{m} \oplus \vec{a} \oplus \vec{e}_2$ in Step 6 of Protocol 2b, where $\vec{e}_2$ is the error pattern on this line and could be correlated with $\vec{e}_1$. Thus, when Bob uses $\vec{b}$ to decrypt his message received on line A-to-B, he faces the error pattern $\vec{e}_1 \oplus \vec{e}_2$, whose error rate we denote as $e_b^{\leftrightarrows}$. To help Bob correct for this error pattern, Alice sends to Bob error correction information encrypted with $N(e_b^{\leftrightarrows})$ bits of the preshared secret key. Therefore, the net key expansion length is

$$N_{\rm key,two-way} = N[1 - h(e_b^{\leftrightarrows}) - h(e_p)]. \tag{10}$$

This represents the key generation rate for the integration of Protocol 1 and any of Protocols 2, 2b, 2c, and 2d. As expected, this is the same formula for the two-way DQKD protocol given in Eq. (4). As a special case, when the error rates on the two lines are both e_b, the overall error rate $e_b^{\leftrightarrows}$ is upper-bounded by $2e_b$ since the errors on the two lines can be correlated. Thus, the key generation rate in this case is $1 - h(2e_b) - h(e_p)$, which is the same as that derived for another two-way DQKD protocol in Ref. [40] (see Sec. III F therein).

Note that as Alice and Bob are correcting the overall error pattern $\vec{e}_1 \oplus \vec{e}_2$, they do not need to separately correct for the error $\vec{e}_1$ in their raw keys; i.e., they do not perform Step 5 of Protocol 1. This is reflected in the original two-way DQKD protocol in Sec. IV.

VI. CONCLUSIONS

The central idea of our paper is delayed PA and we have proved its security. Delayed PA is useful for secret key sharing between nodes of a QKD network assisted by trusted relays and for the security proof of a qubit-based two-way DQKD protocol. We anticipate that delayed PA will have further uses in other applications, such as the security proof of two-way continuous-variable QKD [39].

In this paper, we derived the qubit-based two-way DQKD protocol from an integration of the BB84 protocol and the OTP, with the condition that PA is delayed after the OTP . Because of our security proof of delayed PA, the original security of BB84 directly carries over to the DQKD protocol. Thus, we have proved the security of the DQKD protocol against general attacks with qubit signals. This illustrates the power of the delayed PA idea.

Security analysis of DQKD with multiqubit signals and decoy states [49–51] is beyond the scope of the current paper and will be left for future work. Also, using nonadditive PA functions in delayed PA will be considered in the future.

ACKNOWLEDGMENTS

We thank H.-K. Lo for enlightening discussions. This work is supported in part by RGC under Grant No. 700709P of the HKSAR Government, NSERC, the CRC Program, CIFAR, QuantumWorks, and the NSFC under Grant No. 11074283.

APPENDIX: DERIVATION OF EQ. (8) FROM EQ. (7)

We only derive Eq. (8) from Eq. (7). The derivation of Eq. (9) is similar. First, we decompose $|\Psi\rangle_{A\bar{A}} = \sum_{i=0,1} \lambda_i |i_z\rangle_A |e_i\rangle_{\bar{A}}$, where $|i_z\rangle_A$ are the normalized eigenstates of basis z and $|e_i\rangle_{\bar{A}}$ are normalized but not necessarily

[8]Note that the order of applications of these two operations does not matter in light of Eq. (7) as swapping the order contributes a factor of -1 twice.

[9]The key bit error testing of Step 6 of Protocol DQKD is omitted in Protocol 2d for simplicity of discussion, but this step can easily be added without affecting the result.

orthogonal. We trace the state of Eq. (7) over M_2:

$$\begin{aligned}\rho_{M_1A\bar{A}} &= \frac{1}{4}\sum_{m_1=0,1}|m_1\rangle_{M_1}\langle m_1|\otimes\left[P\left(X_A^{m_1}|\Psi\rangle_{A\bar{A}}\right)+P((X^{m_1}Z)_A|\Psi\rangle_{A\bar{A}})\right]\\ &= \frac{1}{4}|0\rangle_{M_1}\langle 0|\otimes[P(\lambda_0|0_z\rangle_A|e_0\rangle_{\bar{A}}+\lambda_1|1_z\rangle_A|e_1\rangle_{\bar{A}})+P(\lambda_0|0_z\rangle_A|e_0\rangle_{\bar{A}}-\lambda_1|1_z\rangle_A|e_1\rangle_{\bar{A}})]\\ &\quad+\frac{1}{4}|1\rangle_{M_1}\langle 1|\otimes[P(\lambda_0|1_z\rangle_A|e_0\rangle_{\bar{A}}+\lambda_1|0_z\rangle_A|e_1\rangle_{\bar{A}})+P(\lambda_0|1_z\rangle_A|e_0\rangle_{\bar{A}}-\lambda_1|0_z\rangle_A|e_1\rangle_{\bar{A}})]\\ &= \frac{|\lambda_0|^2}{2}\left[|0\rangle_{M_1}\langle 0|\otimes|0_z\rangle_A\langle 0_z|+|1\rangle_{M_1}\langle 1|\otimes|1_z\rangle_A\langle 1_z|\right]\otimes|e_0\rangle_{\bar{A}}\langle e_0|\\ &\quad+\frac{|\lambda_1|^2}{2}\left[|0\rangle_{M_1}\langle 0|\otimes|1_z\rangle_A\langle 1_z|+|1\rangle_{M_1}\langle 1|\otimes|0_z\rangle_A\langle 0_z|\right]\otimes|e_1\rangle_{\bar{A}}\langle e_1|.\end{aligned}$$

The last equation is equal to Eq. (8) by noting that $|\Psi(k,z)\rangle_{\bar{A}}\equiv\langle k_z|_A\otimes I_{\bar{A}}|\Psi\rangle_{A\bar{A}}=\lambda_k|e_k\rangle_{\bar{A}}$.

The derivation of Eq. (9) from Eq. (7) can be done in a similar manner by decomposing $|\Psi\rangle_{A\bar{A}}$ in the x basis as $|\Psi\rangle_{A\bar{A}}=\sum_{i=0,1}\lambda_i'|i_x\rangle_A|e_i'\rangle_{\bar{A}}$.

[1] C. H. Bennett and G. Brassard, in *Proceedings of the IEEE Inernational Conference on Computers, Systems, and Signal Processing* (IEEE Press, New York, 1984), pp. 175–179.
[2] A. K. Ekert, Phys. Rev. Lett. **67**, 661 (1991).
[3] D. Mayers, J. Assoc. Comput. Mach. **48**, 351 (2001). preliminary version in D. Mayers, *Advances in Crytology—Proceedings of Crypto '96*, Vol. 1109 of Lecture Notes in Computer Science, edited by N. Koblitz (Springer, New York, 1996), pp. 343–357
[4] E. Biham, M. Boyer, P. O. Boykin, T. Mor, and V. Roychowdhury, in *Proceedings of the Thirty-Second Annual ACM Symposium on Theory of Computing* (ACM Press, New York, 2000), pp. 715–724.
[5] H. Inamori, N. Lütkenhaus, and D. Mayers, Eur. Phys. J. D **41**, 599 (2007).
[6] H.-K. Lo and H. F. Chau, Science **283**, 2050 (1999).
[7] P. W. Shor and J. Preskill, Phys. Rev. Lett. **85**, 441 (2000).
[8] B. Kraus, N. Gisin, and R. Renner, Phys. Rev. Lett. **95**, 080501 (2005).
[9] R. Renner, N. Gisin, and B. Kraus, Phys. Rev. A **72**, 012332 (2005).
[10] R. Renner and R. König, in *Proceedings of the Second Theory of Cryptography Conference (TCC) 2005*, Vol. 3378 of Lecture Notes in Computer Science (Springer, Berlin, 2005), pp. 407–425.
[11] D. Gottesman, H.-K. Lo, N. Lütkenhaus, and J. Preskill, Quantum Inf. Comput. **5**, 325 (2004).
[12] M. Koashi, New J. Phys. **11**, 045018 (2009).
[13] T. Tsurumaru and K. Tamaki, Phys. Rev. A **78**, 032302 (2008).
[14] N. J. Beaudry, T. Moroder, and N. Lütkenhaus, Phys. Rev. Lett. **101**, 093601 (2008).
[15] C.-H. F. Fung, K. Tamaki, B. Qi, H.-K. Lo, and X. Ma, Quantum Inf. Comput. **9**, 0131 (2009).
[16] L. Lydersen and J. Skaar, Quantum Inf. Comput. **10**, 60 (2010).
[17] M. Hayashi, Phys. Rev. A **76**, 012329 (2007).
[18] M. Hayashi, Phys. Rev. A **74**, 022307 (2006).
[19] V. Scarani and R. Renner, Phys. Rev. Lett. **100**, 200501 (2008).
[20] V. Scarani and R. Renner, in *Lecture Notes in Computer Science*, Vol. 5106 (Springer, Berlin, 2008), p. 83.
[21] R. Y. Cai and V. Scarani, New J. Phys. **11**, 045024 (2009).
[22] C.-H. F. Fung, X. Ma, and H. F. Chau, Phys. Rev. A **81**, 012318 (2010).
[23] X. Ma, C.-H. F. Fung, J.-C. Boileau, and H. F. Chau, Comput. Secur. **30**, 172 (2011).
[24] C. E. Shannon, Bell Syst. Tech. J. **27**, 379 (1948).
[25] C. H. Bennett, G. Brassard, and J.-M. Robert, SIAM J. Comput. **17**, 210 (1988).
[26] C. H. Bennett, G. Brassard, C. Crépeau, and U. M. Maurer, IEEE Trans. Inf. Theory **41**, 1915 (1995).
[27] G. S. Vernam, in *Transactions of the American Institute of Electrical Engineers*, Vol. XLV (American Institute of Electrical Engineers, New York, 1926), pp. 295–301.
[28] C. E. Shannon, Bell Syst. Tech. J. **28**, 656 (1949).
[29] X. Ma and N. Lütkenhaus, "Simplified Trusted Repeater Node for Quantum Key Distribution", U.S. Provisional Patent Application 61/573,137, 2011.
[30] K. Boströem and T. Felbinger, Phys. Rev. Lett. **89**, 187902 (2002).
[31] A. Wójcik, Phys. Rev. Lett. **90**, 157901 (2003).
[32] Q.-Y. Cai, Phys. Rev. Lett. **91**, 109801 (2003).
[33] Q.-Y. Cai and B.-W. Li, Phys. Rev. A **69**, 054301 (2004).
[34] Q.-Y. Cai and B.-W. Li, Chin. Phys. Lett. **21**, 601 (2004).
[35] F.-G. Deng and G. L. Long, Phys. Rev. A **69**, 052319 (2004).
[36] M. Lucamarini and S. Mancini, Phys. Rev. Lett. **94**, 140501 (2005).
[37] M. Lucamarini, A. Ceré, G. Giuseppe, S. Mancini, D. Vitali, and P. Tombesi, Open Syst. Inf. Dyn. **14**, 169 (2007).
[38] K. Boström and T. Felbinger, Phys. Lett. A **372**, 3953 (2008).
[39] S. Pirandola, S. Mancini, S. Lloyd, and S. L. Braunstein, Nat. Phys. **4**, 726 (2008).
[40] H. Lu, C.-H. F. Fung, X. Ma, and Q.-Y. Cai, Phys. Rev. A **84**, 042344 (2011).

[41] R. Canetti, in *Proceedings of the 42nd IEEE Symposium on Foundations of Computer Science (FOCS) 2001* (IEEE Press, New York, 2001), pp. 136–145.
[42] M. Ben-Or and D. Mayers, e-print arXiv:quant-ph/0409062.
[43] M. Ben-Or, M. Horodecki, D. W. Leung, D. Mayers, and J. Oppenheim, in *Proceedings of the Second Theory of Cryptography Conference (TCC) 2005*, Vol 3378 of Lecture Notes in Computer Science (Springer, Berlin, 2005), pp. 386–406.
[44] R. König, R. Renner, A. Bariska, and U. Maurer, Phys. Rev. Lett. **98**, 140502 (2007).
[45] R. Renner, Ph.D. thesis, Swiss Federal Institute of Technology, 2005, also available in Int. J. Quantum Inf. **6**, 1 (2008).
[46] Y. Mansour, N. Nisan, and P. Tiwari, Theor. Comput. Sci. **107**, 121 (1993).
[47] H. Krawczyk, in *Advances in Cryptology—CRYPTO '94, Lecture Notes in Computer Science*, Vol. 893 (Springer-Verlag, Berlin, 1994), pp. 129–139.
[48] H.-K. Lo, H. F. Chau, and M. Ardehali, J. Cryptol. **18**, 133 (2005).
[49] W.-Y. Hwang, Phys. Rev. Lett. **91**, 057901 (2003).
[50] H.-K. Lo, X. Ma, and K. Chen, Phys. Rev. Lett. **94**, 230504 (2005).
[51] X.-B. Wang, Phys. Rev. Lett. **94**, 230503 (2005).