


# Stream Privacy Amplification for Quantum Cryptography

Yizhi Huang<sup>1</sup>, Xingjian Zhang<sup>1</sup>, and Xiongfeng Ma<sup>1\*</sup>

*Center for Quantum Information, Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing 100084, China*

 (Received 25 January 2022; accepted 16 May 2022; published 10 June 2022; corrected 13 July 2022)

Privacy amplification is the key step to guarantee the security of quantum communication. The existing security proofs require the accumulation of a large number of raw key bits for privacy amplification. This is similar to block ciphers in classical cryptography that would delay the final key generation since an entire block must be accumulated ahead of privacy amplification. Moreover, any leftover errors after information reconciliation would corrupt the entire block. By modifying the security proof based on quantum error correction, we develop a stream privacy-amplification scheme, which resembles the classical stream cipher. This scheme can output the final key in a stream way, prevent error from spreading, and hence can put privacy amplification ahead of information reconciliation. The stream scheme can also help to enhance the security of trusted-relay quantum networks and improve the practicality of randomness extraction for quantum random number generators.

DOI: [10.1103/PRXQuantum.3.020353](https://doi.org/10.1103/PRXQuantum.3.020353)

## I. INTRODUCTION

As one of the first applications of quantum-information science, quantum key distribution (QKD) aims at establishing an information-theoretic secure key between two distant parties, Alice and Bob [1,2]. It applies fundamental laws of quantum physics to guarantee secure communication. The procedures of QKD can be divided into two parts: quantum operation and data postprocessing. Quantum operation includes the preparation, transmission, and measurement of quantum states for Alice and Bob to share raw key bits. The purpose of postprocessing is to extract an identical and private key from the raw data. This can be guaranteed by information reconciliation and privacy amplification, where the former guarantees the identity of key strings and the latter removes any potential information leakage to a possible eavesdropper, Eve [3,4].

Over the past three decades of development, QKD has experienced tremendous advancement from initial demonstrations in laboratories to practical implementation [5]. In fiber, the communication distance has been pushed over 500 km [6,7]. Using quantum satellites, the communication distance has reached an intercontinental level [8]. Researchers have also pushed QKD to a secret key rate of

more than 10 Mbits/s [9]. In addition to point-to-point linking, a number of field-test QKD networks have been conducted in many countries [10–15]. In particular, China has recently successfully completed the 2000-km-long fiber-optic backbone link between Beijing and Shanghai [16]. Therefore, QKD is already a mature technique for real-life applications [17].

With the exciting developments on the experimental side, improving the practicality of QKD systems has become one of the essential issues in the field. Among all the stages in a QKD session, privacy amplification is one of the bottlenecks, which might still be technically difficult to implement in some realistic conditions. Existing privacy-amplification methods run as follows. After information reconciliation, Alice randomly chooses a hash function and sends it to Bob via a public classical channel. Both users hash their reconciled key strings with the hash function and obtain the final key. In practice, the family of Toeplitz-matrix hashing is widely adopted. Due to the matrix multiplication, privacy amplification can only process a block of reconciled key bits at a time. Though recent security-analysis techniques have shown that the key rate can still be positive, with critical block sizes on the order of kilobits [18], smaller block sizes tend to make privacy amplification inefficient, resulting in lower key rates. To guarantee the efficiency of privacy amplification, the block size is normally large in practice, typically on the order of megabits [18]. Alice and Bob cannot perform privacy amplification until they accumulate an entire block of a reconciled key. This block feature of privacy amplification would cause unpleasant delays in some practical scenarios. For instance, in the satellite case, since the quantum signals

\*xma@tsinghua.edu.cn

*Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International](https://creativecommons.org/licenses/by/4.0/) license. Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.*

can be transferred only when the ground station can “see” the satellite with a clear atmosphere, it may take the satellite several orbits to accumulate enough data for one block of privacy amplification. Due to the unpredictable condition of the atmosphere, such a delay could take as long as days [19].

There are other cases where block privacy amplification could cause problems. If the ratio of the reconciled key to the final key is extremely large, the computational cost for the hash function can be very heavy. Such an issue has been encountered in randomness extraction of quantum random number generation (QRNG) as well, especially in the device-independent case [20]. Due to the similarity between the definitions of randomness extraction and privacy amplification [21], randomness extractors can be constructed by using universal hashing functions [22,23] and have the same block feature as existing privacy-amplification schemes. The problem of the heavy computational cost is more serious in QRNG due to the larger amount of data and it restricts further improvement of the real-time generation speed.

Moreover, with block privacy amplification, the leftover error in information reconciliation would spread out to the entire block. Information reconciliation is normally done by bilateral error correction. For some error-correction schemes, there is a small probability of leaving some errors uncorrected. If Alice’s and Bob’s two strings are not exactly the same, the output strings from block privacy amplification will be totally different due to the universal property of the hash family.

Furthermore, existing quantum network implementations rely on trusted relays for key distribution [16], due to the limited transmission distance of point-to-point QKD links. Trusted QKD networks have been widely used in building intercity or backbone QKD communication links, such as the Hefei network [24] and the Beijing-Shanghai backbone link [16]. In the trusted-network scenario, all intermediate relays must be trusted because each of the relays can produce the final key. If one relay becomes compromised, the security of the whole network will be seriously threatened. In practice, it is challenging and expensive to guarantee high-level secure relays, which hinders the further commercialization and application of QKD. There are some attempts to reduce the dependence on trusted relays. Unfortunately, these solutions either require duplicate resources [25] or still assume that the intermediate relays do not attack the network intentionally [26].

To address these issues, we reexamine the security proof for QKD based on quantum error correction [27], where privacy amplification is reduced from phase-error correction [28]. As a clear and simple showcase, we mainly focus on the Bennett-Brassard-1984 QKD protocol (BB84) [1] and go back to the original Lo-Chau security proof [27]. By rearranging the phase-error-correcting gates and error-syndrome measurement, we divide privacy

amplification into two steps: (a) the generation of pseudorandom bits from a preshared key seed and a hash function; and (b) XOR of the pseudorandom string from (a) and the reconciled key. We also prove that the hashing matrix in (a) can be reused. Then, Alice and Bob can generate pseudorandom bits offline. For real-time privacy amplification, they only need to perform the XOR operation in a bitwise manner. In the spirit of stream ciphers, the new scheme is conceptually different from the existing block privacy-amplification schemes. Such an essential difference guarantees the new scheme with the following practical features: (1) it can output final key bits in a stream way; (2) it will not spread the errors of the input bit strings; and (3) it can be carried out ahead of information reconciliation.

The rest of this paper is organized as follows. In Sec. II, we review QKD protocols and recap the security proof based on quantum error correction and its reduction to the prepare-and-measure case. In Sec. III, we reduce quantum phase error correction to a new privacy-amplification procedure with a stream output and introduce its possible combination with delayed privacy amplification, classical cryptography, and QRNG. Finally, we conclude the paper and discuss possible future directions in Sec. IV.

## II. PRELIMINARIES

### A. QKD protocols and security definition

Here, we introduce the first and probably the most well-known QKD protocol, BB84 [1], and its entanglement version, Bennett-Brassard-Mermin-1992 (BBM92) [29]. Then, we show how their security can be established.

The procedures of the BB84 protocol are listed in Table I. Alice and Bob have a quantum channel for state transmission and an authenticated classical channel for data postprocessing. In practice, photons are widely used as the information carrier in quantum communication. Various degrees of freedom of a photon can be used for qubit encoding. For example, the four BB84 states  $\{|0\rangle, |1\rangle, |+\rangle, |-\rangle\}$  can be encoded into four polarization states of a photon, namely, vertical, horizontal,  $45^\circ$ , and  $135^\circ$ , respectively.

When Alice prepares a qubit in the  $Z$  basis and Bob measures in the same basis, an error occurs when their bit values are different. The errors in the  $Z$  basis are called *bit errors*. Similarly, when they operate in the  $X$  basis, a *phase error* occurs when their bits are different. Let us denote the bit- and phase-error rates by  $e_b$  and  $e_p$ , respectively:

$$\begin{aligned} e_b &= \frac{\text{number of bit errors}}{n}, \\ e_p &= \frac{\text{number of phase errors}}{n}. \end{aligned} \quad (1)$$

TABLE I. BB84 protocol [1], a prepare-and-measure protocol.

---



---

- (1) *State preparation.* Alice randomly prepares qubits in one of the four states  $|0\rangle$ ,  $|1\rangle$ ,  $|+\rangle$ , and  $|-\rangle$ , where  $|0\rangle$  and  $|1\rangle$  form the  $Z$  basis and  $|\pm\rangle = (|0\rangle \pm |1\rangle)/\sqrt{2}$  form the  $X$  basis.
- (2) *State transmission and measurement.* Alice sends the encoded qubits to Bob through a quantum channel. Bob measures each received qubit in the  $Z$  or  $X$  basis randomly.
- (3) *Key sifting.* Alice and Bob announce their choices of bases publicly, through an authenticated classical channel. They keep only the bits where they use the same bases and discard the rest. They accumulate  $n$  sifted key bits.
- (4) *Key distillation.* Alice and Bob perform classical postprocessing, including information reconciliation and privacy amplification, to generate a secret key from the  $n$ -bit sifted key.

---



---

Measurements on the two bases are noncommuting. Due to quantum mechanics, with any attempt to extract nontrivial information from the qubits, Eve would inevitably introduce disturbance, such as errors, making  $e_b, e_p \neq 0$ . Intuitively, Alice and Bob share a private key if  $e_b = e_p = 0$ .

Before the security analysis, we introduce an entanglement-based protocol with EPR pairs, BBM92, in Table II. The BBM92 protocol can be reduced to the BB84 protocol if Alice measures her half of the state in the  $Z$  or  $X$  basis. Based on her measurement result, Alice equivalently sends a qubit in  $|0\rangle$ ,  $|1\rangle$ ,  $|+\rangle$ , or  $|-\rangle$  to Bob.

For the key-distribution task, Alice and Bob need to make sure that their key bit strings are *identical* and *uniformly random* from Eve’s point of view. To satisfy these two requirements, the ideal key state shared by Alice and Bob, and any outside adversary Eve, is defined to be the following classical-classical-quantum state:

$$\rho_{\text{ideal}} = 2^{-n} \sum_{k \in \{0,1\}^n} |k, k\rangle_{A,B} \langle k, k| \otimes \rho_E, \quad (2)$$

where systems  $A, B$  are keys held by Alice and Bob and system  $E$  is held by Eve. In the ideal key state, Alice’s and Bob’s key bit strings are identical. Eve’s system  $\rho_E$  is independent of the key  $k$ , which brings her no more information about the key string than a random guess. This definition follows the works of Ben-Or *et al.* [30] and Renner and König [31].

In practice, however, Alice and Bob cannot generate an ideal key. It is reasonable to allow for a small failure probability. That is, Alice and Bob can generate a key state that is very close to an ideal one. To put the idea into a rigorous

form, if the realistic key state

$$\rho_{\text{key}} = \sum_{k_A, k_B \in \{0,1\}^n} \Pr(k_A, k_B) |k_A\rangle_A \langle k_A| \otimes |k_B\rangle_B \langle k_B| \otimes \rho_E(k_A, k_B) \quad (3)$$

satisfies

$$\frac{1}{2} \min_{\rho_E} \|\rho_{\text{key}} - \rho_{\text{ideal}}\|_1 \leq \varepsilon, \quad (4)$$

the protocol will be  $\varepsilon$  secure. Here, the distance measure is the trace distance. For two density matrices  $\rho, \sigma$ , the measure is defined as

$$\frac{1}{2} \|\rho - \sigma\|_1 = \frac{1}{2} \sum_i |\lambda_i|, \quad (5)$$

where  $\lambda_i$  are eigenvalues of the operator  $\rho - \sigma$ . The choice of the trace-distance measure is that it satisfies the requirements of a composable-security framework [30,31].

**Definition 1:** (QKD  $\varepsilon$  security). A QKD protocol is  $\varepsilon$  secure if the generated state  $\rho_{\text{key}}$  given in Eq. (3) is  $\varepsilon$  close to the ideal key state  $\rho_{\text{ideal}}$  given in Eq. (2) with respect to the trace-distance definition.

This definition, usually referred to as *soundness*, guarantees that once the protocol is not aborted, the generated key is private with a high probability. In the composable-security framework, there is another security parameter, *completeness*—i.e., the protocol success probability—which guarantees that one protocol is not trivial and that it does not always abort. For most protocols, completeness can be easily established, so we do not discuss the completeness parameter in this work. To connect

TABLE II. BBM92 protocol [29], an entanglement version of BB84.

---



---

- (1) *State preparation.* Alice prepares EPR pairs,  $|\Phi^+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ . She stores one half of the pairs locally and sends the other half to Bob.
- (2) *State storage.* Upon receiving a qubit, Bob stores the state in quantum memories. If the qubit has been lost in the channel or the quantum storage fails, they discard the pair.
- (3) *Quantum error correction.* Alice and Bob measure a random sample of the stored qubit pairs to estimate the quantum bit- and phase-error rates,  $e_b$  and  $e_p$ . They apply quantum error correction to the remaining stored qubit pairs. They share  $n$  (almost) perfect EPR pairs.
- (4) *Key measurement.* Both Alice and Bob measure the EPR pairs in the local  $Z$  basis to obtain the final key.

---



---

the security definition and the EPR pairs, we employ the following lemma.

**Lemma 1:** (Lemma 1 in [18]). *If Alice and Bob share a quantum state  $\rho_{AB}$  that is  $\varepsilon_f$  close to the ideal key state before projective measurements onto  $|\Phi^+\rangle^{\otimes n}$ , i.e.,*

$$\langle \Phi^+ |^{\otimes n} \rho_{AB} | \Phi^+ \rangle^{\otimes n} \geq 1 - \varepsilon_f, \quad (6)$$

then the QKD protocol is  $\varepsilon$  secure with  $\varepsilon = \sqrt{\varepsilon_f(2 - \varepsilon_f)}$ .

Combining Lemma 1 and the security definition, we can conclude that if the state  $\rho_{AB}$  shared by Alice and Bob before the measurement of a QKD protocol satisfies Eq. (6) with a small  $\varepsilon_f$ , this protocol is  $\varepsilon$  secure. Therefore, the security of QKD is closely related to entanglement and the purpose of the security proof is to realize Eq. (6).

### B. Security analysis based on quantum error correction

In establishing the security analysis of QKD, important tools such as entanglement distillation [32] and quantum error correction have been proposed and developed. A profound discovery is the connection between key privacy and entanglement. The number of generated keys can be elegantly linked with distillable entanglement under local operations and classical communication [27]. One can distill almost perfect entanglement via quantum bit- and phase-error correction before measuring the quantum state to obtain the final key. Though security can be proven via entanglement distillation, one does not need to carry out this procedure in reality. Quantum bit- and phase-error correction can be carried out using classical means once the parameters are well estimated in the virtual quantum scenario [28]. In this framework, quantum bit-error correction corresponds to information reconciliation and quantum phase-error correction is transformed into privacy amplification. Below, we briefly introduce the security analysis based on quantum error correction [27,28]. In the following discussions, we call a state transmission and measurement that generates a pair of raw key bits a QKD *round*. After many rounds, Alice and Bob accumulate enough raw key bits as a *block* for postprocessing, which we call a QKD *session*.

Alice and Bob aim to establish a perfect Einstein-Podolsky-Rosen (EPR) pair  $|\Phi^+\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$  for each round, where  $|0\rangle$  and  $|1\rangle$  are the eigenstates of the Pauli operator  $\sigma_z$ . Due to channel disturbance or Eve's interference, the EPR pairs shared by Alice and Bob after  $n$  rounds of state transmission are usually imperfect. We denote the state of these data pairs as  $\rho_{AB}$ . The difference between  $\rho_{AB}$  and  $|\Phi^+\rangle\langle\Phi^+|^{\otimes n}$  can be seen as disturbance and can be characterized by the bit-error rate  $e_b$  and the

phase-error rate  $e_p$ :

$$\begin{aligned} e_b &= \frac{1}{2}(1 - \text{Tr}[\rho_{AB}(\sigma_z \otimes \sigma_z)^{\otimes n}]), \\ e_p &= \frac{1}{2}(1 - \text{Tr}[\rho_{AB}(\sigma_x \otimes \sigma_x)^{\otimes n}]), \end{aligned} \quad (7)$$

where  $\sigma_x$  is another Pauli operation and we take the  $Z$  basis for key generation. The values of  $e_b$  and  $e_p$  can be obtained by parameter estimation. Here, the bit- and phase-error rates defined in Eq. (7) are consistent with those in Eq. (1). Note that Eq. (1) defines error frequencies, while Eq. (7) defines error probabilities. Strictly speaking, these two definitions are only the same in the infinite-data-size limit,  $n \rightarrow \infty$ . In the finite-data-size regime, there is a deviation between them, caused by statistical fluctuations. For simplicity, we ignore the difference for the moment and we take it into account in the evaluation of the failure probability of quantum error correction.

If  $e_b = e_p = 0$ , this means that for the pair of qubits  $\rho$  in each round,

$$\text{Tr}[\rho(\sigma_z \otimes \sigma_z)] = 1, \quad (8)$$

$$\text{Tr}[\rho(\sigma_x \otimes \sigma_x)] = 1, \quad (9)$$

which indicates that  $\rho = |\Phi^+\rangle\langle\Phi^+|$ . To realize Eqs. (8) and (9), Alice and Bob can apply the quantum circuit shown in Fig. 1 to do the quantum bit- and phase-error correction. Quantum bit-error correction guarantees Eq. (8) and quantum phase-error correction guarantees Eq. (9).

The cost of quantum error correction comes from the ancillary EPR pairs used during the procedure. As an analog to the cost in classical error correction, the number of ancillary EPR pairs equals the number of parity check bits. Then, bit- and phase-error correction will cost  $t_b = nh(e_b)$  and  $t_p = nh(e_p)$  ancillary EPR pairs, respectively, where  $h(x) = -x \log(x) - (1-x) \log(1-x)$  is the binary entropy function. Throughout this work, all logarithms are base 2. These costs can be derived from the cost in classical error correction, the details of which we leave to Appendix A. Therefore, the net generation rate of EPR pairs is given by [28],

$$r = 1 - h(e_b) - h(e_p). \quad (10)$$

We can further reduce the procedure to a prepare-and-measure one by moving the final measurement to the front of quantum error correction. The reduction of quantum bit-error correction is straightforward, since all the operations can be done equivalently on classical data bits and it finally becomes information reconciliation. In contrast, the final measurement cannot be moved ahead of the quantum phase-error correction directly. Alice and Bob can replace the final measurements with a properly chosen joint measurement, which can be moved ahead of quantum



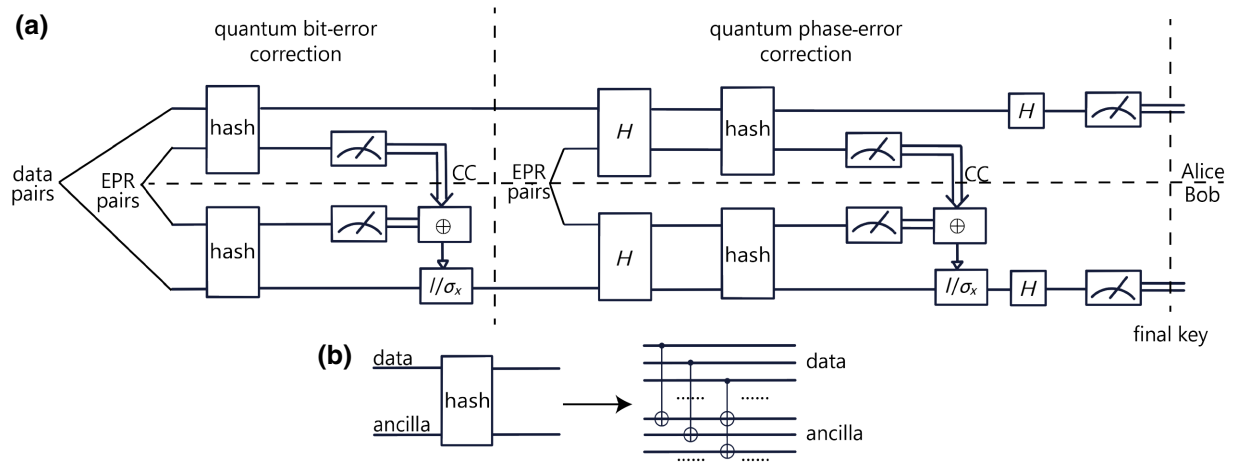


FIG. 1. (a) Quantum bit- and phase-error correction. The measurements in all the figures are  $Z$ -basis measurements by default. “CC” is short for classical communication. The  $\oplus$  operation means XOR operations on classical bit strings.  $H$  represents the Hadamard gate applied to each of the involved qubits.  $I/\sigma_x$  represents an identity or  $\sigma_x$  operation on the qubits, depending on the error syndrome. (b) In the linear case, the hash functions can be represented by matrices and realized by a series of controlled-NOT (CNOT) gates between the data (as control) and ancillary (as target) qubits. The measurement outcomes of ancillary qubits would give the parity information of the data qubits.

phase-error correction [28]. Then, the joint measurement becomes a classical operation called privacy amplification. Finally, quantum error correction is reduced to a “measurement + postprocessing” procedure. The reduction is in the spirit of the Shor-Preskill security proof [28]. We leave the details to Appendices C and D.

Due to the joint measurement, each final key bit depends on the measurement results from all the  $n$  data qubits. Hence, for privacy amplification, Alice and Bob need to wait for all the quantum states to be transmitted and measured in a QKD session. We call this *block* privacy amplification.

### III. STREAM PRIVACY AMPLIFICATION

#### A. Reduction of quantum error correction

In the aforementioned reduction, Alice and Bob cannot move the final key measurement ahead of phase-error correction directly. The main obstacle is that the Hadamard gate does not commute with the dephasing operation caused by the final key measurement,

$$\Delta_{Z^{\otimes n}}[\rho_A] = \sum_{\vec{k} \in \{0,1\}^n} |\vec{k}\rangle\langle\vec{k}| \rho_A |\vec{k}\rangle\langle\vec{k}|, \quad (11)$$

where  $\rho_A$  denotes the state that Alice holds and the dephasing operation is defined with respect to the key-measurement basis  $Z^{\otimes n}$  with outcomes  $\vec{k}$ . The operation on Bob’s side is similar. In the Shor-Preskill reduction, Alice and Bob essentially construct a joint  $Z$ -basis measurement that commutes with hash operations to circumvent this problem. As a result, this makes privacy amplification operate in blocks. Here, we rearrange the reduction of

the phase-error-correcting gates and keep the individual  $Z$ -basis measurements in the quantum phase-error correction. Consequently, we can render stream privacy amplification.

The key idea of the new reduction is to cancel all the Hadamard gates in quantum phase-error correction, shown in Fig. 1. The controlled-NOT (CNOT) gate in the circuit always appears in pairs on Alice’s and Bob’s sides. We focus on one pair of CNOT gates in the quantum phase-error-correction part, as depicted in Fig. 2(a). First, noting that  $H^2 = I$ , we add two consecutive Hadamard gates after each output qubit of the CNOT gate. Then, the four Hadamard gates before and after each CNOT gate exchange the roles of the control and target qubits,  $H^{\otimes 2} C_{\alpha\beta} H^{\otimes 2} = C_{\beta\alpha}$ , where  $C_{\alpha\beta}$  denotes a CNOT gate with control qubit  $\alpha$  and target qubit  $\beta$  and  $C_{\beta\alpha}$  is the other way around. For Bob’s data qubit, the phase-error-correcting operator  $I/\sigma_x$  becomes  $I/\sigma_z$  since  $\sigma_z = H\sigma_x H$ . Hence, we prove that circuits (a) and (b) in Fig. 2 are equivalent. Since the new phase-error-correcting operator  $I/\sigma_z$  does not affect the  $Z$ -basis measurement, this step can be skipped along with the error-syndrome measurements on the ancillary qubits. The remaining operations commute with the dephasing operation,  $\Delta_{Z^{\otimes n}}$ . Alice and Bob can add  $Z$ -basis measurements on ancillary qubits after the CNOT gates, since they are irrelevant at that point. Finally, they can move the final measurement ahead of quantum error correction, as shown in Fig. 2(c).

So far, we have only considered one CNOT gate. The hash operation in phase-error correction shown in Fig. 1 is composed of many CNOT gates. This reduction also works for the general hash-operation case. With this argument, by inserting consecutive Hadamard gates  $H^2 = I$  after each CNOT gate of the phase-error-correction part in Fig. 1, we

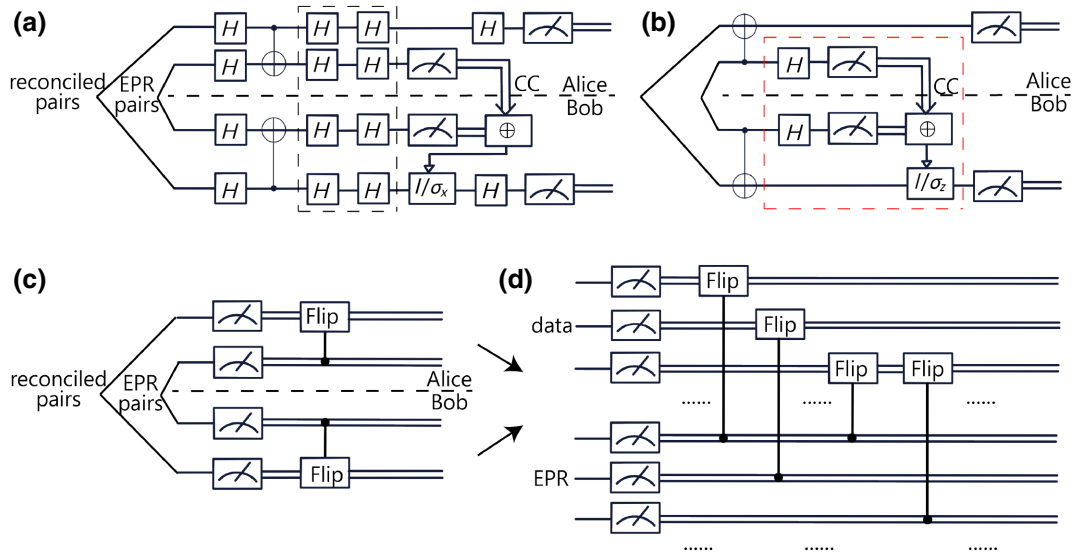


FIG. 2. Circuit (a) is derived from the quantum phase-error-correction phase in Fig. 1 by adding Hadamard gates in dashed boxes that form identity operations. We take one pair of CNOT operations for illustration. Circuit (b) is equivalent to circuit (a) by considering the following facts:  $H^{\otimes 2}C_{\alpha\beta}H^{\otimes 2} = C_{\beta\alpha}$ ,  $H^2 = I$ , and  $H\sigma_x H = \sigma_z$ . Since neither the identity nor the  $\sigma_z$  gate affects the Z-basis measurement, the operations in the dashed box of circuit (b) are redundant and can be removed. Then by moving the Z-basis measurement on ancillary qubits ahead of the hash operation and changing quantum-control flips to classical-control flips, circuit (b) turns into circuit (c), a “measurement + postprocessing” case. Circuit (d) shows the case of multiple CNOT pairs, taking the hashing circuit in Fig. 1(b) as an example. In the end, both Alice and Bob employ circuit (d) to obtain final key strings.

can reduce the whole quantum error-correction circuit to the “measurement + postprocessing” case, as shown in Fig. 2(d).

With the new reduction, the final key is determined by single-qubit measurements plus bit flips. The Z-basis measurement on the ancillary EPR pairs will provide Alice and Bob with a secure key *seed*. The bit flips are controlled by the seed and the hashing matrix. Then, the  $i$ th final key bit, extracted from the  $i$ th data qubit, is independent of the other data qubits. Hence, the new procedure can output the final key in a stream, i.e., the users can obtain a secure key bit once a pair of raw key bits is reconciled successfully between Alice and Bob. Following the term “stream cipher” in classical cryptography, we call this *stream* privacy amplification, as presented in Table III. The hashing matrix  $M$  in step (1) is the transpose of the original hashing matrix used in the quantum phase-error-correction phase of Fig. 1, because the original hashing matrix acts on  $X$  basis while  $M$  acts on the  $Z$  basis. The cost of stream privacy amplification lies in step

(2), where  $nh(e_p)$  preshared secure bits are consumed. The final key rate matches Eq. (10). Note that we consider the infinite-data-size limit here. The finite data-size effects due to statistical fluctuations have been well considered in the literature and are shown in Appendix A.

With the above deduction, the failure probability of privacy amplification  $\varepsilon$  is given by that of quantum phase-error correction. Given a phase-error pattern from a typical set determined by parameter estimation, a randomly chosen hash function can identify it with a high probability  $1 - \varepsilon$ . This is the reason why Alice and Bob need a random hashing matrix in step (1) of Table III. Details of error correction and its failure probability are presented in Appendix A.

Note that the phase-error pattern must be set before choosing the random hash function. That is, Eve cannot know the hashing matrix before Alice and Bob obtain the raw key bits. Naively, Alice can randomly pick up a matrix and send it to Bob via an authenticated but nonencrypted channel, as in the conventional case in block privacy

TABLE III. Stream privacy amplification.

After information reconciliation, denote Alice’s and Bob’s reconciled key as  $\vec{a} \in \{0, 1\}^n$ .

- (1) Alice and Bob randomly choose a hashing matrix  $M$  of size  $nh(e_p) \times n$ .
- (2) Alice and Bob use an  $nh(e_p)$ -bit seed,  $\vec{d} \in \{0, 1\}^{nh(e_p)}$ , to generate a pseudorandom string,  $\vec{d} \cdot M$ , where the dot product between the row vector and the matrix needs to take modulo 2 addition.
- (3) The final key is given by  $\vec{k} = \vec{d} \cdot M \oplus \vec{a}$ .

amplification. Since this public transmission of the hashing matrix must be done after quantum measurement, Alice and Bob have to wait for the whole block to be transmitted and measured. Then, they lose the “stream” property in privacy amplification.

To solve this problem, we apply a different approach, in which Alice and Bob generate an identical random hashing matrix locally with a preshared key and never reveal it in public. Then, they can prepare this matrix [step (1)] and the pseudorandom string [step (2)] before quantum transmission. A naive implementation of this approach, in which Alice and Bob generate  $M$  and  $\vec{d}$  in each run of the privacy amplification, could consume too many preshared secure bits, as for most of the universal hashing matrices, the number of random bits required to generate the matrix is larger than the data size  $n$ . Fortunately, with the following theorem, Alice and Bob can reuse the private hashing matrix in multiple QKD sessions with a failure probability that increases linearly, satisfying the composable-security definition [30,31]. Since the failure probability can be exponentially small, the same hashing matrix can be used for many QKD sessions. Therefore, the cost of generating this hashing matrix is shared with these sessions, making the average cost negligible.

**Theorem 1:** (Reuse of hashing matrix in privacy amplification). *Given a QKD session, the failure probability of a randomly chosen hashing matrix for privacy amplification is upper bounded by  $\epsilon$ . Then, for  $m$  QKD sessions, if Alice and Bob apply the same randomly chosen matrix for each session, the probability that privacy amplification fails in at least one session is upper bounded by  $m\epsilon$ .*

*Proof.* Following the aforementioned deduction of quantum error correction, the failure probability of privacy amplification is determined by phase-error correction. Now, the question becomes that given  $m$  phase-error patterns, if Alice and Bob randomly pick a hash function to correct all the errors, what is the failure probability that at least one phase-error pattern is unsuccessfully corrected? In Appendix B, we provide the failure probability of reusing hash functions for error correction, as given by Lemma 4. Using this result, the answer to the above question is  $m\epsilon$ . ■

Before running QKD sessions, Alice and Bob can perform steps (1) and (2) in Table III and prepare the pseudorandom string in advance. They only need to run step (3) in privacy amplification during real-time QKD, which is essentially composed of simple XOR operations and much faster than hash operations. In block privacy amplification, the computational complexity of the matrix multiplication with Toeplitz hashing is  $O(n \log n)$  with the fast-Fourier-transform algorithm, where  $n$  is the length of the reconciled

key string [23,33]. In contrast, the computational complexity of step (3) is  $n$  and hence stream privacy amplification is faster in real-time QKD, especially when the data size is large.

In reality, Alice and Bob need parameter estimation before privacy amplification. This might restrict the “stream” feature since the parameters cannot be accurately estimated until the whole data block is accumulated. Nevertheless, when the link between Alice and Bob is stable, they can foresee the parameters and apply them to stream privacy amplification. In practice, it is not difficult to maintain stability in a quantum communication network [16]. In addition, the users can double check the parameters after the transmission of the whole block. If the predicted parameters are within a reasonable range, they keep the key. Otherwise, if the actual parameters show that the implemented privacy amplification cannot guarantee security, this implies that the length of the seed chosen in step (2) of Table III is insufficient. Alice and Bob can then choose another seed, according to the difference between the predicted and actual parameters, to carry out additional privacy amplification on the key to make it secure.

Note that the stream scheme can work for any block size in QKD implementations. In order to make privacy amplification efficient, Alice and Bob can employ a large data size without causing delays in real-time key generation. Compared with the previous ones, the unique feature of the new scheme—stream output—can make QKD more practical in scenarios such as the satellite-to-ground link [8].

Moreover, the bit-error locations in the input string will remain the same after stream privacy amplification, since the final key bit is only determined by the pseudorandom bit and the raw key bit at the same location. As a result, the errors will not spread out and then privacy amplification can even be performed ahead of information reconciliation. This feature increases the flexibility of data postprocessing. For example, privacy amplification and information reconciliation can be performed in parallel. The recently proposed scenario of distributed private randomness distillation [34] is also a potential application of the new scheme.

## B. Application I: Enhancing the security of trusted-relay QKD network

The major issue of a trusted-relay QKD network lies in the trustworthiness of the intermediate nodes. There are some attempts to reduce the requirement on trusted relays, one of which is delayed privacy amplification [35,36]. In the normal case, all the QKD links between two end users, Alice and Bob, will generate secure keys between neighboring nodes. Then, the intermediate relays swap the keys by announcing the XOR results of two keys generated with two neighbors. In the delayed privacy-amplification case,

the relays swap the key right after information reconciliation. Then, Alice and Bob perform privacy amplification without the relays. In this case, they can eliminate the relays from the final key generation process. That is, the relays do not obtain the final key directly. Of course, if the relays listen to classical communication to obtain privacy-amplification matrices, they can still obtain the whole final key string. Therefore, the delayed privacy-amplification scheme only works for *honest but curious* relays.

Now, we can combine stream privacy amplification with delayed privacy amplification to further reduce the trustworthiness of the intermediate relays. After information reconciliation, all relays swap their keys by announcing the XOR results of two neighboring keys. Then, Alice and Bob will share a reconciled key string  $\vec{a}$ , which is also known to the relays. Note that Alice and Bob perform the steps in Table III locally. In particular, in step (2), they generate the pseudorandom string  $\vec{d} \cdot M$  privately. Hence, the relays cannot know the final key without  $\vec{d} \cdot M$ . If the relays want to learn the final key, they need to figure out  $\vec{d}$  and  $M$ . The seed  $\vec{d}$  is private and changes in every QKD session. The hashing matrix  $M$ , on the other hand, is reused for many sessions in stream privacy amplification, so the relays might figure it out from final and reconciled key strings in past sessions by methods such as differential cryptanalysis. These analysis methods often consume a lot of computational resources. For an even higher security level with fewer assumptions on the relays, we can add another layer of security based on the computational complexity on the intermediate nodes. In practice, this combined scheme further reduces the requirement of the trustworthiness of the relays and enhances the security of trusted-relay QKD networks.

### C. Application II: Information-theoretic toolbox for classical encryption analysis

There is an interesting property of the pseudorandom string  $\vec{d} \cdot M$  generated in step (2) of Table III. On the one hand, parameter estimation provides an upper bound on the information leakage of  $nh(e_p)$  bits about the raw key string  $\vec{a}$ . On the other hand, the security proof guarantees that the information leakage is removed via the simple XOR operation  $\vec{k} = \vec{d} \cdot M \oplus \vec{a}$ . This implies that  $\vec{d} \cdot M$  has at least an  $nh(e_p)$ -bit uncertainty to Eve. In the security analysis, we do not assume in advance which part of the  $nh(e_p)$ -bit information on  $\vec{a}$  is known by Eve. Then, for any  $nh(e_p)$  bits from  $\vec{d} \cdot M$ , the corresponding  $nh(e_p)$ -bit substring will be uniformly distributed from Eve's point of view, as shown in Fig. 3. This property is called  $k$ -wise independence in classical cryptography and is an essential requirement of many stream ciphers [37].

The above observation inspires a new tool for the security analysis of classical encryption based on quantum

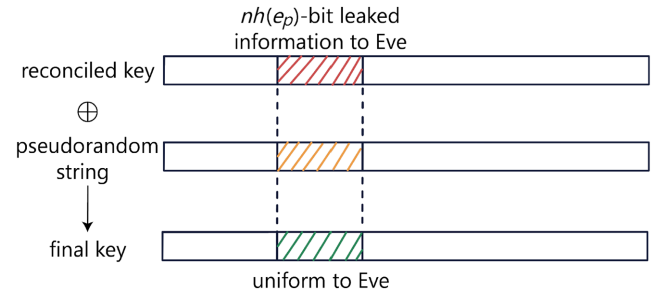


FIG. 3. An illustration of the pseudorandomness property of  $\vec{d} \cdot M$ . The red-shaded area in the reconciled key denotes the  $nh(e_p)$ -bit information leakage to Eve. This leaked information can be the bit values of the key or the parity information about the key. After the XOR of the reconciled key and the pseudorandom string  $\vec{d} \cdot M$ , the reconciled key becomes the final secure key and is uniformly distributed from Eve's point of view. That is, Eve's knowledge about the key is removed. As a corollary, the corresponding yellow-shaded area in the pseudorandom string should be uniform to Eve as well.

phase-error correction. First, we note that the hash function for phase-error correction is not necessarily linear. Alice and Bob can employ an arbitrarily nonlinear code to extend the seed to a pseudorandom string in step (2) of Table III, as shown in Fig. 4(a). This extension operation can be treated as a pseudorandom number generator. Second, we can further generalize this stream privacy amplification as a joint function of the reconciled key and the seed, as shown in Fig. 4(b). The joint operation can be treated as a classical encryption box. Alice and Bob can employ sophisticated schemes here, such as the advanced encryption standard (AES) and lattice-based encryption algorithms. Third, we can express the classical operation in the quantum form, as a joint operation  $\Lambda$  acting on data and ancillary qubits, as shown in Fig. 4(c). Since the operation is classical,  $\Lambda$  commutes with the dephasing operation in Eq. (11):  $\Lambda[\Delta_{Z^{\otimes n}}(\rho)] = \Delta_{Z^{\otimes n}}[\Lambda(\rho)]$ . By definition,  $\Lambda$  is a dephasing incoherent operation (DIO) [38]. At last, we can move  $\Lambda$  ahead of measurement. Since phase-error correction is virtual in the security analysis, Alice and Bob can add an extra operation  $\Omega$  on the ancillary qubits if necessary, as shown in Fig. 4(d).

Similar to stream privacy amplification, we can analyze the security of the final output of Fig. 4(d) by going back to the Lo-Chau security proof based on quantum error correction. The DIO  $\Lambda$  is determined by the classical encryption algorithm, which is viewed as the encoding for a quantum error-correcting code. With this code, Alice and Bob exchange the measurement results of ancillary qubits as a phase-error syndrome, which should give information about the phase errors of data qubits. According to the security analysis, Alice and Bob do not need to actually correct phase errors since this will not affect the final key. Then, Alice and Bob can add an extra virtual operation  $\Omega$



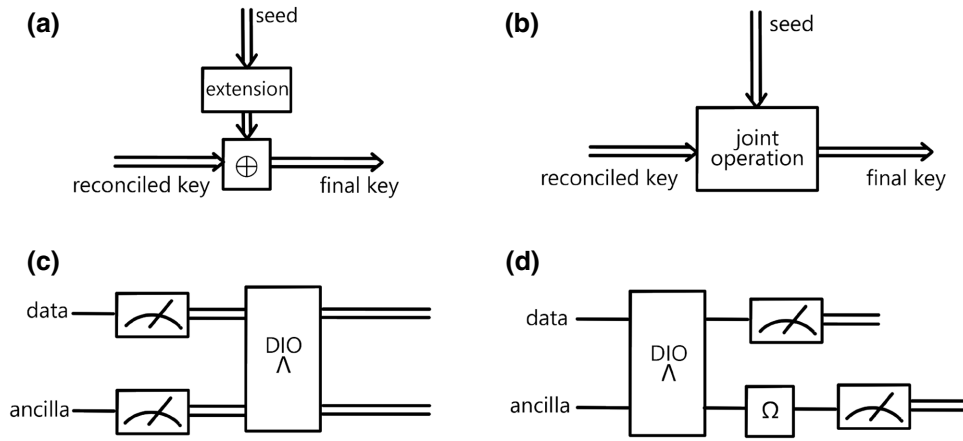


FIG. 4. (a) A schematic diagram for stream privacy amplification. The process of extending the seed can be treated as a pseudorandom number generator in step (2) of Table III. (b) The generalization of the extension and XOR operations as a joint operation, which can be seen as a classical encryption box. (c) The quantum form of (b). The classical joint operation becomes a DIO  $\Lambda$ . (d) Changing the order of  $\Lambda$  and measurement. The final measurement on ancillary qubits can be arbitrary with an extra operation  $\Omega$ .

on the ancillary qubits between  $\Lambda$  and measurement. They can optimize  $\Omega$  to maximize the error-correction capability led by  $\Lambda$ . The security of the final output of the classical encryption will be determined by the phase-error-correction capability of the corresponding code. Since we consider the most general attacks in QKD, the method provides a generic information-theoretic analysis tool for classical encryption algorithms.

**D. Application III: Stream randomness extraction**

Besides QKD, privacy amplification also plays a vital role in many other quantum cryptographic tasks, such as QRNG. In general, the raw data generated from a practical QRNG system are not uniformly random. Due to device imperfection, some information about the raw data might even be leaked and might lead to potential security loopholes. Thus, user Alice needs to apply randomness extraction to the raw data. By definition, randomness extraction is essentially the same as privacy amplification. As a result, the stream privacy-amplification technique can also be directly applied to QRNG—stream randomness extraction.

In QRNG, the amount of intrinsic randomness in the raw data is usually quantified in terms of min-entropy [39]. This randomness measure can be converted to the number

of phase errors in a virtual quantum error-correction protocol [40]. Then, one can convert common block randomness extraction to a stream manner, as presented in Table IV.

For the data postprocessing of practical QRNG, stream randomness extraction can be a favored choice. In this context, Alice can fully characterize the quantum devices in use and hence has good empirical knowledge of the randomness generation rate. In other words, the min-entropy of output randomness can be well predicted in advance. With this property, steps (1) and (2) in Table IV can be done separately in advance without access to the raw data. This enables the main steps of postprocessing to be carried out in parallel with the generation of raw data, which can reduce the storage requirements and modularize quantum random number generators. In addition, the computational complexity of real-time postprocessing [step (3)] is only  $n$  and hence helps to solve the current bottleneck of real-time random number generation—slow extraction.

**IV. DISCUSSION AND CONCLUSIONS**

In this work, we propose a stream privacy-amplification scheme, where Alice and Bob locally generate a pseudorandom bit string and XOR it with the reconciled key to obtain the final key. This scheme has a stream output feature and hence can prevent unpleasant delay and error spreading in practice. In contrast to conventional

TABLE IV. Stream randomness extraction.

<p>Alice generates a raw bit string from the QRNG device, denoted as <math>\vec{a} \in \{0, 1\}^n</math>, the min-entropy of which is <math>H_{\min}</math>.</p> <p>(1) Alice randomly chooses a hashing matrix <math>M</math> of size <math>(n - H_{\min}) \times n</math>.</p> <p>(2) Alice uses an <math>(n - H_{\min})</math>-bit seed, <math>\vec{d} \in \{0, 1\}^{n-H_{\min}}</math>, to generate a pseudorandom string, <math>\vec{d} \cdot M</math>, where the dot product between the row vector and the matrix needs to take modulo 2 addition.</p> <p>(3) The final random bit string is given by <math>\vec{k} = \vec{d} \cdot M \oplus \vec{a}</math>.</p>
-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

block schemes, stream privacy amplification can be carried out ahead of information reconciliation, which makes the data postprocess more flexible. In addition, stream privacy amplification can enhance the security of a trusted-relay QKD network and improve the practicality of randomness extraction for quantum number generators.

We need to emphasize that although we reduce the stream privacy amplification from the Lo-Chau security proof, the technique is independent of security proofs. Other security-proof methods, such as Koashi's complementarity approach [41], can also be easily extended to the stream privacy-amplification case. Moreover, the concept is rather generic and can be applied to other QKD schemes, such as the six-state, continuous-variable, measurement-device-independent, two-way communication postprocessing, and decoy-state schemes [5]. The practical issues, including realistic circumstances, hardware imperfections, and statistic fluctuations, will affect the parameter settings of stream privacy amplification, especially the length of the seed string and the size of the hashing matrix. One can combine with existing analysis methods to deal with these practical issues. In Appendix E, we give an example of employing stream privacy amplification in the Gottesman-Lo-Lütkenhaus-Prekill (GLLP) framework [42]. The further applications of stream privacy amplification in other quantum cryptographic tasks such as quantum oblivious transfer [43,44] are also worth studying.

Here, our proof is mainly based on phase-error correction. According to Ref. [40], in general, this approach is equivalent to the one based on the quantum leftover hashing lemma [45]. An interesting direction is to reconsider the new scheme from the entropic approach point of view.

Our security analysis provides a new perspective to examine classical encryption algorithms information theoretically through quantum-information theories. Rigorous assessment of classical encryption algorithms, such as AES and lattice-based encryption, is often a formidable challenge. To the best of our knowledge, there has been little consideration to date in the context of the information-theoretic study of these encryption algorithms.

## ACKNOWLEDGMENTS

We thank Guang Yang, Guoding Liu, Pei Zeng, and Hongyi Zhou for the helpful discussions. This work was supported by the National Natural Science Foundation of China under Grants No. 11875173 and No. 12174216 and by the National Key Research and Development Program of China under Grants No. 2019QY0702 and No. 2017YFA0303903.

## APPENDIX A: ERROR CORRECTION

We briefly review error correction and derive its cost in this section. Suppose that Alice and Bob have two

$n$ -bit strings,  $\vec{x}, \vec{y} \in \{0, 1\}^n$ , before error correction. The differences between these two strings are the errors to be corrected, represented by an error string  $\vec{e} = \vec{x} \oplus \vec{y}$ . Bob aims to reconcile his bit string to Alice's. For this purpose, the essential task is to locate all the errors, i.e., to figure out the error string  $\vec{e}$ . Denote each bit in an error string as a binary random variable,  $E_i \in \{0, 1\}$ . An error in the  $i$ th bit is represented by  $E_i = 1$ . The error string is the joint random variable of  $\{E_i\}$ , denoted  $\vec{E}$ . We use the corresponding lowercase letters to denote the specific realizations of random variables. All the possible error strings form a linear space  $\Omega$  with a size of  $|\Omega| = 2^n$ . Denote the probability that an error string  $\vec{e} \in \Omega$  occurs by  $p(\vec{e})$ . In general, the joint random variable is not independent and identically distributed (IID).

Before error correction, Alice and Bob need to estimate the set of possible errors. In parameter estimation, we allow a failure probability  $\varepsilon \geq 0$ . With the probability of  $1 - \varepsilon$ , the error string will fall into the  $\varepsilon$ -smallest probable set  $\mathcal{T}_{\vec{E}}^\varepsilon$ . The aim of parameter estimation is to upper bound the cardinality of the probable error set, or error cardinality, via the statistics of all measurements in an experiment [18].

Given the error cardinality  $|\mathcal{T}_{\vec{E}}^\varepsilon|$ , Alice and Bob need to exchange a certain amount of parity information to correct the errors. There exists an error-correction protocol to remove all errors by exchanging an amount of parity-check bits upper bounded by

$$I_{\text{EC}} = \log |\mathcal{T}_{\vec{E}}^\varepsilon| - \log \varepsilon_{\text{EC}}, \quad (\text{A1})$$

where  $\varepsilon_{\text{EC}} \geq 0$  is the failure probability of error correction.

In the rest of this appendix, we show how to derive Eq. A1 and calculate the cost  $I_{\text{EC}}$  associated with the error rate. Let us start with the definition of the  $\varepsilon$ -smallest probable set.

**Definition 2:** ( $\varepsilon$ -smallest probable set). Given  $0 \leq \varepsilon < \frac{1}{2}$ , a random variable  $X \in \mathcal{X}$  that has a finite range  $|\mathcal{X}| < \infty$  and probability mass function  $p_X$ , the  $\varepsilon$ -smallest probable set of  $X$  is defined as the smallest set,  $\mathcal{T}_X^\varepsilon$ , such that a realization of  $X$  lies in it with a failure probability no larger than  $\varepsilon$ ,

$$\mathcal{T}_X^\varepsilon = \arg \min_{\mathcal{S}} |\mathcal{S}|$$

$$\text{such that } \Pr(X \in \mathcal{S}) \geq 1 - \varepsilon. \quad (\text{A2})$$

To tackle the cardinality of the  $\varepsilon$ -smallest probable set, one method is to use the weight of error strings. For an  $n$ -bit string  $\vec{e} \in \{0, 1\}^n$ , we define a weight function, quantifying the number of 1s in the string,

$$\text{wt}(\vec{e}) = \sum_{i=1}^n e_i. \quad (\text{A3})$$

We can bound the cardinality of a set via bounding the weights of the strings, as shown in the following lemma.

**Lemma 2:** *Given constants  $c \geq 0$ ,  $0 < r < 1$ , and  $n \in \mathbb{Z}^+$ , the cardinality of the  $n$ -bit-string set,*

$$\mathcal{D}(c) = \{\vec{e} \in \{0, 1\}^n \mid wt(\vec{e}) \in \mathcal{C}\}, \quad (\text{A4})$$

$$\mathcal{C} = \begin{cases} [0, nr + c], & r \leq \frac{1}{2}, \\ [nr - c, n], & r > \frac{1}{2}, \end{cases} \quad (\text{A5})$$

can be upper bounded by

$$|\mathcal{D}(c)| < 2^{nh(r)+c \left| \log \frac{r}{1-r} \right|}. \quad (\text{A6})$$

*Proof.* When  $r = 1/2$ , the bound in Eq. (A6) is trivial. The case for  $r > \frac{1}{2}$  is the same as the one for  $r < \frac{1}{2}$  by switching the definitions of bits 0 and 1, so we only need to prove the lemma for  $r < \frac{1}{2}$ .

We introduce a function of a binary variable  $e \in \{0, 1\}$ ,

$$p(e) = \begin{cases} r, & e = 1, \\ 1 - r, & e = 0, \end{cases} \quad (\text{A7})$$

and we can show that

$$\sum_{\vec{e} \in \{0,1\}^n} 2^{\sum_{i=1}^n \log p(e_i)} = \prod_{i=1}^n [p(e_i = 0) + p(e_i = 1)] = 1. \quad (\text{A8})$$

For a bit string  $\vec{e} = (e_1, \dots, e_n)$  with a weight of  $wt(\vec{e}) = k$ , we define a function

$$g(k) = k \log r + (n - k) \log(1 - r) = \sum_{i=1}^n \log [p(e_i)]. \quad (\text{A9})$$

For  $\vec{e} \in \mathcal{D}(c)$ , we have  $k \in [0, nr + c]$  and  $g(k)$  is a decreasing function of  $k$  when  $r < \frac{1}{2}$ , so

$$\begin{aligned} g(k) &\geq g(nr + c) \\ &= (nr + c) \log r + (n - nr - c) \log(1 - r) \\ &= -nh(r) - c \left| \log \frac{r}{1-r} \right|. \end{aligned} \quad (\text{A10})$$

Then, combining Eqs. (A8)–(A10), we have

$$\begin{aligned} 1 &= \sum_{\vec{e} \in \{0,1\}^n} 2^{\sum_{i=1}^n \log p(e_i)} \\ &= \sum_{\vec{e} \in \{0,1\}^n} 2^{g(k)} \end{aligned}$$

$$\begin{aligned} &\geq \sum_{\vec{e} \in \mathcal{D}(c)} 2^{g(k)} \\ &\geq 2^{-nh(r)-c \left| \log \frac{r}{1-r} \right|} |\mathcal{D}(c)|, \end{aligned} \quad (\text{A11})$$

where the third and fourth inequalities cannot take equal signs simultaneously. Finally, we obtain Eq. (A6). ■

Because the weight of an  $n$ -bit string  $\vec{e}$  satisfies  $0 \leq wt(\vec{e}) \leq n$ , one can combine the two intervals of Eq. (A5) in the lemma to obtain a smaller set, as given in the following corollary.

**Corollary 1:** *Given constants  $c \geq 0$ ,  $0 < r < 1$ , and  $n \in \mathbb{Z}^+$ , the cardinality of the  $n$ -bit-string set,*

$$\mathcal{D}(c) = \{\vec{e} \in \{0, 1\}^n \mid wt(\vec{e}) \in [nr - c, nr + c]\}, \quad (\text{A12})$$

can be upper bounded by

$$|\mathcal{D}(c)| < 2^{nh(r)+c \left| \log \frac{r}{1-r} \right|}. \quad (\text{A13})$$

We remark that the result holds even when the random variables,  $E_i$ , associated with the set  $\mathcal{D}(c)$  are arbitrarily correlated and not necessarily IID. Alice and Bob can use Lemma 2 to bound the cardinality of the  $\varepsilon$ -smallest probable set if they can estimate  $r$  and  $c$ . Before error correction, they can estimate the error rate with a failure probability of  $\varepsilon$ , say, via the random sampling method. Suppose that they obtain an error frequency of  $r$  in the test samples. Without loss of generality, we assume that  $r \leq 1/2$ . In the asymptotic case  $n \rightarrow \infty$ , the number of errors in the data is given by  $nr$ . With a finite data size, the rate fluctuates around  $r$ . Alice and Bob can bound the number of errors,  $wt(\vec{e}) \leq nr + c$ , via the random sampling method, where  $c/n$  represents the deviation of the error rate from the test samples. The deviation  $c/n$  is usually related to the failure probability of parameter estimation  $\varepsilon$  and typically has an order of  $1/\sqrt{n}$  [18]. Using  $r$ ,  $c$ , and  $n$ , they can apply Lemma 2 to determine the error cardinality.

Here, we introduce another method to upper bound the cardinality of typical error sets, which is tighter under more restricted conditions [18].

**Lemma 3:** ([18]). *Given constants  $c, r \geq 0$  and  $n \in \mathbb{Z}^+$  satisfying  $r + c/n \leq 1/3$ , the cardinality of the  $n$ -bit-string set,*

$$\mathcal{D}(c) = \{\vec{e} \in \{0, 1\}^n \mid wt(\vec{e}) \in [0, nr + c]\}, \quad (\text{A14})$$

can be upper bounded by

$$|\mathcal{D}(c)| < 2^{n \cdot h(r + \frac{c}{n})}. \quad (\text{A15})$$

*Proof.* By definition, we have

$$\begin{aligned} |\mathcal{D}(c)| &= \sum_{0 \leq k < nr+c} \binom{n}{k} \\ &< \binom{n}{\lceil nr+c \rceil} \\ &\leq 2^{n \cdot h(r + \frac{c}{n})}. \end{aligned} \quad (\text{A16})$$

The first inequality holds when  $r \leq 1/3$ .  $\blacksquare$

This lemma provides a tight bound because in Eq. (A16),

$$\binom{n}{\lceil nr+c \rceil - 1} \leq \sum_{0 \leq k < nr+c} \binom{n}{k} < \binom{n}{\lceil nr+c \rceil}, \quad (\text{A17})$$

and the last inequality in Eq. (A16) is also tight in the logarithm sense.

When the error rate deviation is small,  $c/n \ll 1$ , we can take the Taylor-series expansion of  $h(r + c/n)$  at  $r$  and we can obtain

$$\begin{aligned} h\left(r + \frac{c}{n}\right) &= h(r) + h'(r) \frac{c}{n} + \frac{h''(r)}{2!} \left(\frac{c}{n}\right)^2 + \dots \\ &= h(r) + \frac{c}{n} \log \frac{1-r}{r} + \frac{h''(r)}{2!} \left(\frac{c}{n}\right)^2 + \dots \end{aligned} \quad (\text{A18})$$

By comparing Eqs. (A6), (A15), and (A18), one can see that Lemma 2 is a first-order approximation of Lemma 3. Since  $h''(r) < 0$ , Lemma 3 provides a tighter bound than Lemma 2. On the other hand, Lemma 2 does not require  $r \leq 1/3$ , so it can be applied to more general cases.

Now, we show how to locate the errors given the probable error set  $\mathcal{T}^\varepsilon$ , where  $\varepsilon$  is the failure probability for parameter estimation,  $\sum_{\vec{e} \in \mathcal{T}^\varepsilon} p(\vec{e}) \geq 1 - \varepsilon$ . In the following, we introduce error correction based on universal hash functions.

**Definition 3:** (Universal hash family). A family of functions  $\mathcal{F}$  mapping elements  $\vec{e}$  in a space  $\mathcal{T}$  to another space  $\mathcal{S}$  is called a universal hash family if the probability of a randomly chosen hash function outputting the same hashing result for any two different strings is upper bounded by

$$\forall \vec{e}_i \neq \vec{e}_j \in \mathcal{T}, \Pr_{f \in \mathcal{F}} [f(\vec{e}_i) = f(\vec{e}_j)] \leq \frac{1}{|\mathcal{S}|}. \quad (\text{A19})$$

Given the probable error set  $\mathcal{T}^\varepsilon$ , Alice and Bob decide a universal hash family  $\mathcal{F}$  and choose a hash function  $f \in \mathcal{F}$  randomly. Alice applies the hash function to her bit string  $\vec{x}$  and sends the hashing result  $\vec{s}_A = f(\vec{x})$  to Bob, who calculates the syndrome  $\vec{s} = f(\vec{e})$  based on  $\vec{s}_A, \vec{y}$ , and  $f$ . If the

hash functions are linear, then

$$\begin{aligned} \vec{s} &= \vec{s}_A \oplus \vec{s}_B \\ &= f(\vec{x}) \oplus f(\vec{y}) \\ &= f(\vec{x} \oplus \vec{y}) \\ &= f(\vec{e}). \end{aligned} \quad (\text{A20})$$

The linear hash function is normally implemented by multiplication of a hashing matrix.

If there is only one error string  $\vec{e} \in \mathcal{T}^\varepsilon$  that satisfies  $f(\vec{e}) = \vec{s}$ , then in principle Bob can figure out  $\vec{e}$  correctly. The probability of Bob figuring out a wrong error string is upper bounded by

$$\begin{aligned} \sum_{\vec{e}' \in \mathcal{T}^\varepsilon \setminus \{\vec{e}\}} \Pr_{f \in \mathcal{F}} [f(\vec{e}') = f(\vec{e})] &\leq \sum_{\vec{e}' \in \mathcal{T}^\varepsilon \setminus \{\vec{e}\}} \frac{1}{|\mathcal{S}|} \\ &< \frac{|\mathcal{T}^\varepsilon|}{|\mathcal{S}|} \\ &= 2^{-(I_{\text{EC}} - k)} \\ &\equiv \varepsilon_{\text{EC}}, \end{aligned} \quad (\text{A21})$$

where  $\mathcal{T}^\varepsilon \setminus \{\vec{e}\}$  denotes the subset of  $\mathcal{T}^\varepsilon$  excluding the element  $\vec{e}$ , the first inequality comes from Eq. (A19), the second inequality comes from  $|\mathcal{T}^\varepsilon \setminus \{\vec{e}\}| = |\mathcal{T}^\varepsilon| - 1$ ,  $I_{\text{EC}} = \log |\mathcal{S}|$  is the effective length of syndrome  $\vec{s}$ , and  $k = \log |\mathcal{T}^\varepsilon|$  is the logarithm of error cardinality. The failure probability  $\varepsilon_{\text{EC}}$  approaches zero exponentially with respect to  $I_{\text{EC}} - k$ . If  $I_{\text{EC}} - k$  is large enough, the failure probability will become negligible, such that Bob can almost certainly figure out  $\vec{e}$  and then perform error correction according to the syndrome. We emphasize that  $\varepsilon_{\text{EC}}$  is the failure probability for error correction and is different from the failure probability  $\varepsilon$  for parameter estimation. We can derive Eq. (A1) from Eq. (A21) by taking the logarithm,

$$-I_{\text{EC}} - \log |\mathcal{T}^\varepsilon| = \log \varepsilon_{\text{EC}}. \quad (\text{A22})$$

The cardinality of  $\mathcal{T}^\varepsilon$  is given by Lemma 2. Then, we have

$$\begin{aligned} \frac{1}{n} I_{\text{EC}} &= \frac{1}{n} \log |\mathcal{T}^\varepsilon| - \frac{1}{n} \log \varepsilon_{\text{EC}} \\ &< h(r) + \frac{c}{n} \left| \log \frac{r}{1-r} \right| - \frac{1}{n} \log \varepsilon_{\text{EC}} \\ &\xrightarrow{n \rightarrow \infty} h(r). \end{aligned} \quad (\text{A23})$$

In the third line, we take the asymptotic limit,  $c/n \rightarrow 0$ , because normally in parameter estimation  $c$  has an order



of  $1/\sqrt{n}$ , and  $1/n \log \varepsilon_{\text{EC}} \rightarrow 0$  since  $\varepsilon_{\text{EC}}$  is a constant. This gives Shannon's source-encoding theorem in classical information theory.

### APPENDIX B: REUSE OF HASH FUNCTION

In the derivation of Eq. (A21), we implicitly assume that errors cannot depend on the choice of the hash function. This requirement can be further understood in an adversary scenario, where the error string  $\vec{e}$  is determined by an adversary, Eve. In this case, Eve does not know the hash function chosen by Alice and Bob before she fixes the error string  $\vec{e}$  or before Alice and Bob obtain their bit strings  $\vec{x}$  and  $\vec{y}$ , respectively. Otherwise, with prior knowledge of the hash function, Eve can choose the error string craftily such that it lies outside the error space that the chosen hash function can handle. As a result, Alice and Bob cannot figure out the correct error string  $\vec{e}$  and the error correction will fail. To guarantee this requirement, there are two possible ways for Alice and Bob to decide the hash function:

- (1) Alice and Bob decide the hash function after the error string is fixed, i.e., after the bit strings  $\vec{x}, \vec{y}$  are obtained. Then, they exchange some random bits through a public channel to determine a hash function randomly.
- (2) Alice and Bob decide the hash function randomly by consuming some preshared private randomness and keep the hash function secret from Eve. In later error correction, they can reuse the same matrix at a cost, namely that the total failure probability increases linearly with the number of sessions. We state the result formally in Lemma 4.

**Lemma 4:** *Given a set of error strings  $\mathcal{T}^\varepsilon$  and a family of hash functions  $\mathcal{F}$ ,  $\forall \vec{e} \in \mathcal{T}^\varepsilon$ , suppose that the failure probability of a randomly chosen hash function to identify  $\vec{e}$  is upper bounded by  $\varepsilon_{\text{EC}}$ . For any  $m$  error strings in  $\mathcal{T}^\varepsilon$ , the failure probability of a randomly chosen hash function to simultaneously identify all the  $m$  error strings in each session is upper bounded by  $m\varepsilon_{\text{EC}}$ .*

*Proof.*  $\forall \vec{e} \in \mathcal{T}^\varepsilon$ , the failure probability of error correction is given by Eq. (A21):

$$\Pr_{f \in \mathcal{F}} [\exists \vec{e}' \in \mathcal{T}^\varepsilon \setminus \{\vec{e}\}, s.t., f(\vec{e}') = f(\vec{e})] \leq \varepsilon_{\text{EC}}, \quad (\text{B1})$$

where the probability is defined in the hashing family. Then, the failure probability of identifying  $m$  error strings simultaneously is given by

$$\Pr_{f \in \mathcal{F}} [\exists \vec{e}' \in \mathcal{T}^\varepsilon, s.t., \vec{e}' \neq \vec{e}_1, f(\vec{e}') = f(\vec{e}_1) \text{ or } \vec{e}' \neq \vec{e}_2, f(\vec{e}') = f(\vec{e}_2) \text{ or } \dots \text{ or } \vec{e}' \neq \vec{e}_m, f(\vec{e}') = f(\vec{e}_m)]$$

$$\begin{aligned} &\leq \sum_{i=1}^m \Pr_{f \in \mathcal{F}} [\exists \vec{e}' \in \mathcal{T}^\varepsilon \setminus \{\vec{e}_i\}, s.t., f(\vec{e}') = f(\vec{e}_i)] \\ &\leq m\varepsilon_{\text{EC}}, \end{aligned} \quad (\text{B2})$$

where the first inequality follows the union bound and  $\vec{e}_i \in \mathcal{T}^\varepsilon$  is the error string in the  $i$ th session. ■

Here, we note that Eve does not know the hash function  $f$  before she determines the error patterns  $\vec{e}_i$ . That is, her choices of  $\vec{e}_i$  should be independent of Alice's and Bob's choice of  $f \in \mathcal{F}$ . Otherwise, the failure-probability bound might not hold. Interestingly, this is not the case if Alice and Bob can verify the leftover errors in corrected strings, say, by exchanging an authentication tag [18]. Then, the failure probability is determined by the error-verification process but not the property of the hashing family. In this case, the hash function can be fixed at the beginning and known to Eve.

### APPENDIX C: QUANTUM ERROR CORRECTION WITH HASHING

As mentioned in the main text, we can use Eq. (6) to further derive Eqs. (8) and (9), which are related to the bit and phase error, respectively, and these two equations can be achieved by quantum error correction. We divide the procedure for quantum error correction into two steps: bit-error correction to guarantee Eq. (8) and phase-error correction to guarantee Eq. (9).

Quantum error correction can be seen as an extension of classical error correction and accomplished by using universal hashing [46–48]. In the classical case, Alice and Bob each possess a bit string. The differences between these two strings are called “errors.” The main job of classical bilateral error correction is to figure out the error locations. Alice hashes her string and sends the parity information to Bob. With the same hash function, Bob hashes his string and compares it to Alice's. After enough iterations of this procedure, Bob can figure out the error locations and flip his corresponding bits to correct the errors. In the end, Bob's bit string is reconciled with Alice's. The number of parity-check bits is given by  $nh(e)$  in the Shannon limit, where  $n$  is the bit-string length and  $e$  is the error rate. For the case of a finite data size, there is the possibility that error correction will fail. Details of error correction along with analysis of the finite size effect and the failure probability are presented in Appendix A.

In the following discussions, we consider linear hash functions, which can be represented by hashing matrices, for simplicity. One of the most widely used linear hash families is the family of Toeplitz matrices. The elements in a Toeplitz matrix  $M$  satisfy  $\forall i - j = i' - j', M_{ij} = M_{i'j'}$  and there are  $m + n - 1$  free bits in a Toeplitz matrix of size  $m \times n$ . We give an example of the Toeplitz hashing

matrix as follows:

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 & 0 & \dots \\ 0 & 1 & 0 & 1 & 1 & \dots \\ 0 & 0 & 1 & 0 & 1 & \dots \\ \vdots & & & \ddots & & \dots \end{pmatrix}_{nh(e) \times n}. \quad (C1)$$

As shown before, if Alice and Bob randomly choose a Toeplitz matrix of size  $nh(e) \times n$  for error correction, the efficiency converges to the Shannon limit very quickly when  $n$  is large.

Now, we can apply classical error correction to the quantum case. Let us start with quantum bit-error correction. The parity hashing on the raw key bit strings can be implemented by a series of CNOT gates between the data qubits and ancillary EPR pairs, as shown in Fig. 5, which is a concrete example of the quantum bit-error-correction part in Fig. 1(a). Alice and Bob can obtain the parity information by measuring the ancillary qubits. The measurement result of one ancillary qubit will reflect 1 bit of parity information of the data qubits. The measurement results on Alice’s and Bob’s ancillary pair will be different if there is an odd number of errors in these control qubits and the results will be the same if there is no error or an even number of errors. Alice sends the parity information to Bob, who then figures out the error syndromes and corrects the errors. The property of the universal hash family guarantees that Bob can correct all the errors with a small failure probability.

A similar approach can be employed for quantum phase-error correction, with additional Hadamard gates before hash operations and measurements.

In entanglement distillation, both bit- and phase-error correction should be successfully implemented. The two quantum error-correction procedures are carried out sequentially. Hence, we need to make sure that these two error-correction procedures do not interfere with each other. Fortunately, by using ancillary EPR pairs, we can decouple these two steps using the following lemma.

**Lemma 5:** (Bit- and phase-error-correction decoupling [49]). *By using EPR pairs as ancillary qubits, bit-error correction has no effect on phase errors and vice versa.*

*Proof.* Let us first show that the phase-error-measurement results are the same with or without the bit-error-correcting operations. The phase error is evaluated when both Alice and Bob perform the  $X$ -basis measurement on the data qubits, denoted by the measurement of the joint observable  $\sigma_x \otimes \sigma_x$ . From Fig. 1(a), we can see that the bit-error-correcting operations on the data qubits are essentially  $I$ ,  $\sigma_x$ , and  $\sigma_z$ . The operations  $I$  and  $\sigma_x$  will not change the  $X$ -basis measurement outcomes. The operation  $\sigma_z$  comes from the CNOT gate between the data and ancillary qubits. Since Alice’s and Bob’s hash functions are the same, the CNOT gates always appear in pairs. That is, if there is a CNOT gate between Alice’s share of a data-qubit pair and

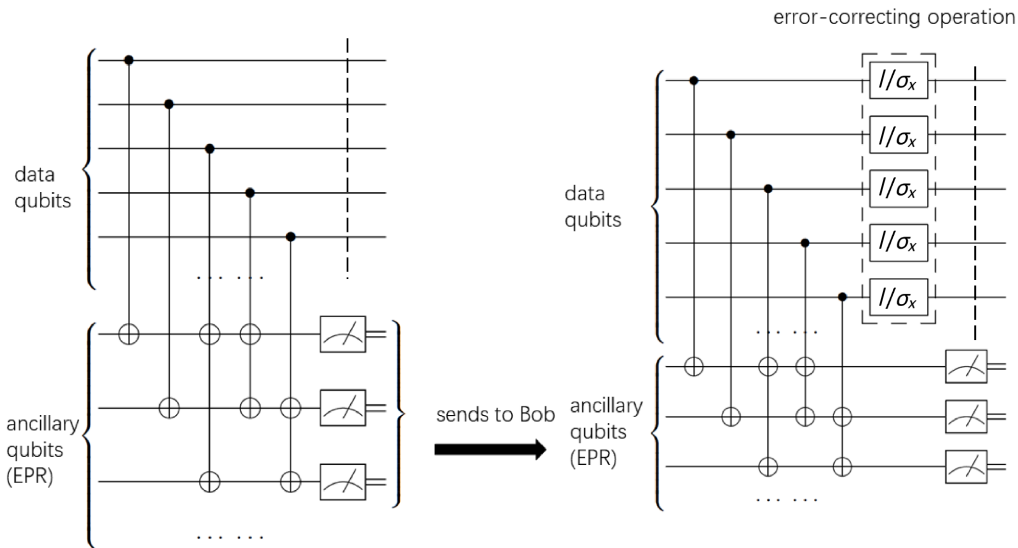


FIG. 5. An illustration of quantum bit-error correction. As an example, the circuit corresponds to the hashing matrix  $M$  in Eq. (C1). The  $i$ th row of  $M$  corresponds to the CNOT control data qubits targeting on the  $i$ th ancillary qubit and the  $j$ th column of  $M$  corresponds to the CNOT target ancillary qubits controlled by the  $j$ th data qubit. The measurement result of one ancillary qubit equals the XOR sum of the  $Z$ -basis measurement results of the ancillary qubit and its controlling data qubits. By comparing the measurement results, Bob can learn the 1-bit parity information of the data qubits. After knowing enough parity information on the data qubits, Bob can locate the bit errors and correct them with the quantum gate  $\sigma_x$ . A similar circuit can be applied for quantum phase-error correction, with additional Hadamard gates according to Fig. 1(a).

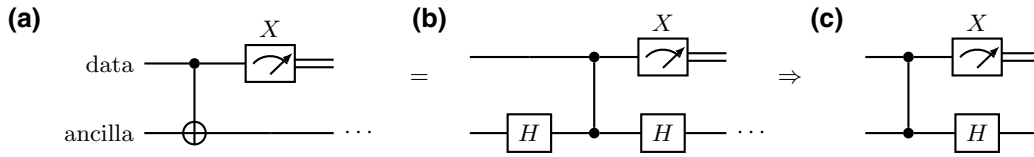


FIG. 6. A diagram showing the reason why bit-error correction does not change phase errors. In the showcase, we take a pair of CNOT gates as an example. In the figures, we only depict the circuit on Alice’s side and the circuit is the same on Bob’s side. Here, the control qubit is one of the data qubits and the target qubit is one of the ancillary qubits, as shown in Fig. 5. The equivalence of (a)–(c) comes from the fact that  $\sigma_x = H\sigma_zH$  and  $H^{\otimes 2}|\Phi^+\rangle = |\Phi^+\rangle$ .

an ancillary EPR pair, there will be a CNOT gate between Bob’s share of the data-qubit pair and the ancillary EPR pair. From the circuit equivalency shown in Fig. 6, we can see that the  $\sigma_z$  operation always appears in pairs on Alice’s and Bob’s data qubits. That is, the  $\sigma_z$  operation will simultaneously flip Alice’s and Bob’s  $X$ -basis measurement results, which leaves the measurement results of  $\sigma_x \otimes \sigma_x$  on the data-qubit pairs unchanged. Therefore, quantum bit-error correction does not affect phase errors.

With the duality between the  $X$  and  $Z$  bases, using the same arguments, we can also prove that the phase-error correction will not affect bit errors. ■

**APPENDIX D: REDUCTION FROM QUANTUM OPERATIONS TO CLASSICAL ONES**

Now, we want to reduce the quantum operations to classical ones, mainly following the Shor-Prekill security proof [28]. By classical, we mean that Alice and Bob can equivalently perform the operations after key measurement, which then become classical data processing on key bits. The key idea is to move the final measurement ahead of quantum error correction. Then, bit-error correction becomes classical bilateral error correction and phase-error correction becomes privacy amplification. To do so, we need to make sure that the dephasing operation  $\Delta_{Z^{\otimes n}}$  in Eq. (11) caused by the final  $Z$ -basis measurement on each side commutes with all the quantum error-correction operations.

Looking back at Fig. 1(a), it is not difficult to see that bit-error correction only consists of  $I$ ,  $\sigma_x$ , and CNOT gates.

These quantum operations commute with the dephasing operation  $\Delta_{Z^{\otimes n}}$ . Therefore, Alice and Bob can change the order of  $Z$ -basis measurement and bit-error correction. By doing so, quantum operations are replaced with corresponding classical operations on the measurement results, as shown in Fig. 7. For example, the  $\sigma_x$  gate becomes a flip on classical bits and the CNOT gate becomes a classical control-flip operation. Finally, quantum bit-error correction becomes classical bilateral error correction, a typical information reconciliation method.

Quantum phase-error correction is more complicated because it contains one more operation, the Hadamard gate, which does not commute with the dephasing operation,  $\Delta_{Z^{\otimes n}}$ . Here, we follow the idea of the Shor-Prekill security proof to reduce phase-error correction into classical privacy amplification [28].

First, in the quantum circuit shown in Fig. 1(a), we combine the Hadamard gate together with the adjacent  $Z$ -basis key-generation measurement, which becomes the  $X$ -basis measurement. The phase-error-correcting operation on Bob’s side is essentially  $\sigma_x$ , which does not affect the result of the  $X$ -basis measurement. Therefore, neither the correction operation nor the phase-error-syndrome measurement is necessary and the circuit of quantum phase-error correction becomes the one shown in Fig. 8(a).

Second, since Alice’s and Bob’s positions are now symmetric, we focus on Alice’s side. We show that she can replace the individual key measurement with a series of joint  $X$ -basis measurements and still obtain an  $[n - nh(e_p)]$ -bit secure final key, as shown in Fig. 8(b). The joint measurement can be represented by an  $n$ -bit vector

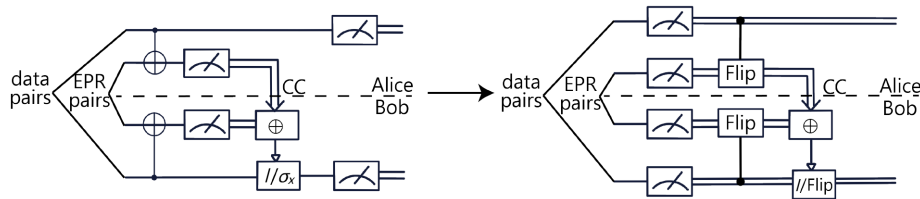


FIG. 7. The transformation from quantum bit-error correction to information reconciliation. The left-hand circuit is the quantum bit-error-correction part in Fig. 1. The “Flip” gate means that Alice and Bob flip the corresponding classical bits. Bob calculates the error syndrome using his parity information and that sent by Alice and then flips his erroneous bits to reconcile the key.

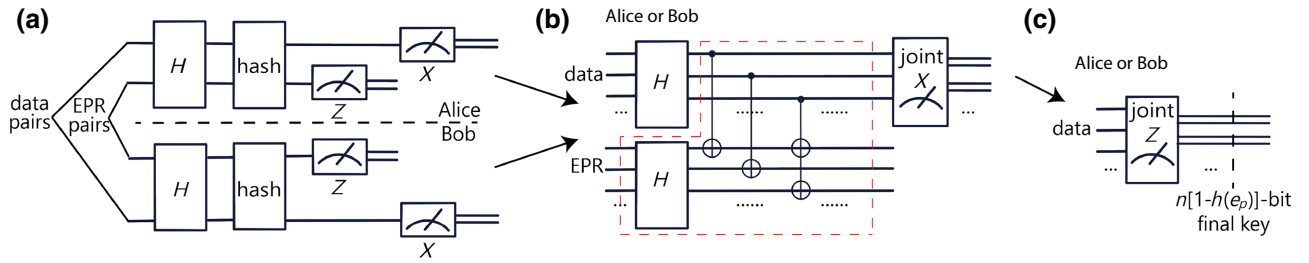


FIG. 8. (a) The reduced circuit from the phase-error-correction part in Fig. 1 by considering the following two facts: Hadamard+Z-basis measurement =  $X$ -basis measurement; neither the identity nor the  $\sigma_x$  gate affects the  $X$ -basis measurement. Now, Alice's and Bob's circuits become the same. (b) The reduction of Alice's circuit: remove the measurement on the ancillary qubits since the results do not affect the measurement on data qubits; replace individual  $X$ -basis measurements with joint  $X$ -basis measurements on data qubits; and explicitly express the hash operation with the CNOT gates shown in Fig. 1(b) as an example. If the joint  $X$ -basis measurements commute with the hash operation, the circuit in the red dashed box does not affect the measurements and hence can be removed. (c) Further reduction of Alice's circuit: remove the redundant circuit in the red dashed box and combine joint  $X$ -basis measurements with the Hadamard gates, which become joint  $Z$ -basis measurements. Finally, Alice can obtain an  $n[1 - h(e_p)]$ -bit secure key from the joint  $Z$ -basis measurements in a QKD session.

$\vec{v}$ , which measures the observable

$$O_{\vec{v},x} = \bigotimes_{i=1}^n \sigma_x^{v_i}, \quad (\text{D1})$$

where the  $v_i$  are the element values of  $\vec{v}$ . Here, we require  $\vec{v}$ s to be linearly independent, so that the final key is still secure after phase-error correction. Note that since the key measurements differ from the ones in Fig. 8(a), the obtained key bits may be different.

The effect of hash operations on the  $n$  data qubits is either identity when the measurement outcome of the ancillary qubit is 0 or a series of  $n$ -qubit operations consisting of  $\sigma_z$  and  $I$  when the outcome is 1. For example, the hash operation corresponding to the first row vector  $\vec{m}_1$  of the matrix in Eq. (C1) is an identity or

$$O_{\vec{m}_1,z} = \sigma_z \otimes I \otimes \sigma_z \otimes \sigma_z \otimes I \cdots \quad (\text{D2})$$

In general, we represent the operation corresponding to the  $i$ th row vector  $\vec{m}_i$  of the matrix  $M$  as follows:

$$O_{\vec{m}_i,z} = \bigotimes_{j=1}^n \sigma_z^{M_{ij}}, \quad (\text{D3})$$

where  $M_{ij}$  is the value of the element in the  $i$ th row and the  $j$ th column of the matrix.

Normally, the operators in Eqs. (D1) and (D3) do not commute because  $[\sigma_x, \sigma_z] \neq 0$ . Fortunately, we have  $[\sigma_x \otimes \sigma_x, \sigma_z \otimes \sigma_z] = 0$ . To make sure that a joint  $X$ -basis measurement commutes with a series of  $Z$  operations, we only need to design the joint  $X$ -basis measurement such that the number of qubits that are applied with the  $\sigma_z$  operation and measured in the  $X$  basis is even. That is,  $\vec{m}_i \cdot \vec{v} = 0$  holds,  $\forall i \in \{1, 2, 3, \dots, nh(e_p)\}$ . This is equivalent to finding the

kernel of  $M$ . The kernel of the matrix  $M$  is defined as the set of vectors such that

$$\ker M = \{\vec{v} : M\vec{v} = \vec{0}\}. \quad (\text{D4})$$

In phase-error correction, the hashing matrix that we use has a full rank. Therefore, the rank of  $\ker M$  is  $n - nh(e_p)$ , or,  $\ker M$  can be constructed from  $[n - nh(e_p)]$  linearly independent vectors. We arrange these vectors in columns and form a matrix,  $V$ , which is of size  $n \times [n - nh(e_p)]$ . We call the matrix  $V$  the dual matrix of  $M$ . Then, we can design joint  $X$ -basis measurements according to this dual matrix, where each joint measurement corresponds to a column vector of  $V$  according to the correspondence in Eq. (D1). By construction, these joint measurements all commute with the hash operation.

Finally, the commutation property shows that the hash operation will not affect the results of the joint  $X$ -basis measurements. Then, Alice and Bob can remove the hash operation along with the ancillary qubits. The joint  $X$ -basis measurements can be further combined with the Hadamard gate and become the joint  $Z$ -basis measurements, as shown in Fig. 8(c). The measurement results of the joint  $Z$ -basis measurements give an  $[n - nh(e_p)]$ -bit secret key.

With the reduction, one can see that the ancillary EPR pairs are unnecessary for phase-error correction, as shown in Fig. 8(c). Does that mean that there is no cost of shared private randomness in phase-error correction? Unfortunately, the answer is no. The cost is reflected in the joint  $Z$ -basis measurement. The number of final key bits is determined by the number of joint  $Z$ -basis measurements, which is limited by the kernel of the hashing matrix for phase-error correction.

Meanwhile, the joint  $Z$ -basis measurement is compatible with bit-error-correction reduction. After considering the cost of shared private randomness in bit-error correction,



the net gain of the secret key is  $n[1 - h(e_b) - h(e_p)]$  bits, which matches the key rate in Eq. (10).

Here are a few notes on information reconciliation and privacy amplification:

- (1) We take the family of Toeplitz matrices as an example here. The decoding for such random-Toeplitz-matrix hashing is computationally hard in practice. To bypass the hardness in information reconciliation, more practical error-correcting codes, such as the low-density parity-check code [50,51], are used. For privacy amplification, Alice and Bob can still adopt random-Toeplitz-matrix hashing, as the decoding is unnecessary for phase-error correction.
- (2) In general, the failure probability of error correction  $\varepsilon$  is a property of the hash family. But if Alice and Bob can perform error verification, the failure probability of error correction is determined by the verification process instead [18]. In this case, the hash function can be prefixed and known to Eve at the beginning. This is the case for information reconciliation. Unfortunately, there is no way to verify the result of privacy amplification, because the virtual quantum phase-error correction is not implemented in real life.
- (3) The reduction from quantum phase-error correction to privacy amplification is different from Shor-Preskill's original argument, where the Calderbank-Shor-Steane quantum error-correcting code is adopted.

### APPENDIX E: GLLP FRAMEWORK

In practical implementation, the devices may deviate from the ideal assumptions in the security analysis. The actual error rate may also vary from the prior estimation from random sampling due to statistical fluctuations. We need to take these issues into account in the security analysis, where they affect the secure key rate given in Eq. (10). Thus, in practice, we also need to modify the parameter settings of the stream privacy-amplification scheme accordingly. Here, we take the GLLP framework as an example to show how the stream scheme can be combined with existing approaches to handle the practical issues.

In Ref. [42], Gottesman *et al.* have established a general framework for security analysis with realistic devices. In this framework, Alice and Bob characterize their devices to quantify the deviation from the ideal case. For this purpose, Alice and Bob can perform a virtual measurement on the devices and then, for each round, they tag the sifted key bit as “good” if the virtual measurement projects to the ideal case and “bad” if it is the orthogonal case. More generally, Alice and Bob can label sifted key bits with an arbitrary tag  $g$  and derive the corresponding phase-error rate  $e_p^g$ . Then,

we can obtain the extended GLLP key-rate formula [52],

$$r \geq -h(e_b) + \sum_g q_g \left[ 1 - h(e_p^g) \right], \quad (\text{E1})$$

where  $e_b$  is the bit-error rate and  $\{q_g\}$  gives the ratio of the sifted key bits with tag  $g$ , satisfying  $\sum_g q_g = 1$  and  $q_g \geq 0, \forall g$ . The first term on the right-hand side of Eq. (E1) represents the cost of information reconciliation and the second term corresponds to the ratio of privacy amplification.

With this formula, we can directly employ stream privacy amplification to the GLLP framework. Here, we still consider the case of performing information reconciliation first and then privacy amplification in the postprocessing. We denote the length of the sifted key string as  $n$ . If Alice and Bob perform encryption in information reconciliation, then in stream privacy amplification, the seed length is  $n \sum_g q_g h(e_p^g)$  and the size of the hashing matrix is  $\left[ n \sum_g q_g h(e_p^g) \right] \times n$ . As for the case where Alice and Bob do not perform encryption in information reconciliation, the seed length will be  $n \left[ h(e_b) + \sum_g q_g h(e_p^g) \right]$  and the size of the hashing matrix will be  $\left[ nh(e_b) + n \sum_g q_g h(e_p^g) \right] \times n$  in stream privacy amplification. The remaining steps are the same as those in Table III.

As we can see from this example, the new scheme is highly portable. Given a QKD protocol in the presence of practical issues, once the users can analyze the amount of information leakage and obtain a good estimation, they can carry out privacy amplification in a stream way. Thus, the stream scheme is adequate to deal with practical issues.

- 
- [1] C. H. Bennett and G. Brassard, in *Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing* (IEEE Press, New York, 1984), p. 175.
  - [2] A. K. Ekert, Quantum Cryptography Based on Bell's Theorem, *Phys. Rev. Lett.* **67**, 661 (1991).
  - [3] C. H. Bennett, G. Brassard, and J.-M. Robert, Privacy amplification by public discussion, *SIAM J. Comput.* **17**, 210 (1988).
  - [4] U. Maurer, Secret key agreement by public discussion from common information, *IEEE Trans. Inf. Theory* **39**, 733 (1993).
  - [5] F. Xu, X. Ma, Q. Zhang, H.-K. Lo, and J.-W. Pan, Secure quantum key distribution with realistic devices, *Rev. Mod. Phys.* **92**, 025002 (2020).
  - [6] J.-P. Chen, C. Zhang, Y. Liu, C. Jiang, W. Zhang, X.-L. Hu, J.-Y. Guan, Z.-W. Yu, H. Xu, and J. Lin, *et al.*, Sending-or-Not-Sending with Independent Lasers: Secure Twin-Field Quantum Key Distribution over 509 km, *Phys. Rev. Lett.* **124**, 070501 (2020).

- [7] X.-T. Fang, P. Zeng, H. Liu, M. Zou, W. Wu, Y.-L. Tang, Y.-J. Sheng, Y. Xiang, W. Zhang, and H. Li, *et al.*, Implementation of quantum key distribution surpassing the linear rate-transmittance bound, *Nat. Photon.* **14**, 422 (2020).
- [8] S.-K. Liao, W.-Q. Cai, J. Handsteiner, B. Liu, J. Yin, L. Zhang, D. Rauch, M. Fink, J.-G. Ren, and W.-Y. Liu, *et al.*, Satellite-Relayed Intercontinental Quantum Network, *Phys. Rev. Lett.* **120**, 030501 (2018).
- [9] N. T. Islam, C. C. W. Lim, C. Cahall, J. Kim, and D. J. Gauthier, Provably secure and high-rate quantum key distribution with time-bin qudits, *Sci. Adv.* **3**, 00 (2017).
- [10] C. Elliott, A. Colvin, D. Pearson, O. Pikalo, J. Schlafer, and H. Yeh, in *Quantum Information and Computation III (International Society for Optics and Photonics, 2005)*, vol. 5815, p. 138.
- [11] M. Peev, C. Pacher, R. Alléaume, C. Barreiro, J. Bouda, W. Boxleitner, T. Debuisschert, E. Diamanti, M. Dianati, and J. F. Dynes, *et al.*, The SECOQC quantum key distribution network in Vienna, *New J. Phys.* **11**, 075001 (2009).
- [12] D. Stucki, M. Legre, F. Buntschu, B. Clausen, N. Felber, N. Gisin, L. Henzen, P. Junod, G. Litzistorf, and P. Monbaron, *et al.*, Long-term performance of the SwissQuantum quantum key distribution network in a field environment, *New J. Phys.* **13**, 123001 (2011).
- [13] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, and A. Tanaka, *et al.*, Field test of quantum key distribution in the Tokyo QKD Network, *Opt. Express* **19**, 10387 (2011).
- [14] T.-Y. Chen, H. Liang, Y. Liu, W.-Q. Cai, L. Ju, W.-Y. Liu, J. Wang, H. Yin, K. Chen, and Z.-B. Chen, *et al.*, Field test of a practical secure communication network with decoy-state quantum cryptography, *Opt. Express* **17**, 6540 (2009).
- [15] S. Wang, W. Chen, Z.-Q. Yin, Y. Zhang, T. Zhang, H.-W. Li, F.-X. Xu, Z. Zhou, Y. Yang, and D.-J. Huang, *et al.*, Field test of wavelength-saving quantum key distribution network, *Opt. Lett.* **35**, 2454 (2010).
- [16] Y.-A. Chen, Q. Zhang, T.-Y. Chen, W.-Q. Cai, S.-K. Liao, J. Zhang, K. Chen, J. Yin, J.-G. Ren, and Z. Chen, *et al.*, An integrated space-to-ground quantum communication network over 4,600 kilometres, *Nature* **589**, 214 (2021).
- [17] J. Qiu, Quantum communications leap out of the lab, *Nat. News* **508**, 441 (2014).
- [18] C.-H. F. Fung, X. Ma, and H. F. Chau, Practical issues in quantum-key-distribution postprocessing, *Phys. Rev. A* **81**, 012318 (2010).
- [19] S.-K. Liao, J. Lin, J.-G. Ren, W.-Y. Liu, J. Qiang, J. Yin, Y. Li, Q. Shen, L. Zhang, and X.-F. Liang, *et al.*, *Chinese Phys. Lett.* **34**, eid090302 (pages 0) (2017).
- [20] Y. Liu, Q. Zhao, M.-H. Li, J.-Y. Guan, Y. Zhang, B. Bai, W. Zhang, W.-Z. Liu, C. Wu, and X. Yuan, *et al.*, Device-independent quantum random-number generation, *Nature* **562**, 548 (2018).
- [21] C. Bennett, G. Brassard, C. Crépeau, and U. Maurer, Generalized privacy amplification, *IEEE Trans. Inf. Theory* **41**, 1915 (1995).
- [22] X. Ma, F. Xu, H. Xu, X. Tan, B. Qi, and H.-K. Lo, Postprocessing for quantum random-number generators: Entropy evaluation and randomness extraction, *Phys. Rev. A* **87**, 062327 (2013).
- [23] M. Hayashi and T. Tsurumaru, More efficient privacy amplification With less random seeds via dual universal hash function, *IEEE Trans. Inf. Theory* **62**, 2213 (2016).
- [24] T.-Y. Chen, X. Jiang, S.-B. Tang, L. Zhou, X. Yuan, H. Zhou, J. Wang, Y. Liu, L.-K. Chen, and W.-Y. Liu, *et al.*, Implementation of a 46-node quantum metropolitan area network, *NPJ Quantum Inf.* **7**, 134 (2021).
- [25] H. Zhou, K. Lv, L. Huang, and X. Ma, *IEEE/ACM Trans Netw* p. 1 (2022).
- [26] N. Lütkenhaus and X. Ma, *System and Method for Quantum Key Distribution* (2016), U.S. Patent 9,294,272.
- [27] H. K. Lo and H. F. Chau, Unconditional security of quantum key distribution over arbitrarily long distances, *Science* **283**, 2050 (1999).
- [28] P. W. Shor and J. Preskill, Simple Proof of Security of the BB84 Quantum Key Distribution Protocol, *Phys. Rev. Lett.* **85**, 441 (2000).
- [29] C. H. Bennett, G. Brassard, and N. D. Mermin, Quantum Cryptography without Bell's Theorem, *Phys. Rev. Lett.* **68**, 557 (1992).
- [30] M. Ben-Or, M. Horodecki, D. W. Leung, D. Mayers, and J. Oppenheim, in *Proceedings of the Second International Conference on Theory of Cryptography* (Springer-Verlag, Berlin, Heidelberg, 2005), TCC'05, p. 386.
- [31] R. Renner and R. König, in *Proceedings of the Second International Conference on Theory of Cryptography* (Springer-Verlag, Berlin, Heidelberg, 2005), TCC'05, p. 407.
- [32] C. H. Bennett, D. P. DiVincenzo, J. A. Smolin, and W. K. Wootters, Mixed-state entanglement and quantum error correction, *Phys. Rev. A* **54**, 3824 (1996).
- [33] X. Ma, Z. Zhang, and X. Tan, (2011), arXiv preprint [ArXiv:1109.6147](https://arxiv.org/abs/1109.6147).
- [34] D. Yang, K. Horodecki, and A. Winter, Distributed Private Randomness Distillation, *Phys. Rev. Lett.* **123**, 170501 (2019).
- [35] C.-H. F. Fung, X. Ma, H. F. Chau, and Q.-y. Cai, Quantum key distribution with delayed privacy amplification and its application to the security proof of a two-way deterministic protocol, *Phys. Rev. A* **85**, 032308 (2012).
- [36] W. Stacey, R. Annabestani, X. Ma, and N. Lütkenhaus, Security of quantum key distribution using a simplified trusted relay, *Phys. Rev. A* **91**, 012338 (2015).
- [37] N. Alon, O. Goldreich, J. Håstad, and R. Peralta, Simple constructions of almost  $k$ -wise independent random variables, *Rand. Struct. Alg.* **3**, 289 (1992).
- [38] A. Streltsov, G. Adesso, and M. B. Plenio, Colloquium: Quantum coherence as a resource, *Rev. Mod. Phys.* **89**, 041003 (2017).
- [39] X. Ma, X. Yuan, Z. Cao, B. Qi, and Z. Zhang, Quantum random number generation, *npj Quantum Inf.* **2**, 1 (2016).
- [40] T. Tsurumaru, Equivalence of three classical algorithms with quantum side information: Privacy amplification, error correction, and data compression, *IEEE Trans. Inf. Theory* **68**, 1016 (2022).
- [41] M. Koashi, Simple security proof of quantum key distribution based on complementarity, *New J. Phys.* **11**, 045018 (2009).

- [42] D. Gottesman, H.-K. Lo, N. Lütkenhaus, and J. Preskill, Security of quantum key distribution with imperfect devices, *Quantum Info. Comput.* **4**, 325 (2004).
- [43] C. H. Bennett, G. Brassard, C. Crépeau, and M.-H. Skubiszewska, in *Advances in Cryptology—CRYPTO '91*, edited by J. Feigenbaum (Springer-Verlag, Berlin, Heidelberg, 1992), p. 351.
- [44] C.-Y. Wei, X.-Q. Cai, B. Liu, T.-Y. Wang, and F. Gao, A generic construction of quantum-oblivious-key-transfer-based private query with ideal database security and zero failure, *IEEE Trans. Comput.* **67**, 2 (2017).
- [45] R. Renner, Security of quantum key distribution, *Int. J. Quantum Inf.* **6**, 1 (2008).
- [46] G. Brassard and L. Salvail, in *Advances in Cryptology EUROCRYPT'93* (1994), vol. 765, p. 410.
- [47] N. Lütkenhaus, Estimates for practical quantum cryptography, *Phys. Rev. A* **59**, 3301 (1999).
- [48] X. Ma, C.-H. F. Fung, J.-C. Boileau, and H. Chau, Universally composable and customizable post-processing for practical quantum key distribution, *Comput. Secur.* **30**, 172 (2011).
- [49] H.-K. Lo, Method for decoupling error correction from privacy amplification, *New J. Phys.* **5**, 36 (2003).
- [50] R. Gallager, Low-density parity-check codes, *IRE Trans. Inf. Theory* **8**, 21 (1962).
- [51] D. J. MacKay, Good error-correcting codes based on very sparse matrices, *IEEE Trans. Inf. Theory* **45**, 399 (1999).
- [52] X. Ma, Ph.D. thesis, Department of Physics, University of Toronto (2008), also available in [ArXiv:0808.1385](https://arxiv.org/abs/0808.1385).

*Correction:* A proof change request for Eq. (11) was not implemented properly and has been rectified.