Speedy Error Reconciliation

No Author Given

No Institute Given

Abstract. Although the introduction of small errors in the lattice-based key exchange protocol can increase the strength of the protocol against quantum computing attacks, it will result in only approximately equal secret values between the negotiating parties, which cannot be used for subsequent secure communication. In order to ensure that both communicating parties reach a consensus, scholars propose to use error reconciliation mechanisms to eliminate errors in key negotiation. In this paper, a new error reconciliation mechanism, called Speedy Error Reconciliation (SER), is proposed to efficiently complete key agreement while ensuring the correctness and security of the key. SER can negotiate a two-bit shared key by simultaneously extracting information from the most and least significant bits of a coefficient. In particular, SER only needs to transmit the signal value of the most significant bit to achieve the negotiation of both the most and the least significant bits. By sharing g-bit auxiliary information between the two parties, SER expands the fault tolerance interval and reduces the failure rate during reconciliation. We integrate SER into key exchange protocols based on the learning with errors (LWE), the ring LWE (RLWE) and the module LWE (MLWE) problems, such as Frodo and NewHope, to test the generality and practical performance of SER under different difficult problems. By comparing parameters such as failure rate, security strength, and the number of CPU cycles required for total and per-bit keys, we find that SER performs well in various modes, especially in RLWE-based protocol. This is attributed to the fact that the RLWE-based key exchange scheme reduces the ratio of the error to the modulus q in SER by choosing a larger parameter q, which in turn reduces the failure rate. Compared with the original protocols, after replacing the error reconciliation mechanism with SER, the negotiation efficiency of the per-bit key in replaced Frodo and Newhope is increased by 61.6% and 797.6%, respectively.

Keywords: Post Quantum · Key Exchange · Error Reconciliation.

1 Introduction

Two entities, over an insecure channel, expect to leverage ephemeral secret materials to negotiate a shared session key for the subsequent secure communication. This process is called the key exchange (KE) protocol. In traditional network communications, the Diffie-Hellman (DH) key exchange protocol [1] and its variants are commonly used and embedded in Transport Layer Security (TLS) and

IP Security (IPSec) to establish secure communications on the Internet. However, advances in computational efficiency have fundamentally weakened the security of cryptographic algorithms, allowing potential attackers to use more powerful computing systems and the best cryptanalysis algorithms to increase the speed of their attacks. One aspect of this comes from Moore's Law, which predicts that computing systems will become more powerful than ever, allowing ever-larger brute force attacks. Quantum computing [2–4] and its algorithms are expected to further weaken the strength of existing cryptography and its applications. Some quantum algorithms are designed to solve the prime factorization of large prime numbers and the discrete logarithm problem in polynomial time. In 1999, Shor [5] proposed a quantum algorithm for integer factorization that runs in polynomial time. This algorithm can be used to break RSA [6] on an ideal quantum computer as the security of RSA depends on the fact that integer factorization of a large number is hard. Similarly, Grover's quantum algorithm [7] can be utilized for searching an unstructured database in polynomial time. Thus, it is believed that key exchange protocols based on traditional number-theoretic problems (such as the discrete logarithm problem and the integer factorization problem) are no longer secure.

In December 2016, National Institute of Standards and Technology (NIST) [35] issued a standardization call for quantum-resistant public key algorithms, together with requirements and evaluation criteria. Since computing platforms have different goals and constraints, this poses a great challenge to design and implement emerging cryptography standards in a single environment. We see a variety of computing capabilities in these, ranging from high-performance (real-time) virtualized environments to highly resource-constrained Internet of Things platforms. Therefore, as the current mainstream trend in computer technology and other security fields, it has become an urgent need to strengthen computer security and also increase the diversity of cryptographic primitives.

At present, there exists several alter techniques including based on hash function, error correcting code, lattice, multivariate polynomial and supersingular isogeny to construct post-quantum cryptographic systems. According to [9], lattice is the most promising and ideal competitive primitives for the construction of post-quantum KE schemes. In 2005, Regev proposed the learning with errors (LWE) problem [10], and demonstrated a reduction from the worst-case generic lattice problem to the LWE problem [10–12]. Subsequently, inspired by the NTRU protocol, Lyubashevsky et al. presented the ring learning with errors (RLWE) problem [13]. RLWE with additional algebraic structure relies on the worst-case hardness of problems in ideal lattices. Combining with the matrix format from LWE and the algebraic structure from RLWE [13], Langlois et al. constructed the module learning with errors (MLWE) problem [14]. In [15–17], various LWR-based KEMs and Fully Homomorphic Encryption are shown. Under the pressing requirement of the transition from traditional public-key cryptography to the post-quantum cryptography, many works proposed simple and practical post-quantum KE schemes based-on LWE and its variants [18].

However, small errors introduced in the (R/M)LWE-based KE [19, 20] schemes and LWR-based KE schemes [21, 22] lead to two approximate negotiated values generated by communication parties. Accordingly, considerable attention has been devoted to addressing the question of how to design an error reconciliation mechanism which makes the communication participants agree on the same shared-key. As a result, more and more research has been done on error reconciliation, but most have focused on extracting the most or the least significant bit information to negotiate the same shared-key.

1.1 Related works

Error Reconciliation Mechanism
Lattice-based key exchange protocol introduces errors into the mutual information of the two parties, and the calculated similar value has certain errors. Therefore, researches on error coordination mechanisms (used to eliminate such errors to negotiate the same shared-key) are particularly important. We first consider the process of lattice-based key exchange protocols, where the goal is for both parties need to communicate, after a series of information exchanges, both parties obtain a common key K. In 2012, Ding et al. [23, 24] proposed the first error reconciliation mechanism which extracts the least-significant bit of negotiated value as the shared secret bit. Conversely, Peikert et al. [25] designed a radically different approach which takes advantage of the high-bit information to derive the same shared-key. Subsequently, based on the mechanism of Peikert, Alkim et al. [26] designed the $\widetilde{D}_4$ reconciliation mechanism which is able to negotiate a one-bit shared-key by four coefficients. Compared to previous mechanisms [23, 25], $\widetilde{D}_4$ expands the error tolerance interval to $\frac{3}{4}q$. Saarinen et al. [27] proposed an improved Peikert reconciliation mechanism by throwing away the coefficient with high failure rate in the negotiation process. This method was instantiated in the RLWE-based post-quantum scheme HILA5 [27], and decreases the failure rate of HILA5 by sacrificing multiple unsatisfactory coefficients. In [28] Jin and Zhao formally formulated a universal and convenient error reconciliation mechanism referred to as optimally-balanced key consensus with noise (OKCN). Same idea as Peikert reconciliation mechanism [25], OKCN also utilized the most significant bits of each coefficient to negotiate the shared-key. Moreover, the inherent upper-bound analyzed in Jin's paper guides the parameter selection and the trade-off between the accuracy and the bandwidth.

In sum, most of the existing error reconciliation mechanisms based on lattices, such as Peikert [25], HILA5 [27], OKCN [28], etc., are negotiated based on the most significant bit information; while the error reconciliation mechanism of Ding is mainly based on the least significant bit information for negotiation. According to our meticulous observation, the existing works only extract the most or least significant bits of information to achieve the agreement of both parties. To some extent, these methods result in the waste of effective information in the negotiated values. So, in this paper, we propose an error reconciliation mechanism, which can negotiate the most and least significant bits of information at the same time.

Key exchange protocols
In order to complete the task of key exchange between two entities on an insecure channel, various key exchange protocols have been proposed. In 2012, Ding proposed an error reconciliation mechanism based on the least-significant bit [23], and designed a key exchange protocol based on RLWE. The Frodo protocol [9] based on LWE uses Peikert's error reconciliation to complete the key negotiation. This work proposes efficient sampleable noise distribution, efficient and dynamic generation of public parameters. In NewHope[26], a new error reconciliation (on $\tilde{D}_4$ lattice) is proposed. The protocol can greatly reduce the failure rate and improve post-quantum security. At the same time, it use NTT to accelerate the operation of polynomial multiplication. Although Kyber's key exchange protocol[29] is based on the RLWE problem, it has additional flexibility and security advantages. And, Saber[30] is based on the LWR problem. Cleverly[30] designed how to use the LWR to negotiate the shared-key K. The negotiation efficiency is greatly improved, and the negotiation information is more flexible and has higher security strength. In recent years, LWE-based AKEs can be achieved by instantiating generic constructions from public-key encryption (PKE) or KEMs. For example, recently quantum-safe AKEs from lattice-based KEMs for the TLS [31, 32, 1] have been constructed.

Recently, the researches of password authentication key exchange (PAKE) protocol have been paid more and more attention due to the requirement of remote access to privacy information in the prevalence of mobile devices. For applying PAKE protocols to establish secure remote communications in the quantum era, Ding et al. [33] first proposed the parallel extension of PAK and PPK based on the RLWE problem and proved the theoretical feasibility. To optimize the efficiency of Ding's PAK and PPK, Gao et al. [34] reduced the modulus q from $2^{32} - 1$ to $1073479681 < 2^{30}$, for the reason that the new modulus satisfies the condition of the number theoretic transform (NTT) algorithm to accelerate the polynomial multiplication. Yang et al. [35] further optimized the implementation of Ding-PAK by proposing a lightweight parameter set inspired by the scheme from [26]. Moreover, inspired by the two-party, Liu et al. [36] presented a three-party RLWE-based PAKE protocol, where two clients aim to agree on a session key with the help of a trusted server.

1.2 Our Contributions

To ensure that communication is established correctly and efficiently, the error reconciliation mechanism plays an essential role in the lattice-based post-quantum key exchange protocol. In this paper, we propose an efficient and universal reconciliation mechanism Speedy Error Reconciliation (SER) which takes full advantage of the effective information of each negotiation coefficient. Specifically, SER simultaneously extracts the effective information from the most and least significant bits of one coefficient to negotiate a two-bit shared-key. In particular, SER only needs to transmit the signal value of the most significant bit to achieve the negotiation of both the most and the least significant bits. Our contribution are listed as follows:

- SER makes full use of the effective information of the shared secret values σ_1 and σ_2 when performing the error reconciliation. By extracting the most and least significant bits of the secret value, a 2-bit key can be negotiated in one reconciliation, which improves the efficiency of key negotiation. In addition, SER expands the fault tolerance interval during key reconciliation by sharing a g-bit auxiliary signal between two peers. Thus, the improves the accuracy of negotiation.
- SER only transmits the auxiliary signal of the most significant bit of the negotiated values. As for the signal value of the least significant bit, it can be computed from the signal value of the most significant bit. Different from the current signal mechanism, SER negotiates the most and least significant bits. And the transmitted signal value proposed in this paper can help both parties negotiate the most and the least significant bits.
- SER is integrated into key exchange protocols based on LWE, RLWE and MLWE to evaluate practical performance. After comparing performance indicators such as failure rate, security strength, and CPU cycles, we found that SER is suitable for various scenarios. Among them, in the LWE-based key exchange scheme, the average efficiency of SER in negotiating each bit is 61.6% higher than that of Frodo [9]. Compared with NewHope [11], RLWE-based SER improves the average efficiency of per-bit key negotiation by 797.6%. In the MLWE-based scheme, SER performs well in the proposed three parameter sets.

We briefly introduce the main contributions of this paper and related works in Section 1. In Section 2, we review the background knowledge and theory to be used later, and introduce four most known error reconciliation mechanisms. Section 3 describes our error reconciliation in detail, along with correctness and security proofs. Next, Section 4 presents the application of our mechanism to lattice-based hard problems: LWE, Ring-LWE, Module-LWE, and reports the test effciency. Section 5 discusses the direction of future experiments and the accuracy of the coordination mechanism under different parameters. Section 6 concludes the full paper.

2 Preliminaries

In this section, we introduce some essential notations and review the basic background of the error reconciliation mechanism.

2.1 Notations

Let $\mathbb{Z}$ be the ring of rational integers. If $x \in \mathbb{R}$, the rounding function $\lfloor x \rceil = \lfloor x + \frac{1}{2} \rceil \in \mathbb{Z}$. Lattices are set of points in n dimensional plane. Mathematically, the definition is as follows: Let $\mathbb{R}^k$ be the k dimensional Euclidean space. A lattice in $\mathbb{R}^k$ is the set

$$L(a_1, a_2, a_3, \ldots, a_n) = \Big\{ \sum_{i=1}^{n} x_i \mathbf{a}_i \ : \ x_i \in \mathbb{Z} \Big\}$$

of all integral combinations of n linearly independent vectors $a_1, a_2, a_3, \ldots, a_n$ in $\mathbb{R}^k$ where $k \geq n$. The integers n and k are called rank and dimension of lattice respectively. If $n = k$, then the lattice is called full rank lattice. The sequence of vectors $a_1, a_2, a_3, \ldots, a_n$ is called lattice basis which can be represented in matrix form as $\mathbf{B} = [a_1, a_2, a_3, \ldots, a_n] \in \mathbb{R}^{k \times n}$

The LWE problem focuses on the three parameters: the modulus q, the dimension of the matrix n, and the error distribution χ. And we denote sampling x uniformly at random from S: $x \overset{\$}{\leftarrow} S$ (S is a set). The bold face capital letters represent matrices. And, the matrix $\mathbf{A}^T$ denotes the transpose of $\mathbf{A}$. If χ is a distribution over a set S, we use $\mathbf{X} \overset{\$}{\leftarrow} \chi(S^{n \times m})$ to represent generating an $n \times m$ matrix $\mathbf{X}$ by sampling each of its entries independently from S according to χ.

In the RLWE problem, let $\mathbb{Z}$ be the ring of rational integers. Let $\mathbb{Z}_q$, for an integer $q \geq 1$, denote the quotient ring $\mathbb{Z}/q\mathbb{Z}$. We define $R = \mathbb{Z}[X]/(X^n + 1)$ as the ring of integer polynomials modulo $X^n + 1$. By $R_q = \mathbb{Z}_q[X]/(X^n + 1)$ we mean the ring of integer polynomials modulo $X^n + 1$ where each coefficient is reduced modulo q. If χ is a probability distribution over R, $x \overset{\$}{\leftarrow} \chi$ means the sampling of $x \in R$ according to χ. If S is a set, $x \overset{\$}{\leftarrow} S$ means the sampling of x uniformly at random from S. The distribution χ is typically taken to be a rounded continuous or discrete Gaussian distribution over $\mathbb{Z}$ with center zero and standard deviation σ.

As for the MLWE problem, we define R_q^d as the vector including d ring polynomials in $R_q = \mathbb{Z}[X]/(X^n + 1)$. Moreover, we define $R_q^{k \times d}$ as the ring polynomial matrix with rank k and dimension d. The MLWE problem is the general version of the LWE problem and the RLWE problem: If we set the ring R_q to $\mathbb{Z}_q$, it becomes the LWE problem, and if we set $d = k = 1$, it becomes RLWE problem.

In other words, for the RLWE-based KE schemes, the ring polynomial in R_q is denoted as the bold lowercase such as $r \in R_q$. For the MLWE-based KE schemes, we use the bold capital letter to denote the polynomial matrix with $m \times n$ entries where each entry is sampled from R_q, such as $\mathbf{B} \in R_q^{m \times n}$. Moreover, the bold lowercase is denoted as the polynomial vector of dimension n, such as $\mathbf{v} \in R_q^n$.

2.2 LWE, RLWE, MLWE Probolems

For lattice-based hard problems, the LWE problem and its variants (RLWE and MLWE) are widely used to construct post-quantum schemes. In 2005, Regev proposed the LWE problem [39]. We define the decision LWE problem as follows:

Definition 1 (Decision LWE problem). *Let n and $q \geq 2$ be the dimension of a vector and the modulus, respectively. Let χ be an error distribution and $\mathbf{s} \leftarrow \chi(\mathbb{Z}_q^n)$. Define $O_{\chi,\mathbf{s}}$ as the oracle which does the following:*

1. *Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^n, e \leftarrow \chi(\mathbb{Z}_q)$;*
2. *Return $(\mathbf{A}, \mathbf{As} + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.*

The decision LWE problem for n, q, χ is to distinguish between polynomial independent samples from $O_{\chi,\mathbf{s}}$ and the same number of independent samples from an oracle $\mathcal{U}$ that returns uniform random samples from $(\mathbb{Z}_q^n, \mathbb{Z}_q)$.

However, the large computation of the matrix format in the LWE problem makes it inefficient when designing an LWE-based post-quantum scheme. In 2012, inspired by the NTRU protocol, Lyubashevsky et al. [24] proposed the RLWE problem. Compared with the LWE-based scheme, the RLWE-based scheme introducing the algebraic structure shows more efficient performance. Here, we define the decision RLWE problem as follows:

Definition 2 (Decision RLWE problem). *Let n and $q \geq 2$ be the rank of a polynomial and the modulus, respectively. Let χ be an error distribution and $s \leftarrow \chi$. Define $O_{\chi,s}$ as the oracle which does the following:*

1. *Sample $A \leftarrow R_q, e \leftarrow \chi$;*
2. *Return $(A, As + e) \in R_q \times R_q$.*

The decision RLWE problem for n, q, χ is to distinguish between polynomial independent samples from $O_{\chi,s}$ and the same number of independent samples from an oracle $\mathcal{U}$ that returns uniform random samples from (R_q, R_q).

In 2014, Langlois and Stehlé proposed the MLWE problem [40]. As a compromise between the LWE problem and the RLWE problem, the MLWE problem retains the matrix format, and concurrently, introduces the algebraic structure. Therefore, when designing a lattice-based scheme in multiple security scenarios, it is more flexible and simple than LWE and RLWE. Here, we define the decision version of the MLWE problem as follows.

Definition 3 (Decision MLWE problem). *Let n, d, k and $q \geq 2$ be the rank of a polynomial, the dimension of an error vector, the dimension of a secret vector and the modulus, respectively. Let χ be an error distribution and $\mathbf{s} \leftarrow \chi^k$. Define $O_{\chi,\mathbf{s}}$ as the oracle which does the following:*

1. *Sample $\mathbf{A} \leftarrow R_q^{d \times k}, \mathbf{e} \leftarrow \chi^d$;*
2. *Return $(\mathbf{A}, \mathbf{As} + \mathbf{e}) \in R_q^{d \times k} \times R_q^d$.*

The decision MLWE problem for n, d, q, χ is to distinguish between polynomial independent samples from $O_{\chi,\mathbf{s}}$ and the same number of independent samples from an oracle $\mathcal{U}$ that returns uniform random samples from $(R_q^{d \times k}, R_q^d)$.

2.3 Review of error reconciliation mechanisms

For two approximate values σ_1, σ_2 held by Alice and Bob respectively, the error reconciliation mechanism is to assist Alice and Bob to negotiate the same value k. Usually, a necessary signal value derived from one-side is used for assisting the negotiation of both parties. Next, we will review four known error reconciliation methodologies.

Ding's Error Reconciliation [23]
Since the relationship of values $\sigma_1 \in \mathbb{Z}_q$ and $\sigma_2 \in \mathbb{Z}_q$ in Ding's mechanism [23] is $\sigma_1 = \sigma_2 + 2\delta$, an important observation is that the parity of σ_1 and σ_2 is consistent. Thus, Alice and Bob can extract the same least-significant bit of σ as the final shared-key. In reality, a rewinding problem introduced from the odd modulus q causes the reverse parity of σ_1 and σ_2. Thus, the signal function Cha is responsible for the above consideration.

Let $E = \{\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{3q}{4} \rfloor\}$ be the middle interval of $\mathbb{Z}_q$. The signal function Cha can be defined as follows:

$$\mathsf{Cha}(\sigma) = \begin{cases} 0, & \sigma \in E; \\ 1, & \text{otherwise.} \end{cases}$$

Once Bob sends the signal value $b = \mathsf{Cha}(\sigma)$ to Alice, both parties can utilize the Mod_2 function to agree on the same shared-key $k \in \{0, 1\}$. The function $\mathsf{Mod}_2 : \mathbb{Z}_q \times \{0, 1\} \to \{0, 1\}$ is defined as follows:

$$\mathsf{Mod}(\sigma, b) = (\sigma + b \cdot \frac{q-1}{2}) \bmod q \bmod 2$$

Peikert's Error Reconciliation [25]
Contrary to Ding's mechanism [23], Peikert et al. made use of the high-bit information to negotiate the shared-key. Assume that Alice and Bob holds two approximate values $w \in \mathbb{Z}_q$ and $v \in \mathbb{Z}_q$, respectively. Bob invokes the modular rounding function and the cross-rounding function to finish its conciliation process where the output of the modular rounding function is the shared-key and the output of the cross-rounding function is the signal value. These two functions are defined as follows:

Definition 4. *Let q be an even modulus. Define the modular rounding function is defined as*

$$\lfloor x \rceil_{q,2} := \lfloor \frac{2}{q} x \rceil \bmod 2$$

and, the cross-rounding function is defined as

$$\langle x \rangle_{q,2} := \lfloor \frac{4}{q} x \rfloor \bmod 2.$$

Once Alice receives the signal value from Bob, she generates a shared-key in virtue of the reconciliation function.

Definition 5 (The reconciliation function). *Given an element $w \in \mathbb{Z}_q$ and the cross-rounding value $h = \langle v \rangle_{q,2}$ of a close element $v \in \mathbb{Z}_q$. Define the sets $I_0 = \{0, 1, \cdots, \lfloor \frac{q}{4} \rceil - 1\}$ and $I_1 = \{-\lfloor \frac{q}{4} \rfloor, \cdots, -1\}$. Let $E = [-\frac{q}{8}, \frac{q}{8})$, the reconciliation function is*

$$rec(w, h) = \begin{cases} 0, & \textit{if } w \in I_h + E \bmod q, \\ 1, & \textit{otherwise.} \end{cases}$$

Once the modulus q is odd, the bit derived by rounding will be biased. Thus, to avoid the bias in the derived bits, the randomized doubling function is introduced.

$$\mathsf{dbl}(x) = 2x + \bar{e}$$

where $\Pr[\bar{e} = 0] = \frac{1}{2}$, $\Pr[\bar{e} = 1] = \Pr[\bar{e} = -1] = \frac{1}{4}$.

HILA5's Error Reconciliation [27]
Similar to Peikert's mechanism [12], Alice and Bob holds two approximate values $w \in \mathbb{Z}_q$ and $v \in \mathbb{Z}_q$, respectively. And use error reconciliation to coordinate and calculate shared-key.HILA5 greatly improves the fault tolerance interval in the Peikert protocol, so HILA5 greatly reduces the failure rate of Peikert.

Definition 6. *Let q be an even modulus. The rounding function is defined as*

$$\lfloor x \rceil_{q,2} := \lfloor \frac{2}{q} x \rceil \bmod 2$$

and, the cross-rounding function, which comes from the Peikert error coordination mechanism, is defined as

$$\langle x \rangle_{q,2} := \lfloor \frac{4}{q} x \rfloor \bmod 2.$$

In this error reconciliation, Bob chooses a boundary window b, which defines shared bits to be used, and then sends his binary selection vector d to Alice:

$$d_i = \begin{cases} 1, \text{ if } & y_i \in [\lfloor \frac{q}{4} \rceil - b, \lfloor \frac{q}{4} \rceil + b] \cup [\lfloor \frac{3q}{4} \rceil - b, \lfloor \frac{3q}{4} \rceil + b], \\ 0, & \text{otherwise.} \end{cases}$$

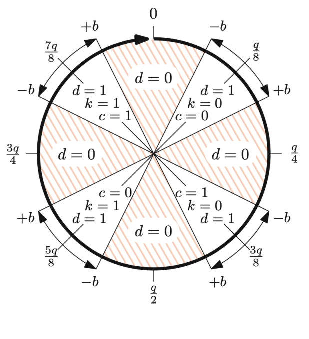


Fig. 1. Bob's key generation

Bob uses the rounding function to calculate a shared key k. In Fig. 1, if x belongs to the region of $d = 0$, the current value of k is discarded. In the area of $d = 1$, keep the current shared-key value, calculate the signal value C, and send C to Alice. Once Alice receives the signal value C from Bob, she generates a shared-key in virtue of the reconciliation function.

Optimal Key Consensus in Presence of Noise [28]
In the OKCN's machanism [28], it made use of the high-bit information to negotiate the shared-key. Alice and Bob holds two approximate values $\sigma_2 \in \mathbb{Z}_q$ and $\sigma_1 \in \mathbb{Z}_q$, respectively. Bob use the function $Con(\sigma_1, params)$ to finish its conciliation process. $Con(\sigma_1, params)$ can output the shared-key and the signal value v. Then Alice made use of σ_2 and the signal value v to compute the shared-key. Alice use $Rec(\sigma_2, v, pramas)$ to achieve that. k_1 and k_2 are the shared-keys calculated by both parties in the key negotiation.

Definition 7. *$params = (q, m, g, d, aux)$, $aux = \{q' = lcm(q, m), \alpha = q'/q, \beta = q'/m\}$. Define the function $Con(\sigma_1, params)$ and the function $Rec(\sigma_2, v, parmas)$ as:*

Algorithm 1 $Con(\sigma_1, params)$ & $Rec(\sigma_2, v, parmas)$

```
 1: procedure $Con(\sigma_1, params)$        $\sigma_1 \in [0, q-1]$
 2:     $e \leftarrow [-\lfloor(\alpha-1)/2\rfloor, \lfloor\alpha/2\rfloor]$
 3:     $\sigma_A = (\alpha\sigma_1 + e) \bmod q'$
 4:     $k_1 = \lfloor\sigma_A/\beta\rfloor \in \mathbb{Z}_m$
 5:     $v' = \sigma_A \bmod \beta$
 6:     $v = \lfloor v'g/\beta\rfloor$                $v \in \mathbb{Z}_g$
 7:     return$(k_1, v)$
 8: end procedure
 9: procedure $Rec(\sigma_2, v, params)$        $\sigma_1 \in [0, q-1]$
10:     $k_2 = \lfloor\alpha\sigma_2/\beta = (v+1/2)/g\rfloor \bmod m$
11:     return $k_2$
12: end procedure
```

3 Speedy Error Reconciliation

With the goal of improving the efficiency of error reconciliation and make full use of shared secret information, we propose the speedy error reconciliation (SER) mechanism in this section. By simultaneously reconciling the most and least significant bits of similar secret values, multiple bits can be reconciled in one negotiation. To keep the parity of the two similarity values σ_1 and σ_2 consistent, we make some minor adjustments to the lattice-based key exchange scheme. Two similar keys σ_1 and σ_2 computed by two peers should satisfy $\sigma_1 \bmod q = \sigma_2 \bmod q + 2\delta$, where $\delta \in \mathbb{Z}_q$.

3.1 Overall Process of SER

Definition 8. *The speedy Error Reconciliation mechanism $SER = (params, Neg, Com)$, briefly depicted in Fig. 1, is specified as follows.*

Alice		**Bob**
σ_1	$\approx$	σ_2
		$v \leftarrow Signal(\sigma_2, params)$
		$k^{'} \leftarrow Neg(\sigma_2, v, params)$
	$\xleftarrow{v}$	
$k \leftarrow Com(\sigma_1, v, params)$		

Fig. 2. Brief depiction of SER

- $params = (q, g)$ denotes the system parameters, where q and g are positive integers satisfying $q \geq 2^{11}, 1 \leq g \leq log_2(q) - 3$ (which will influence the security, correctness and bandwidth of a lattice-based KE scheme).
- $v \leftarrow Signal(\sigma_2, params)$: Taking σ_2 and $params$ as inputs, the deterministic polynom-ial-time algorithm $Signal(\sigma_2)$ outputs an auxiliary value v, which is an indication signal that will be publicly passed to Alice to help both parties reach a consensus.
- $k^{'} \leftarrow Neg(\sigma_2, v, params)$: Taking σ_2, v and $params$ as inputs, the deterministic polynomial-time algorithm $Neg(\sigma_2, v)$ outputs $k^{'} \in \mathbb{Z}_2^2$.
- $k \leftarrow Com(\sigma_1, v, params)$: Taking σ_1, v and $params$ as inputs, the deterministic polynomial-time algorithm $Com(\sigma_1, v)$ outputs $k \in \mathbb{Z}_2^2$.

3.2 Construction and analysis of SER

As can be seen from Section 3.1, the SER scheme contains three functions, the signal generation function, the key generation function and the coordination function. To ensure successful coordination, we also need to determine the fault tolerance interval. First, we define $\overline{k}$ to represent the binary representation of k, then $\overline{k} = \overline{k_h k_l}$, $\overline{k'} = \overline{k_h^{'} k_l^{'}}$, where k_h and k_l are the high and low bits of the 2-bit shared key calculated by Alice, and $k_h^{'}$ and $k_l^{'}$ also represent the high and low bits of the 2-bit shared key calculated by Bob. The specific algorithms are described as follows.

Signal generation
The function $Signal(\sigma)$ is defined as follows.

$$v = \overline{v_g \ldots v_2 v_1} = Signal(\sigma) = \lfloor \frac{2^{g+1}}{q} \cdot \sigma \rfloor - \lfloor \frac{2}{q} \cdot \sigma \rfloor \cdot 2^{g+1}$$

where q represents the overall modulus, g reprensents the number of bits to send the binary signal, and σ reprsents similar public values calculated after the two parties exchange information. where g represents .

Key negotiation (Neg)
In this phase, Bob calculates $k_l^{'}$ and $k_h^{'}$ separately to form $k^{'}$, which is the key coordinated between Alice and Bob and $\overline{k'} = \overline{k_h^{'} k_l^{'}}$. We define odd q>2, $\mathbb{Z}_q = \{0, \cdots, q-1\}$, and half of $\mathbb{Z}_q$ as $D = (\{\lfloor \frac{q}{4} \rfloor, \cdots \lfloor \frac{3q}{4} \rfloor - 1\} \cap \mathbb{Z}_q$.

To assist in the calculation of k_l', we define the variable v_0 as follows.

$$v_0 = \begin{cases} 0, & \sigma \in D \\ 1, & else \end{cases}$$

k_l' is the least significant bit of information that Bob obtained through reconciliation. $k_l^{'}$ generation: To prevent the rewinding problem introduced from the odd modulus q, we define $k_l^{'}$ as follows.

$$k_l^{'} = (\sigma + v_0 \cdot \frac{q-1}{2}) \bmod q \bmod 2$$

$k_h^{'}$ generation: k_h' is the most significant bit of information that Bob obtained through reconciliation. We define $k_h^{'}$ as follows.

$$k_h^{'} = \lfloor \frac{2\sigma}{q} \rceil \bmod 2$$

Key compromise (Com)
In this phase, according to the signal value v passed by Bob, Alice calculates k_l and k_h separately to form k, which is the key coordinated between Alice and Bob and $\overline{k} = \overline{k_h k_l}$.

k_l generation: k_l is the least significant bit of information that Alice obtained through reconciliation. we define k_l as follows.

$$k_l = (\sigma + v_0^{'} \cdot \frac{q-1}{2}) \bmod q \bmod 2$$

where σ is Alice's approximate secret value and v_0' can be calculated according to the following formula

$$v_0' = \begin{cases} 0, & (k_h \oplus v_g) \cdot \frac{q}{2} + v_g \cdot \frac{q}{4} + \cdots + v_1 \cdot \frac{q}{2^{g+1}} \in D \\ 1, & else \end{cases}$$

where k_h is the most significant bit of information that Alive obtained through reconciliation, v_1 to v_g represent the value of the signal value v in different binary bits.

k_h generation: As we methioned above, k_h is the most significant bit of information that Alice obtained through reconciliation. To eliminate the errors that may be caused by truncation, we define the tolerance interval $E = (-\frac{q}{4} \cdot (1-(\frac{1}{2})^g),\ \frac{q}{4} \cdot (1-(\frac{1}{2})^g)) \cap \mathbb{Z}_q$, and define k_h as follows.

$$k_h = \begin{cases} 0, & \sigma \in E_0 \\ 1, & \sigma \in E_1 \end{cases}$$

where $E_0 = C_0 + E$, $E_1 = C_1 + E$, C_0 and C_1 are calculated according to the function of Coordinate, which will be introduced below.

Error Tolerance Interval

Since the interval to which σ_1 and σ_2 belong determines the values of k_h and k'_h. In this part, In this part, we identify specific ranges that help Alice and Bob reach the consensus. C_b is the interval in which σ_2 is located and is defined as follows:

$$\begin{aligned} C_b &= Coordinate(v,b) \\ &= [(b \oplus v_g)\cdot \frac{q}{2} + v_g \cdot \frac{q}{4} + \cdots + v_{g-k}\cdot \frac{q}{2^{k+2}} + \cdots + v_1 \cdot \frac{q}{2^{g+1}}, \\ &(b \oplus v_g)\cdot \frac{q}{2} + v_g \cdot \frac{q}{4} + \cdots + v_{g-k}\cdot \frac{q}{2^{k+2}} + \cdots + v_1 \cdot \frac{q}{2^{g+1}} + \frac{q}{2^{g+1}}) \\ &b \in \{0,1\},\ k \in \{1,2,\cdots,g\} \end{aligned}$$

When σ_1 falls outside of C_0 and C_1, within a certain fault tolerance, Alice and Bob can use the reconciliation mechanism to get the same value. We define this tolerance range as: $E_b = C_b + E$, where $b \in \{0,1\}$.

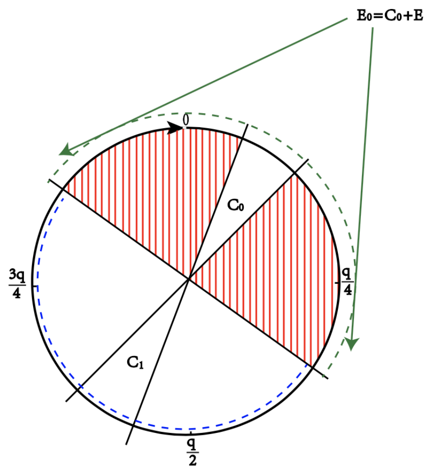


Fig. 3. An example of error tolerance interval

Fig. 3 shows the relationship between the deterministic interval C_0 and C_1 and the fault-tolerant interval. C_0 and C_1 are the intervals to which σ_2 belongs. If σ_1 falls within these two intervals, Alice and Bob can deterministically calculate the same key. Otherwise, if σ_1 is not within this range, then within the fault tolerance interval, Alice can obtain the same key as Bob through the reconciliation mechanism.

E_0 and E_1 are the deterministic interval plus the error tolerance interval E. We can get C_0 and C_1 through the function $Coordinate(v,b)$, and then get E_0 and E_1, where $E_0 = C_0 + E$, $E_1 = C_1 + E$, which satisfy the following relationship: $E_0 \cap E_1 = \emptyset$, $E_0 \cup E_1 = \mathbb{Z}_q$. It is not difficult to see that the spans of the intervals C_0 and C_1 are both $q/2^{g+1}$. Fig. 3 shows that when σ_2 falls in the C_0 interval, as long as σ_1 falls in the E_0 interval, Alice and Bob can reach the consensus.

3.3 Correctness of *SER*

Theorem 1. *The SER scheme is correct with the error tolerance:*

$$\sigma_1 - \sigma_2 = d \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$$

Remark 1. Theorem 1 reveals the upper limit of the error tolerance.

Proof. Since the ranges of C_0 and C_1 are both $q/2^{g+1}$, the range size of the remaining intervals are $q - 2 \cdot \frac{q}{2^{g+1}}$. Dividing these remaining intervals into four parts, we get the fault tolerance interval $d \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$.

On the premise that $d<q$, we divide the proof process into two parts: First we prove that k_l and k_l' are equal, and then we prove that k_h and k_h' are equal.

Lemma 1. *The Least significant bit reconciliation mechanism works correctly with* $|\sigma_1 - \sigma_2|_q \leq d$.

Proof. Suppose $|\sigma_1 - \sigma_2|_q \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$, $\sigma_1 \bmod q = \sigma_2 \bmod q + 2\delta\, (in\, \mathbb{Z})$.

For any $\sigma_1, \sigma_2 \in \mathbb{Z}_q$ such that $\sigma_1 - \sigma_2 = 2\delta$ and $d \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$.

Let $D = \{\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{3q}{4} \rfloor - 1\}$. If $\sigma_2 \in D$, $v_0 \leftarrow 0$, else $v_0 \leftarrow 1$. If $\sigma_2 \in D$, $v_0 \leftarrow 0$, and $|\sigma_2 + v_0 \cdot \frac{q-1}{2} \bmod q| \in D$. If $\sigma_2 \notin D$, $v_0 \leftarrow 1$, it is not hard to see

$$(\sigma_2 + v_0 \cdot \frac{q-1}{2}) \bmod q \in D.$$

It can be seen that σ_1 and σ_2 have the following relationship:

$$\begin{aligned}(\sigma_1 + v_0 \cdot \frac{q-1}{2}) \bmod q &= (\sigma_2 + v_0 \cdot \frac{q-1}{2} + 2\delta) \bmod q \\ &= (\sigma_2 + v_0 \cdot \frac{q-1}{2}) \bmod q + 2\delta,\end{aligned} \tag{1}$$

Since $(\sigma_2 + v_0 \cdot \frac{q-1}{2}) \bmod q \in D$ So, it can be seen:

$$(\sigma_2 + v_0 \cdot \frac{q-1}{2}) \bmod q + 2\delta \in \mathbb{Z}_q, \, (2\delta \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^{g-1}))$$

This ensures that σ_1 and σ_2 do not round thus:

$$\begin{aligned}k_l &= (\sigma_1 + v_0 \cdot \frac{q-1}{2} \quad \bmod q) \quad \bmod 2 \\ &= (\sigma_2 + v_0 \cdot \frac{q-1}{2} \bmod q) \bmod 2 \\ &= k_l^{'}.\end{aligned} \tag{2}$$

The above process shows our error reconciliation mechanism (in generating k_l and $k_l^{'}$) is correct.

Lemma 2. *The most significant bit reconciliation mechanism works correctly with $|\sigma_1 - \sigma_2|_q \leq d$.*

Proof. Let $\overline{v_g v_{g-2} \cdots v_1} = Signal_h(\sigma_2)$, where $\overline{v_g v_{g-2} \cdots v_1}$ is a specific description of the g bits of σ_2, which is used to help Alice determine the interval to which σ_2 belongs. When σ_2 falls into the C_0 interval, as long as σ_1 falls into the E_0 interval, Alice and Bob can get the same value 0, otherwise when σ_2 falls into the C_1 interval, as long as σ_1 falls into the E_1 interval, both parties can get the same value 1.

For example, if $k_h^{'} = 0$, then $\sigma_2 \in C_0$, since $|\sigma_1 - \sigma_2|_q \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$, it can be inferred that $\sigma_1 \in E_0$ and $k_h = 0$. Conversely, if $k_h = 0$, then $\sigma_1 \in E_0$, since $|\sigma_1 - \sigma_2|_q \leq \frac{q}{4} \cdot (1 - (\frac{1}{2})^g)$, it can be inferred that $\sigma_2 \in C_0$ and $k_h^{'} = 0$.

□

3.4 Security of *SER*

Theorem 2 (Security of SER). *SER is secure if k and k' are independent with v when σ_1 and $\sigma_2 \leftarrow \mathbb{Z}_q$, and k and k' follow a uniform distribution over $\mathbb{Z}_2$.*

We divide the proof process into two parts: First we prove that k' and v are independent of each other, and k' follows a uniform distribution on $\mathbb{Z}_2^2$. And then we prove that k and v are independent of each other, and k follows a uniform distribution on $\mathbb{Z}_2^2$.

Lemma 3. *SER is secure if k' and v are independent of each other, and k' follows a uniform distribution on $\mathbb{Z}_2^2$.*

Proof. In generating k_l', we demonstrate that the computation of v_0 follows a uniform distribution on $\mathbb{Z}_2$. Consider the map $f : \mathbb{Z}_q \to \mathbb{Z}_2$, and

$$v_0 = f(\sigma) = \begin{cases} 0, & \sigma \in D \\ 1, & else \end{cases}$$

It is easy to see that f is a one-to-one map. If q is even, $\sigma \leftarrow \mathbb{Z}_q$ is distributed uniformly on $\mathbb{Z}_q$, v_0 also follows a uniform distribution. If q is odd, for any $\sigma \in \mathbb{Z}_q$, we define the function $Signal_b(\sigma)$, where $b \overset{\$}{\leftarrow} \{0,1\}$ (See Section 4.3 for details). Since $\sigma \leftarrow \mathbb{Z}_q$ is ditributed uniformly on $\mathbb{Z}_q$, v also follows a uniform distribution.

In the simliar way, we define $f^{'} : \mathbb{Z}_q \cdot \mathbb{Z}_2 \to \mathbb{Z}_2$. According to $k_l' = (\sigma + v \cdot \frac{q-1}{2}) \bmod q \bmod 2$, $\sigma = k_l' + \delta_1 q + 2\delta - \frac{q-1}{2} v = f^{'}(k_l', v)$. $f^{'}(k_l', v)$ is obviously a one-to-one map. Since σ and v are uniformly distributed on $\mathbb{Z}_q$ and $\mathbb{Z}_2$, k_l' is also uniformly distributed on $\mathbb{Z}_2$. Furthermore, as k_l' and v_0 are uniformly distributed on $\mathbb{Z}_2 \times \mathbb{Z}_2$, k_l' and v are independent.

In generating k_h', We demonstrate that $v = Signal(\sigma)$ follows a uniform distribution over $\mathbb{Z}_2^g$. Consider the map $f : \mathbb{Z}_q \to \mathbb{Z}_2^g$:

$$f(\sigma) = Signal(\sigma) = \lfloor \frac{2^g + 1}{q} \cdot \sigma \rfloor - \lfloor \frac{2}{q} \cdot \sigma \rfloor \cdot 2^{g+1}$$

It is easy to see that f is a one-to-one map. Since $\sigma \leftarrow \mathbb{Z}_q$ follows a uniform distribution. For every v, $\sigma \leftarrow \mathbb{Z}_q$ where multiple σ values correspond to one v value, and the number of σ corresponding to each v is the same. So v follows a uniform distribution over $\mathbb{Z}_2^g$.

And also, we define $f^{'} : \mathbb{Z}_2 \to \mathbb{Z}_q$. According to $k_h^{'} = \lfloor \frac{2\sigma}{q} \rceil \bmod 2$, $\sigma = \frac{k_1' q + \delta q + 2\delta q}{2} = f^{'}(k_1')$, and $f^{'}$ is obviously a one-to-one map. Since σ is uniformly distributed on $\mathbb{Z}_q$, k_l' is also uniformly distributed on $\mathbb{Z}_2$. □

Lemma 4. *SER is secure if k and v are independent of each other, and k follows a uniform distribution on $\mathbb{Z}_2^2$.*

Proof. As for generating k: In generating k_l, we need v_0' and define:

$$v_0' = \begin{cases} 0, & (k_h \oplus v_g) \cdot \frac{q}{2} + v_g \cdot \frac{q}{4} + \cdots + v_1 \cdot \frac{q}{2^{g+1}} \in D \\ 1, & else \end{cases}$$

In the above, v follows a uniform distribution over $\mathbb{Z}_2^g$. So, v_0' follows a uniform distribution over $\mathbb{Z}_2$. And, $k_l = (\sigma + v_0^{'} \cdot \frac{q-1}{2}) \bmod q \bmod 2$. Similar to generate k_l' and k_l also follows a uniform distribution over $\mathbb{Z}_2$ and independent with v.

In generating k_h, we define:

$$k_h = \begin{cases} 0, & \sigma \in E_0 \\ 1, & \sigma \in E_1 \end{cases}$$

The intervals E_0 and E_1 are depended on v. Because of the lattice problem, σ follows a uniform distribution over $\mathbb{Z}_q$. Thus, it is easy to see k_h follows a uniform distribution over $\mathbb{Z}_2$ and independent with v.

Furthermore, as k_l' and k_h' are uniformly distributed on $\mathbb{Z}_2$, k_l' is independent with σ. In summary, in SER, k' and v are independent of each other, and k' follows a uniform distribution on $\mathbb{Z}_2^2$. Within the allowable error tolerance interval, $k = k'$. k also follows a uniform distribution on $\mathbb{Z}_2^2$. □

3.5 Special parameters and a simplified version

Since both k_h and k_l require at least one bit of signal to aid consensus, g(the number of digits in v) needs to be greater than or equal to 1. If we select $g = 1$, it is necessary to strictly control the distribution of e in different protocols, otherwise it will cause excessive errors and consensus failure. After comprehensive consideration, we choose $g = 2$ as the simplified version. The following is the reconciliation process for the case of $g = 2$.

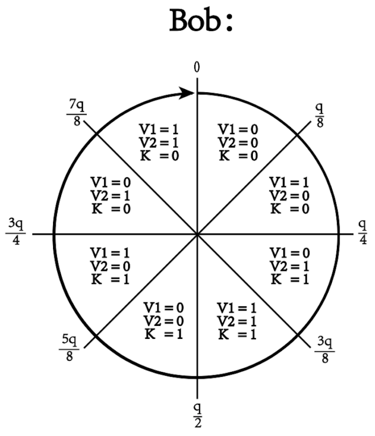
Alice		**Bob**
σ_1	$\approx$	σ_2
		$v = \overline{v_2 v_1} \leftarrow Signal(\sigma_2)$
		If $\sigma_2 \in D$ $v_0 \leftarrow 0$, else $v_0 \leftarrow 1$
		$k_l^{'} = (\sigma + v_0 \cdot \frac{q-1}{2}) \bmod q \bmod 2$
		$k_h^{'} = \lfloor \frac{2\sigma}{q} \rceil \bmod 2$
		$k^{'} \leftarrow \overline{k_h^{\prime} k_l^{\prime}}$
	$\xleftarrow{v}$	
$\Delta = (k_h \oplus v_2) \cdot \frac{q}{2} + v_2 \cdot \frac{q}{4} + v_1 \cdot \frac{q}{8}$		
If $\Delta \in D$ $v_0^{'} \leftarrow 0$, else $v_0^{'} \leftarrow 1$		
$k_l = (\sigma_1 + v_0^{'} \cdot \frac{q-1}{2}) \bmod q \bmod 2$		
$\overline{v_2 v_1} \leftarrow v$		
If $\sigma_1 \in E_0$, $k_h \leftarrow 0$, else $k_h \leftarrow 1$		
$k \leftarrow \overline{k_h k_l}$		
k	$=$	$k^{'}$

Fig. 4. Brief depiction of SER

Alice computes C_0 and C_1 using the Coordinate function as follows:

v_2&v_1	C_1	E_1	C_0	E_0
0&0	$[\frac{q}{2}, \frac{5q}{8}) \cap \mathbb{Z}$	$[\frac{5q}{16}, \frac{13q}{16}) \cap \mathbb{Z}$	$[0, \frac{q}{8}) \cap \mathbb{Z}$	$[0, \frac{5q}{16}) \cup [\frac{13q}{16}, q-1] \cap \mathbb{Z}$
0&1	$[\frac{5q}{8}, \frac{3q}{4}) \cap \mathbb{Z}$	$[\frac{7q}{16}, \frac{15q}{16}) \cap \mathbb{Z}$	$[\frac{q}{8}, \frac{q}{4}) \cap \mathbb{Z}$	$[0, \frac{7q}{16}) \cup [\frac{15q}{16}, q-1] \cap \mathbb{Z}$
1&0	$[\frac{q}{4}, \frac{3q}{8}) \cap \mathbb{Z}$	$[\frac{q}{16}, \frac{9q}{16}) \cap \mathbb{Z}$	$[\frac{3q}{4}, \frac{7q}{8}) \cap \mathbb{Z}$	$[0, \frac{q}{16}) \cup [\frac{9q}{16}, q-1] \cap \mathbb{Z}$
1&1	$[\frac{3q}{8}, \frac{q}{4}) \cap \mathbb{Z}$	$[\frac{3q}{16}, \frac{11q}{16}) \cap \mathbb{Z}$	$[\frac{7q}{8}, q-1) \cap \mathbb{Z}$	$[0, \frac{3q}{16}) \cup [\frac{11q}{16}, q-1] \cap \mathbb{Z}$

In Fig. 4, it shows the flow of coordination between Alice and Bob when the parameter $g = 2$. Fig. 5 shows a schematic diagram of Bob's computation of v_1, v_2 and $k_h^{\prime}$ by σ_2 in the case of $g = 2$. Fig. 6 shows the schematic diagram of Alice's calculation of k_h through the signal values v_1 and v_2.



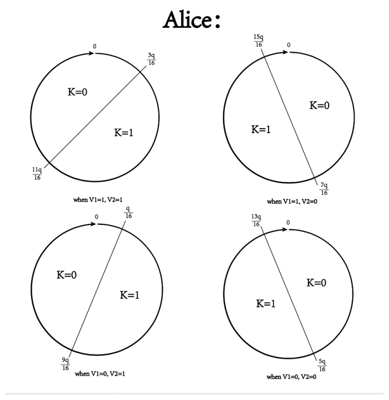


Fig. 5. Bob's secret value K and signal v_1, v_2 in different intervals

Fig. 6. Alice's secret value K in different intervals

4 Implementations of SER

In this section, we integrate SER into key exchange protocols based on LWE, RLWE and MLWE respectively to observe the correctness, security and efficiency of the SER scheme. By comparing with the original protocol in terms of security, efficiency, etc., we can tap the advantages of SER and find suitable scenarios for it. We run these protocols on macOS 11.2.3, Apple clang version 12.0.5 (clang-1205.0.22.9), Intel Core i5 2.30GHz. Its value represents the maximum value of the error [14, 42].

4.1 Comparison of three error reconciliation mechanisms

Table 1. Comparison of SER, OKCN, and Peikert in Frodo protocol

	q	n	l		g			DIST.	error probability			bw. (kB)			—K—		
			SER	OKCN & Frodo	SER	OKCN	Frodo		SER	OKCN	Frodo	SER	OKCN	Frodo	SER	OKCN	Frodo
Challenge	2^{11}	352	8	8	3	2	1	3	$2^{-22.8}$	$2^{-80.1}$	$2^{-41.8}$	7.75	7.76	7.75	128	64	64
Classical	2^{12}	592	8	8	3	2	1	4	$2^{-77.6}$	$2^{-70.3}$	$2^{-36.2}$	14.22	14.22	14.22	128	128	128
Recommended	2^{15}	752	16	8	3	3	1	5	$<2^{-300}$	$2^{-105.9}$	$2^{-38.9}$	22.58	22.58	22.57	512	256	256
Paranoid	2^{15}	864	16	8	3	3	1	6	$<2^{-300}$	$2^{-91.9}$	$2^{-33.8}$	25.94	25.94	25.93	512	256	256

We replace the error reconciliation mechanism in Frodo[9] with SER and OKCN [28] to compare the performance of the three mechanisms. Table 1 shows the failure rate, traffic flow and the number of key bits obtained by negotiation for the three error reconciliation mechanisms in the four scenarios. It can be seen that in *Classical*, *Recommended* and *Paranoid*, the failure rate of SER is lower than that of OKCN [28] and Frodo[9]. In *Recommended* and *Paranoid*, the failure rate of SER is less than 2^{-300}, which is much lower than other mechanisms. The traffic flow of the three error reconciliation mechanisms is roughly equivalent. When comparing the number of key bits generated in *Challenge*, *Recommended*, and *Paranoid*, SER is twice as high as the other two mechanisms. Clearly, SER outperforms the other two schemes in terms of failure rate and negotiated key bits. The reason for the higher failure rate of SER in *Challenge* is that the value of q is too small, and the error distribution Dist. is too large after doubling, resulting in a higher failure rate. Therefore, in Section 4.2, we propose parameter settings more suitable for SER.

4.2 SER in LWE

To observe the performance of SER in LWE-based protocols, we replace the error reconciliation mechanism in Frodo[9] with SER and compare the performance of these two protocols after running. Since we assumed $\sigma_1 \bmod q = \sigma_2 \bmod q + 2\delta$ in Section 3, we do some special processing to the Frodo[9] protocol to facilitate the replacement of the error reconciliation mechanism. In Fig. 7, We show a specific protocol scheme that replaces Frodo's error reconciliation mechanism with SER.

Client C		**Server** S
$seed_A \overset{\$}{\leftarrow} U(\{0,1\}^s)$		
$A \leftarrow Gen(seed_A)$		
$S, E \overset{\$}{\leftarrow} \chi(\mathbb{Z}_q^{n\times\overline{n}})$		
$B \leftarrow AS + 2E$	$\xrightarrow[\in\{0,1\}^s \times \mathbb{Z}_q^{n\times\overline{n}}]{seed_A, B}$	$A \leftarrow Gen(seed_A)$
		$S', E' \overset{\$}{\leftarrow} \chi(\mathbb{Z}_q^{\overline{m}\times n})$
		$B' \leftarrow S'A + 2E'$
		$E'' \overset{\$}{\leftarrow} \chi(\mathbb{Z}_q^{\overline{m}\times\overline{n}})$
		$V \leftarrow S'B + 2E''$
		$C \leftarrow Signal(V)$
	$\xleftarrow[\in\mathbb{Z}_q^{\overline{m}\times n} \times \mathbb{Z}_q^{\overline{m}\times\overline{n}}]{B', C}$	
$K \leftarrow Com(B'S, C)$		$K \leftarrow Neg(V, params)$

Fig. 7. The LWE-based protocol

Proposed Parameters

Frodo proposes four parameter sets: *Challenge*, *Classical*, *Recommended* and *Paranoid*, which also apply to SER. We make some small adjustments to these parameters to accommodate SER. Under the premise that correctness and safety are acceptable, we change q to improve the overall efficiency. In *Challenge* and *Classical* we increase the parameter q, in *Recommend* and *Paranoid* we decrease the parameter q. Table 2 lists the parameter selection and failure rates of Frodo and SER under the four parameter sets. It can be seen from the table that after adjusting q, the correct rate of SER has been greatly improved.

Table 2. Parameter selection of Frodo and SER.

Scheme	n	q	g	DIST.	$B \cdot m^2$	failure	—K—
Challenge(Frodo)	352	2^{11}	1	3	$1 \cdot 8^2$	$2^{-41.8}$	64
Classical(Frodo)	592	2^{12}	1	4	$2 \cdot 8^2$	$2^{-36.2}$	128
Recommend(Frodo)	752	2^{15}	1	5	$4 \cdot 8^2$	$2^{-38.9}$	256
Paranoid(Frodo)	864	2^{15}	1	6	$4 \cdot 8^2$	$2^{-33.8}$	256
Challenge(SER)	352	2^{12}	3	3	$2 \cdot 8^2$	$2^{-86.5}$	128
Classical(SER)	592	2^{12}	3	4	$2 \cdot 8^2$	$2^{-77.6}$	128
Recommend(SER)	752	2^{13}	3	5	$2 \cdot 16^2$	$2^{-85.8}$	512
Paranoid(SER)	864	2^{13}	3	6	$2 \cdot 16^2$	$2^{-75.0}$	512

Table 3. Security of Frodo under four parameters.

Scheme (Frodo)	Attack	Rounded Gaussian					Post-reduction		
		m	b	C	Q	P	C	Q	P
Challenge	Primal	338	266	-	-	-	-	-	-
	Dual	331	263	-	-	-	-	-	-
Classical	Primal	549	442	138	126	100	132	120	95
	Dual	544	438	136	124	99	130	119	94
Recommend	Primal	716	489	151	138	110	145	132	104
	Dual	737	485	150	137	109	144	130	103
Paranoid	Primal	793	581	179	163	129	178	162	129
	Dual	833	576	177	161	128	177	161	128

Table 4. Security of SER under four parameters.

Scheme (SER)	Attack	Rounded Gaussian					Post-reduction		
		m	b	C	Q	P	C	Q	P
Challenge	Primal	338	266	-	-	-	-	-	-
	Dual	331	263	-	-	-	-	-	-
Classical	Primal	549	442	138	126	100	132	120	95
	Dual	544	438	136	124	99	130	119	94
Recommend	Primal	699	582	179	163	129	172	156	123
	Dual	706	577	177	162	128	170	155	122
Paranoid	Primal	798	687	210	191	151	209	190	151
	Dual	778	682	208	190	150	208	189	150

Performance Comparison

Tables 5 show the number of CPU cycles for Frodo and SER under four-parameter sets, respectively. The 'total' in the table records the average and median of the total number of CPU runs, and the 'perbit' in the table records the average and median of the number of cycles used by CPU to complete a 1-bit key.

Table 5. Number of CPU cycles required for Frodo.

Scheme	total median	total average	perbit median	perbit average	—K—
Challenge(Frodo)	826369	859624	12912	13431	64
Classical(Frodo)	1854861	1978352	14491	15455	128
Recommend(Frodo)	2573847	2678434	10054	10462	256
Paranoid(Frodo)	3773053	3973444	14738	15521	256
Challenge (SER)	836789	878876	6537	6866	128
Classical (SER)	1918765	2022579	14990	15801	128
Recommend(SER)	2133762	2202166	4168	4301	512
Paranoid(SER)	3375476	3298573	6593	6443	512

As can be seen from Table 2-5, the key exchange protocol replaced by SER has a significant improvement in correctness, security and CPU operation efficiency compared with the traditional Frodo key exchange protocol. Especially in terms of CPU operation efficiency, it can be seen from Table 4-5 that our scheme improves the key consensus of each bit key by 61.6%.

4.3 SER in RLWE

We replace the error reconciliation mechanism in the RLWE-based key exchange protocol NewHope [26] with SER in this section. To accommodate SER, we also make some adjustments to the NewHope [26] protocol. In Fig. 8, we show a specific protocol scheme that replaces NewHope's error reconciliation mechanism [26] with SER.

Client C		**Server** S
$seed_A \overset{\$}{\leftarrow} \{0,1\}^{256}$		
$a \leftarrow \text{Parse}(\text{SHAKE}(seed))$		
$s, e \overset{\$}{\leftarrow} \Psi_{16}^{n}$		$a \leftarrow \text{Parse}(\text{SHAKE}(seed))$
$b \leftarrow as + 2e$	$\xrightarrow{seed,b}$	$u \leftarrow as' + 2e'$
		$v \leftarrow bs' + 2e''$
$v' \leftarrow us$	$\xleftarrow{v,r}$	$r \overset{\$}{\leftarrow} Signal(v)$
$k' \leftarrow Com(v', r)$		$k \leftarrow Neg(v, r)$
$\mu \leftarrow \text{SHA3-256}(k')$		$\mu \leftarrow \text{SHA3-256}(k)$

Fig. 8. The RLWE-based protocol

We know that in the RLWE-based scheme, the value of q must be an odd number, which results in the number of values in the D interval being one less than the number of values outside the D interval, resulting in a slightly higher probability of the signal value v_0 taking 0 than taking 1.

To get uniformly distributed 0 and 1 values, we modify the *Signal* function as follows. $b \overset{\$}{\leftarrow} \{0,1\}$ denotes randomly to choose coins and assigning the output to b.

$$v_0 = Signal_b(\sigma) = \begin{cases} 1, & \sigma \in \{\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{3q}{4} \rfloor - b\} \\ 0, & else \end{cases} \quad b \overset{\$}{\leftarrow} \{0,1\}$$

Proposed Parameters

In NewHope [26], only one set of parameters is given, so we also give a set of *classical* parameters accordingly, and test its correctness and safety. The specific results are shown in Table 6. It is worth mentioning that in NewHope [26], four approximations are negotiated to obtain a one-bit key, so the failure rate of the NewHope scheme [26] is very low and can be ignored.

Table 6. Parameters for SER of key exchange from RLWE.

Scheme	n	q	g	DIST.	failure	primal	Dual	$\|K\|$
NewHope	1024	12289	-	4	–	200	199	1024
SER	1024	12289	3	4	$2^{-124.4}$	198	191	2048

Performance Comparison

Table 7 shows the CPU cycles of the NewHope scheme [26] and our SER scheme under the classic parameters. Same as Section 4.2, the 'total' in the table records the average and median of the total number of CPU runs, and the 'perbit' in the table records the average and median of the number of cycles used by CPU to complete a one-bit key consensus.

Table 7. Number of CPU cycles required for SER and NewHope [26]

Scheme	total median	total average	perbit median	perbit average	—K—
NewHope	442840	468923	1729	1831	1024
SER	386650	417872	188	204	2048

It can be seen that our proposed error coordination mechanism performs well in the RLWE-based key exchange protocol. While ensuring accuracy and security, it also has a significant improvement in efficiency compared to NewHope [26]. Under the premise of ensuring a lower failure rate, the perbit coordination efficiency is increased by 797.6%.

4.4 SER in MLWE

Since there is no mature MLWE-based key exchange scheme yet, we construct a SER-based key exchange protocol to test the correctness, security and operational efficiency of SER on MLWE. In Fig. 9, We show a specific protocol scheme based on MLWE with SER.

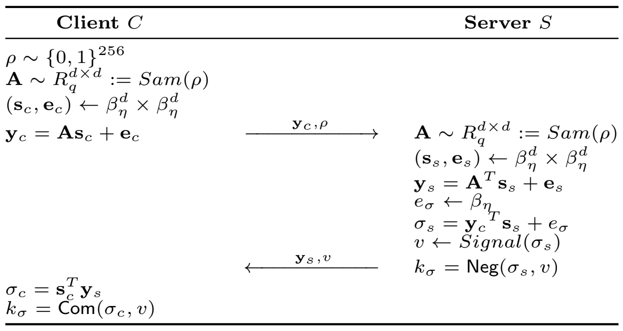

Client C
Server S
$\rho \sim \{0,1\}^{256}$
$\mathbf{A} \sim R_q^{d\times d} := Sam(\rho)$
$(\mathbf{s}_c, \mathbf{e}_c) \leftarrow \beta_\eta^d \times \beta_\eta^d$
$\mathbf{y}_c = \mathbf{A}\mathbf{s}_c + \mathbf{e}_c$
$\xrightarrow{\mathbf{y}_c, \rho}$
$\mathbf{A} \sim R_q^{d\times d} := Sam(\rho)$
$(\mathbf{s}_s, \mathbf{e}_s) \leftarrow \beta_\eta^d \times \beta_\eta^d$
$\mathbf{y}_s = \mathbf{A}^T\mathbf{s}_s + \mathbf{e}_s$
$e_\sigma \leftarrow \beta_\eta$
$\sigma_s = \mathbf{y}_c{}^T\mathbf{s}_s + e_\sigma$
$v \leftarrow Signal(\sigma_s)$
$\xleftarrow{\mathbf{y}_s, v}$
$k_\sigma = \mathsf{Neg}(\sigma_s, v)$
$\sigma_c = \mathbf{s}_c^T\mathbf{y}_s$
$k_\sigma = \mathsf{Com}(\sigma_c, v)$

Fig. 9. The MLWE-based protocol

As mentioned in Section 4.3, q in MLWE is also an odd number, and we need to convert the *Signal* function to the $Signal_{lb}$ function.

Proposed Parameters

In the MLWE-based key exchange protocol designed in this paper, we propose four sets of parameters: *Challenge*, *Classic*, *Recommended* and *Paranoid*. For the four sets of parameters, we calculate the correctness and safety of the scheme respectively, and the specific results are shown in Table 8.

Table 8. Parameters for SER of key exchange from MLWE.

Scheme	n	q	d	g	DIST.	failure	primal	Dual	—K—
Challenge	256	7681	3	3	3	$2^{-115.5}$	156	155	512
Classical	256	7681	3	3	4	$2^{-69.0}$	163	161	512
Recommend	256	7681	4	4	3	$2^{-117.3}$	221	218	512
Paranoid	256	7681	4	4	4	$2^{-69.4}$	229	226	512

Performance Comparison

Table 9 shows the number of CPU cycles of our MLWE-based SER scheme under four sets of parameters. Same as the previous sections, the 'total' in the table records the average and median of the total number of CPU runs. The 'perbit' in the table records the average and median of the number of cycles used by CPU to complete a one-bit key consensus.

Table 9. Number of CPU cycles required for SER

Scheme	total median	total average	perbit median	perbit average	—K—
Challenge	523856	552900	1023	1079	512
Classical	745658	768838	1456	1501	512
Recommend	1053560	1079795	2057	2108	512
Paranoid	1413004	1438586	2759	2809	512

From the above data, it can be seen that SER also performs well in MLWE-based key exchange protocols. It not only guarantees the low failure rate and security under the primal and dual attacks [38, 19], but also significantly improves the efficiency. If a 256-bit key can be negotiated using the traditional MLWE-based key exchange protocol, then under the same number of CPU cycles, a 512-bit key can be negotiated using SER, which is more efficient.

5 Discussion

As far as SER itself is concerned, it has two significant advantages. First, SER reconciles the most significant and least significant bits of the secret value, and a two-bit key can be obtained by coordinating the two coefficients. In Frodo, only a one-bit key can be obtained by coordinating two coefficients, while in NewHope, four coefficients are required to negotiate a one-bit key. It can be seen that the error reconciliation efficiency of SER is much better than other schemes. Second, by transmitting the g-bit signal value to the communication peer, the fault tolerance interval of SER is expanded, thereby reducing the failure rate of reconciliation.

When applying SER to key exchange schemes for various difficult problems such as LWE, RLWE, and MLWE, we found that SER is versatile, adaptable to various scenarios, and compatible with various protocols. From the implementation part of Section 4, it can be seen that after integrating SER into the above key exchange protocol, the protocol can meet the requirements of failure rate and security, and has good performance.

However, it is worth noting that when SER is integrated in an LWE-based key exchange protocol, such as Frodo, under the *Challenge* parameter set, the failure rate of SER is higher than that of the error reconciliation mechanism in Frodo. There are two main reasons for this phenomenon: First, The size q in *Challenge* mode is too small, and the error distribution Dist. is too large after doubling. Second, SER doubles the error to perform key reconciliation, and the large matrix operation further magnifies the doubled error. The combination of these two reasons results in a higher failure rate of SER under *Challenge* mode. Fortunately, we can reduce the failure rate by changing the size of q. By changing q, we reduce the failure rate of SER in Section 4.2.

Observing the experiments in Section 4, we find that SER performs well in all three implementations, especially in RLWE-based protocol. As we know, SER doubles the error to reconcile the lowest bit information, which in turn leads to a large error in SER. The RLWE-based key exchange scheme adopts a polynomial ring, and by choosing a larger parameter q, the ratio of error to modulus q in SER can be reduced, thereby reducing the failure rate. In RLWE-based SER, its failure rate is only $2^{-124.4}$, and in the worst case, it can resist 2^{-198} primary attacks and 2^{-191} dual attacks. In addition, it takes an average of 204 CPU cycles to complete a 1-bit key exchange.

Fig. 10 and Fig. 11 list the relationship between the failure rate and the parameter g in LWE, RLWE, and MLWE, respectively. It can be seen that in the three schemes, the failure rate decreases with the increase of the parameter g. Among them, the failure rate of the protocol constructed based on the RLWE problem is much smaller than other protocols. The main reason is that g represents the number of bits of the auxiliary signal value sent by Bob, and the increase of g indicates that the amount of information sent increases, thereby reducing the failure rate during reconciliation.

6 Conclusion

With the goal of improving the efficiency and success rate of error coordination, this paper proposes an efficient and general error reconciliation scheme: Speedy Error Reconciliation (SER). Compared with other schemes, SER has three notable features. First, it is more efficient. The most significant and least significant bits of each coefficient can be used for reconciliation at the same time, and a two-bit key can be obtained for one reconciliation. Second, the reconciliation failure

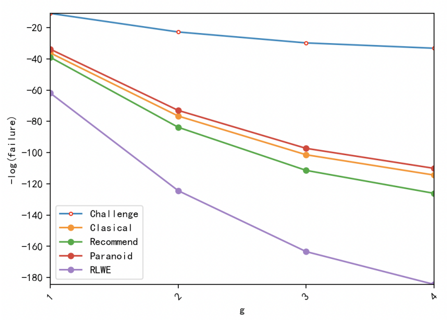


Fig. 10. The relationship between failure rate and g in LWE and RLWE schemes

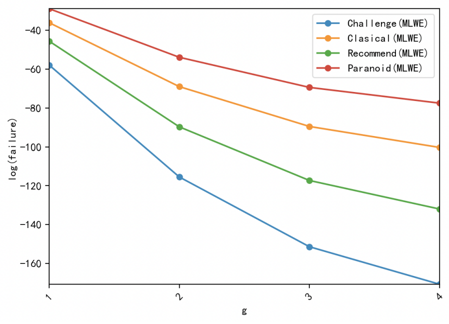


Fig. 11. The relationship between the failure rate and g in MLWE scheme

rate is lower. By sharing g-bit auxiliary information between two communication peers, SER expands the fault tolerance interval when negotiating keys, thereby improving the correct rate of negotiation. Third, it is universal and can be applied to key exchange protocols based on a variety of difficult problems. By integrating SER into LWE, RLWE, and MLWE-based key exchange protocols, we compare the performance of SER in various protocols, such as failure rate, security strength, and the number of CPU cycles required for total and perbit keys, and find that SER can be competent in various scenarios, especially in RLWE. After replacing the error coordination mechanism with SER, compared with the original protocol, the per-bit key negotiation efficiency of the replaced Frodo and Newhope is improved by 61.6% and 797.6%, respectively.

References

1. Gunther, F., Towa, P.: KEMTLS with delayed forward identity protection in (almost) a single round trip. Cryptology ePrint Archive, Report 2021/725
2. Hamid Nejatollahi, Nikil Dutt, Sandip Ray, Francesco Regazzoni, Indranil Banerjee, and Rosario Cammarota. 2017. Software and hardware implementation of lattice-cased cryptography schemes. University of California Irvine, CECS TR 17-04 (2017).
3. Peter W. Shor. 1997. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Journal of Computing (1997).
4. Gui-Lu Long. 2001. Grover algorithm with zero theoretical failure rate. Physical Review A (2001).
5. Shor, P. W. (1999). Polynomial-time algorithms for prime factoriza- tion and discrete logarithms on a quantum computer. SIAM Review, 41(2), 303–332.
6. Ronald L. Rivest, Adi Shamir, Leonard M. Adleman: A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. Commun. ACM 21(2): 120-126 (1978)
7. Grover, L. K. (1996). A fast quantum mechanical algorithm for database search. arXiv preprint arXiv:quant-ph/9605043.
8. NIST: Submission requirements and evaluation criteria for the post-quantum cryptography standardization process. Official Call for Proposals, National Institute for

Standards and Technology, December 2016, http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/call-for-proposals-final-dec-2016.pdf

9. Bos, Joppe W. et al. “Frodo: Take off the Ring! Practical, Quantum-Secure Key Exchange from LWE.” Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (2016): n. pag.
10. Albrecht, Martin R. et al. “On the concrete hardness of Learning with Errors.” Journal of Mathematical Cryptology 9 (2015): 169 - 203.
11. Lyubashevsky, Vadim et al. “A Toolkit for Ring-LWE Cryptography.” IACR Cryptol. ePrint Arch. (2013).
12. Micciancio, Daniele and Chris Peikert. “Hardness of SIS and LWE with Small Parameters.” IACR Cryptol. ePrint Arch. 2013 (2013): 69.
13. Bos, Joppe W. et al. “Post-Quantum Key Exchange for the TLS Protocol from the Ring Learning with Errors Problem.” 2015 IEEE Symposium on Security and Privacy (2015): 553-570.
14. Mehic, Miralem et al. “Error Reconciliation in Quantum Key Distribution Protocols.” Selected Results of the COST Action IC1405 (2020).
15. Fucai Luo, Fuqun Wang, Kunpeng Wang, Jie Li, Kefei Chen: LWR-Based Fully Homomorphic Encryption, Revisited. Secur. Commun. Networks 2018: 5967635:1-5967635:12 (2018)
16. Jan-Pieter D’Anvers, Angshuman Karmakar, Sujoy Sinha Roy, Frederik Vercauteren: Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. AFRICACRYPT 2018: 282-305
17. Long Chen, Zhenfeng Zhang, Zhenfei Zhang:
18. Song Bian, Masayuki Hiromoto, Takashi Sato: Filianore: Better Multiplier Architectures for LWE-based Post-Quantum Key Exchange. DAC 2019: 113
19. Ducas, Léo and Alain Durmus. “Ring-LWE in Polynomial Rings.” IACR Cryptol. ePrint Arch. (2012).
20. Gueron, Shay and Fabian Schlieker. “Speeding up R-LWE Post-quantum Key Exchange.” IACR Cryptol. ePrint Arch. 2016 (2016): 467.
21. Bogdanov, Andrej et al. “On the Hardness of Learning with Rounding over Small Modulus.” TCC (2016).
22. Cheon, Jung Hee et al. “Lizard: Cut off the Tail! // Practical Post-Quantum Public-Key Encryption from LWE and LWR.” IACR Cryptol. ePrint Arch. 2016 (2016): 1126.
23. Ding, Jintai. “A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem.” IACR Cryptol. ePrint Arch. 2012 (2012): 688.
24. Lyubashevsky, Vadim et al. “On Ideal Lattices and Learning with Errors over Rings.” IACR Cryptol. ePrint Arch. 2012 (2013): 230.
25. Peikert, Chris. “Lattice Cryptography for the Internet.” IACR Cryptol. ePrint Arch. 2014 (2014): 70.
26. Alkim, Erdem et al. “Post-quantum Key Exchange - A New Hope.” USENIX Security Symposium (2016).
27. Saarinen, MJ.O. (2018). HILA5: On Reliability, Reconciliation, and Error Correction for Ring-LWE Encryption. In: Adams, C., Camenisch, J. (eds) Selected Areas in Cryptography - SAC 2017. SAC 2017. Lecture Notes in Computer Science(), vol 10719.
28. Jin, Zhengzhong and Yunlei Zhao. “Optimal Key Consensus in Presence of Noise.” IACR Cryptol. ePrint Arch. 2017 (2017): 1058.
29. J. Bos et al., ”CRYSTALS - Kyber: A CCA-Secure Module-Lattice-Based KEM,” 2018 IEEE European Symposium on Security and Privacy (EuroS&P), 2018, pp. 353-367, doi: 10.1109/EuroSP.2018.00032.

30. D'Anvers, JP., Karmakar, A., Sinha Roy, S., Vercauteren, F. (2018). Saber: Module-LWR Based Key Exchange, CPA-Secure Encryption and CCA-Secure KEM. In: Joux, A., Nitaj, A., Rachidi, T. (eds) Progress in Cryptology – AFRICACRYPT 2018. AFRICACRYPT 2018. Lecture Notes in Computer Science(), vol 10831. Springer, Cham.
31. Schwabe, P., Stebila, D., Wiggers, T.: Post-Quantum TLS Without Handshake Signatures. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 1461–1480
32. Schwabe, P., Stebila, D., Wiggers, T.: More efficient post-quantum KEMTLS with pre-distributed public keys. In: ESORICS 2021. LNCS, Springer
33. Ding, J., Alsayigh, S., Lancrenon, J., RV, S., Snook, M.: Provably secure password authenticated key exchange based on RLWE for the post-quantum world. In Cryptographers' Track at the RSA conference (pp. 183-204). Springer, Cham.
34. Gao, X., Ding, J., Li, L., Saraswathy, R. V., Liu, J.: Efficient implementation of password-based authenticated key exchange from RLWE and post-quantum TLS. Cryptology ePrint Archive.
35. Yang, Y., Gu, X., Wang, B., Xu, T.: Efficient Password-Authenticated Key Exchange from RLWE Based on Asymmetric Key Consensus. In International Conference on Information Security and Cryptology (pp. 31-49). Springer, Cham.
36. Liu, C., Zheng, Z., Jia, K., You, Q. (2019, November). Provably secure three-party password-based authenticated key exchange from RLWE. In International Conference on Information Security Practice and Experience (pp. 56-72). Springer, Cham.
37. Adrian, David et al. "Imperfect Forward Secrecy: How Diffie-Hellman Fails in Practice." Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (2015): n. pag.
38. Shor, Peter W.. "Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer." SIAM J. Comput. 26 (1999): 1484-1509.
39. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. J. ACM, 56(6), 2009. Preliminary version in STOC 2005.
40. Langlois, A., Stehlé, D. Worst-case to average-case reductions for module lattices. Des. Codes Cryptogr. 75, 565–599 (2015). https://doi.org/10.1007/s10623-014-9938-4
41. Mehic, Miralem et al. "Error Reconciliation in Quantum Key Distribution Protocols." Selected Results of the COST Action IC1405 (2020).
42. Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. SIAM Journal on Computing, volume 38, issue 1, pages 97-139, 2008.
43. Ali Ansarmohammadi, Saeed Shahinfar, and Hamid Nejatollahi. 2015. Fast and area efficient implementation for chaotic image encryption algorithms. In CADS.
44. Diffie, W., Hellman, M.: New directions in cryptography. IEEE transactions on Information Theory 22(6), 644–654