Comparison analysis and efficient implementation of reconciliation-based RLWE key exchange protocol

Xinwei Gao

Beijing Key Laboratory of Security and Privacy in Intelligent Transportation,
Beijing Jiaotong University,
No.3 ShangYuanCun, HaiDian District, Beijing, 100044, P.R.China
Email: xinweigao@bjtu.edu.cn

Jintai Ding*, Saraswathy RV

Department of Mathematical Sciences, University of Cincinnati,
French Hall, West, 2815 Commons Way, Cincinnati, Ohio, 45219, United States
Email: jintai.ding@gmail.com
Email: rvsaras86@gmail.com
*Corresponding author

Lin Li, Jiqiang Liu

Beijing Key Laboratory of Security and Privacy in Intelligent Transportation,
Beijing Jiaotong University,
No.3 ShangYuanCun, HaiDian District, Beijing, 100044, P.R.China
Email: lilin@bjtu.edu.cn
Email: jqliu@bjtu.edu.cn
*Corresponding author

Abstract: Error reconciliation is an important technique for Learning With Error (LWE) and Ring-LWE (RLWE)-based constructions. In this paper, we present a comparison analysis on two error reconciliation-based RLWE key exchange protocols: Ding et al. in 2012 (DING12) and Bos et al. in 2015 (BCNS15). We take them as examples to explain core idea of error reconciliation, building key exchange over RLWE problem, implementation, real-world performance and compare them comprehensively. We also analyse a LWE key exchange "Frodo" that uses an improved error reconciliation mechanism in BCNS15. To the best of our knowledge, our work is the first to present at least 128-bit classic (80-bit quantum) and 256-bit classic (>200-bit quantum) secure parameter choices for DING12 with efficient portable C/C++ implementations. Benchmark shows that our efficient implementation is 11x faster than BCNS15 and one key exchange execution only costs 0.07ms on a 4-year-old middle range CPU. Error reconciliation is 1.57x faster than BCNS15.

Keywords: RLWE, Post Quantum, Key Exchange, Implementation, Analysis

Biographical notes: Xinwei Gao received B.S degree from Beijing Jiaotong University in 2014. He is a Ph.D. student at Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. He is currently a visiting student at University of Cincinnati with sponsorship from China Scholarship Council. His research interest include post-quantum cryptography and RLWE-based key exchange.

Jintai Ding received Ph.D. degree in Yale in 1995. He is currently a professor of Mathematics at the University of Cincinnati. He was Humboldt fellow and visiting professor at TU Darmstadt in 2006-2007. He received the Zhong Jia Qing Prize from the Chinese math society in 1990. His research interest lies in post-quantum cryptography (PQC). He was a co-chair of the second international workshop on PQC. He and his colleagues invented LWE and RLWE-based key exchange protocols, Rainbow signature, "GUI" HFEV- signature and Simple Matrix encryption.

Saraswathy RV received bachelor degree in mathematics from University of Madras, India in 2006. She worked as a software quality assurance engineer in information technology industry for a few years till 2011 and is now a Ph.D. candidate in mathematics at University of Cincinnati. Her current research interests include lattice based cryptography, Learning with Errors and key exchange using RLWE.

Lin Li received Ph.D. degree from Shandong University in 2007. She is currently an assistant

professor at Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. Her current research interests include cryptography and privacy preserving.

Jiqiang Liu received B.S. and Ph.D. degree from Beijing Normal University in 1994 and 1999 respectively. He is currently a professor at the Beijing Key Laboratory of Security and Privacy in Intelligent Transportation at Beijing Jiaotong University. He is also the vice dean of graduation school of Beijing Jiaotong University. His research interests include security protocols, trusted computing and privacy preserving.

1 Introduction

The groundbreaking Diffie-Hellman key exchange protocol was proposed in "New Direction in Cryptography" in 1976 [1]. It has been over 40 years since Diffie-Hellman key exchange was proposed. We have seen numerous similar cryptosystems and real world deployments. Most current public key cryptosystems are constructed from classical hard mathematical problems in number theory (e.g., integer factorization problem, discrete logarithm problem or elliptic curve variant etc.). They are hard to break with securely chosen parameter and implementation on classical computers. However, with the advent of large quantum computers, such public key algorithms are vulnerable and can be broken practically. Efficient quantum algorithms (e.g., Shor's algorithm [2]) can solve these hard problems and therefore break most current public key cryptosystems. Experiments had shown great undiscovered potential of quantum computers. Therefore, it is imperative to focus on designing novel cryptography constructions for coming quantum world.

Post-quantum cryptography has become one of the forefront research topics during the past years. These cryptosystems are thought to be secure against classic and quantum computers. There are several approaches to realize post-quantum cryptosystems, including: lattice-based, multivariate-based, hash-based, code-based etc. Various extremely versatile and efficient constructions had come to horizon in recent years, including: encryption, signature, key exchange, identity-based encryption, fully homomorphic encryption, multi-linear maps etc. In 2015, National Security Agency announced their plan to switch to quantum-resistant cryptography primitives in near future. At PQCrypto 2016 conference, National Institute of Standards and Technology (NIST) announced formal call for quantum-resistant cryptographic algorithms for future and their plans for post-quantum cryptography standards. It is estimated that draft of post-quantum cryptography standards will be ready within 10 years after the submission.

1.1 LWE & RLWE key exchange

Regev introduced Learning With Errors (LWE) problem in 2005 [3]. It is an average-case problem with strong security guarantee and high efficiency when parameters are properly chosen. Lyubashevsky et al. introduced its ring variant, Ring Learning With Errors (RLWE) in 2010 [4]. Hardness of LWE and RLWE can be reduced to solving hard problems in regular lattice and ideal lattice respectively. Since no classic or quantum algorithms can solve lattice problems and their versatility, LWE and RLWE are considered as important building blocks for post-quantum cryptography. Several proof-of-concept implementations of real world application that adopt LWE/RLWE-based cryptography algorithms have been proposed, including post-quantum TLS, VPN etc.

An important line of post-quantum cryptography is post-quantum key exchange. Key exchange is crucial for real world applications since data that transferred between parties is encrypted using symmetric encryption algorithms (e.g., AES), not public key encryption algorithms since they are too costly. Encryption key for symmetric encryption algorithms is derived from key exchange, therefore secure and efficient key exchange protocol is very important. Key exchange protocols based on RLWE problem are considered to be most secure and efficient among all approaches to achieve post-quantum key exchange. In the case of RLWE, the biggest problem for key exchange is how to agree on an identical value (or session key) since shared key computation value perturbed by small error terms.

First Diffie-Hellman like LWE and RLWE-based key exchange protocols were introduced in 2012. We have seen various works that take advantage of LWE or RLWE to design and implement post-quantum key exchange protocols. These constructions cover unauthenticated and authenticated key exchange, Diffie-Hellman-like and HMQV-like, explicit and implicit authenticated etc.

1.2 Related works

Ding et al. presented first LWE and RLWE-based key exchange protocols in 2012 [5] (denoted as DING12). This work gives analogues of classical Diffie-Hellman key exchange over LWE and RLWE. Main contribution of this work is giving the first LWE and RLWE key exchange constructions with a novel error reconciliation mechanism. This reconciliation mechanism utilizes a "signal" value to assist error reconciliation over approximately equal values and extracts uniformly random bits in order to generate same session key both parties. It also enjoys high error tolerance and efficiency. No parameter choice or implementation is provided in this work.

There are a few variants that share similar general structure and the idea of error reconciliation using signal value: A RLWE exchange protocol was given by Peikert in 2014 [6] (denoted as PKT14). This work gives a slightly modified error reconciliation mechanism over DING12. PKT14 is instantiated by Bos et al. at IEEE S&P 2015 in [7] with 128-bit secure parameters and implementation (denoted as BCNS15). BCNS15 also integrates their implementation

into OpenSSL as post-quantum TLS ciphersuite. A highly optimized version of BCNS15 was proposed by Alkim et al. at USENIX 2016 [8] (denoted as NewHope). This work improves BCNS15 by adopting alternative error reconciliation mechanism, more compact parameter choices, efficient implementation and other improvements. They claim that their portable and AVX2-optimized implementation achieve 9x and 24x performance boost over BCNS15 respectively.

A LWE-based key exchange protocol "Frodo" was proposed by Bos et al. in 2016 [9]. This work modifies error reconciliation in BCNS15 to extract more bits from each coefficient in order to choose smaller parameters and reduce communication cost. Performance of this work is 8x slower than NewHope and communication cost is 4.7x larger.

For authenticated key exchange (AKE) protocol, a RLWE-based protocol which is a RLWE analogue of HMQV was proposed by Zhang et al. in 2015 [10]. Two provably secure password-based authenticated key exchange (PAKE) protocols that are RLWE analogues of PAK and PPK were given in 2017 [11]. These two works use same error reconciliation mechanism as DING12.

1.3 Contribution

In this paper, we first present a comprehensive comparison analysis on two error reconciliation-based RLWE key exchange protocols: DING12 and BCNS15. Our analysis and comparison efforts are focusing on protocol design and error reconciliation mechanism. Error reconciliation mechanism is the major difference between these two protocols. In addition, error reconciliation of Frodo LWE key exchange is also analysed and compared since it uses a modified error reconciliation in BCNS15. Principle and realization of error reconciliation in these key exchange protocols are well explained and compared. We conclude that they share similar protocol structure and error reconciliation mechanism.

Second, we instantiate DING12 with two parameter choices. To the best of our knowledge, our work is the first to instantiate and implement this protocol with two practical and common parameter choices efficiently. Both parameter choices choose same degree $n = 1024$ for ring R_q. First parameter choice P30 is 128-bit classic (80-bit quantum) secure and adopts a 30-bit prime which is smaller than BCNS15 (32-bit modulus) and discrete Gaussian distribution for sampling with same standard deviation $\sigma = 3.192$ as BCNS15. Second parameter choice P14 is 256-bit classical (>200-bit quantum) secure which adopts same 14-bit prime $q = 12289$ and centered binomial distribution for sampling with parameter $k = 16$ as [8]. Efficient portable C/C++ implementations of P30 and P14 are also provided. For complete key exchange execution, implementation of P30 and P14 is 11x and 3.94x faster than BCNS15 respectively, error reconciliation part is 1.57x faster. Our results show that DING12 can be instantiated with compact parameters and implemented truly efficiently. We also present comparison and discussion between our implementations and BCNS15.

We believe that RLWE-based key exchange protocols and our implementations are truly practical toward real-world applications for the upcoming post-quantum world.

1.4 Organization

In section 2, we recall background knowledge on LWE and RLWE. In sections 3, we present detailed analysis and comparisons between DING12, BCNS15 and Frodo. We discuss and compare overall protocol structure and error reconciliation mechanism of these protocols. Principle and differences between DING12, BCNS15 and Frodo on error reconciliation are well explained. In section 4, we introduce two parameter choices for DING12 RLWE key exchange - P30 and P14. Efficient implementations, classic and quantum security level analysis, performance measurements of standalone operations and overall protocol execution, communication cost. Discussions are given in section 5. We conclude the paper in section 6.

2 LWE & RLWE

LWE problem was introduced by Oded Regev in 2005 [3]. It is an average-case problem and the security of LWE can be reduced to solving worst-case hard lattice problems. Currently there are no publicly known algorithms can solve LWE or underlying lattice problems. LWE distribution $A_{s,\chi}$ over $Z_q^n \times Z_q$ is generated by uniformly random sampled $a \in Z_q^n$, small secret vector $s \in Z_q^n$ and error vector $e \in Z_q^n$. e is sampled from some distribution χ (common choice is discrete Gaussian distribution), outputs $(a, b = a \cdot s + e \bmod q)$. There are two versions of LWE problem: search and decision. Search-LWE problem is to recover fixed secret term s given various LWE samples and decision-LWE problem is to distinguish LWE samples from uniform randomly generated ones. Properly chosen parameters, perturbation from secret error term and indistinguishable from uniform random keep LWE problem very hard to solve.

Lyubashevsky et al. introduced RLWE (Ring-LWE) problem in 2010 [4]. It is the ring analogue of LWE. RLWE problem is much more efficient than LWE and its hardness can be reduced to solving hard problems in ideal lattices, which is also very hard to solve on classical and quantum computers currently when secure parameters are chosen properly. Define ring $R = Z[x]/f(x)$, where $f(x) = x^n + 1$ with n be a power of 2. Modulus q defines quotient ring $R_q = R/qR$. Let ring R be cyclotomic ring and χ be discrete Gaussian distribution. Define $\|a\|_\infty = \max\{|a_i|\}$, which is the l_∞ norm of a. RLWE distribution $A_{s,\chi}$ over $R_q \times R_q$ is sampled by uniformly random chosen vector $a \in R_q$, small secret vector $s \in R_q$, error term $e \in R_q$ is sampled from discrete Gaussian distribution, outputs $(a, b = a \cdot s + e \bmod q)$. RLWE also has search and decision version as LWE and we omit details here. If one can solve decision version of RLWE problem, then he can also solve search version of RLWE problem and underlying hard lattice problems.

For security of LWE and RLWE problems, they can both be reduced to hard problems in lattice, which are very hard to solve with properly instantiated instances. LWE and RLWE problems can be reduced to lattice problems including: Shortest Vector Problem (SVP), Closest Vector Problem (CVP), Bounded Distance Decoding (BDD) etc. These problems are hard for both classical and quantum computers, which serve as solid foundation for post-quantum cryptography. Moreover, RLWE-based constructions are extremely efficient while key size is larger than current public key cryptosystems.

Compared with RLWE-based constructions, LWE-based ones are considered to have much larger key size (at least quadratic) and slower computation due to structure of LWE (large matrix). It is believed that RLWE shares same security level with LWE since no attack takes advantage of ring structure. LWE and RLWE are extremely versatile for cryptographic constructions. They served as fundamental building blocks for modern lattice-based constructions, including encryption, digital signature, key exchange, homomorphic encryption, attribute-based encryption etc.

3 Analysis of reconciliation-based RLWE key exchange

There are two major approaches to realize RLWE-based key exchange: one is error reconciliation-based, the other is key encapsulation mechanism (KEM)-based. In [6], they claimed that an important advantage of reconciliation-based protocols is reduced bandwidth with nearly halve the ciphertext size, therefore they are more attractive. Additional advantage of error reconciliation-based Diffie-Hellman like protocols is forward security. Encryption-based approaches to achieve key exchange are more "static" like RSA key exchange. Once long-term private key is revealed, attacker can decrypt and recover all past messages. Latest draft of TLS 1.3 has removed support of RSA due to this reason.

The reason why we need error reconciliation mechanism is the construction of key exchange over LWE/RLWE problem. Generally we have the following construction: both parties generate public key $pk = a \cdot s + e$, then compute key exchange material $k' = pk \cdot s' + e_1 = ass' + es' + e_1$. Note that s, s', e, e', e_1 are error terms that sampled from some distribution. For party i and j, k_i cannot rigorously equal to k_j since if we expand k_i and k_j, we can see that $k_i - k_j = es' + e_1 - e's - e_2$. The difference between k_i and k_j is small therefore they are approximately equal, unlike $(g^a)^b \bmod p = (g^b)^a \bmod p$ in Diffie-Hellman key exchange. Therefore there has to be some mechanism to reconcile the errors in order to agree on same key.

Figure 1 DING12 RLWE key exchange protocol.

In this section, we recall and compare error reconciliation techniques in literatures. Our major focus is on two major reconciliation-based RLWE key exchange protocols: RLWE version of DING12 and BCNS15. A LWE-based key exchange called Frodo is also analysed since it uses a slightly modified error reconciliation in BCNS15 to realize LWE key exchange. Comprehensive analysis on error reconciliation mechanism is an important contribution and focus of this work. We compare protocol design, error reconciliation mechanism, implementation and performance of these three protocols. First two parts are discussed in this section, the rest are presented in section 4. We believe that DING12, BCNS15 and Frodo LWE/RLWE key exchange protocols adopt similar error reconciliation mechanism to reconcile errors in order to agree on same key for both parties.

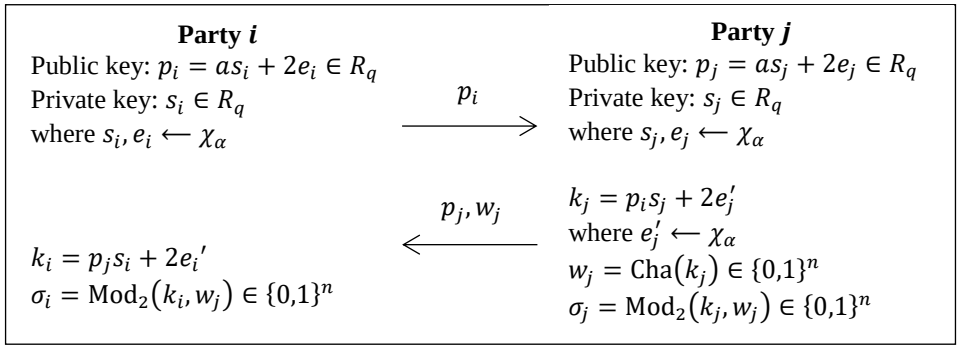
3.1 DING12 RLWE key exchange

3.1.1 Introduction

In 2012, Ding et al. introduced LWE and RLWE-based key exchange protocols which are analogues of classic Diffie-Hellman key exchange [5]. Both LWE and RLWE-based key exchange protocols in DING12 are quantum-resistant, practically efficient and provably secure. This work introduces an error reconciliation mechanism that can cancel out differences between approximately equal values. It generates a serial of uniformly distributed 0 and 1 bits called "signal", which is a binary string that indicates which region does each coefficient of key exchange material belongs to. Signal is computed by one side and sent to the other party to reconcile errors.

The motivation of using signal is that in order to agree on identical value over approximately equal values, one intuitive approach is to mod 2 on each coefficient of key exchange material for both parties simultaneously. However, due to difference between key exchange materials is even and representation of Z_q is $\{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$, mod 2 on both sides cannot generate same value for all cases (e.g., boundary of Z_q). This is where the importance of signal is highlighted. Signal value implies which region does k_j belongs to, therefore the other party can decide whether adding $\frac{q-1}{2}$ to its own value or not in order to keep the difference to be even and agree on same key. In order to ensure correctness of key exchange with overwhelming probability, the difference between key exchange materials of two parties (i.e., error tolerance) is bounded. The approach to realize this is to control modulus q carefully.

Figure 1 illustrates DING12 RLWE key exchange protocol:



3.1.2 Error reconciliation

Here we recall error reconciliation mechanism:

Signal Function. For prime $q > 2$, hint functions $\sigma_0(x), \sigma_1(x)$ from Z_q to $\{0,1\}$ are defined as: $\sigma_0(x) = \begin{cases} 0, x \in [-\lfloor \frac{q}{4} \rfloor, \lfloor \frac{q}{4} \rfloor] \\ 1, otherwise \end{cases}$; $\sigma_1(x) = \begin{cases} 0, x \in [-\lfloor \frac{q}{4} \rfloor + 1, \lfloor \frac{q}{4} \rfloor + 1] \\ 1, otherwise \end{cases}$.

Signal function. $\mathrm{Cha}()$ is defined as: For any $y \in Z_q$, $\mathrm{Cha}(y) = \sigma_b(y)$, where $b \xleftarrow{\$} \{0,1\}$. We denote the coefficient is in "inner region" if $\mathrm{Cha}(y) = 0$. Otherwise, coefficient lies in "outer region".

Robust Extractor. The robust extractor is defined as: $\mathrm{Mod}_2(x, w) = \left(x + w \cdot \frac{q-1}{2} \bmod q\right) \bmod 2$.

$\mathrm{Mod}_2()$ is a robust extractor on Z_q with error tolerance δ with respect to signal w derived from function $\mathrm{Cha}()$, if the following holds:

Deterministic algorithm $\mathrm{Mod}_2()$ takes input $x \in Z_q$ and signal $w \in \{0,1\}$, outputs $k = \mathrm{Mod}_2(x, w) \in \{0,1\}$.

Signal function $\mathrm{Cha}()$ takes input $y \in Z_q$ and outputs signal $w \leftarrow \mathrm{Cha}(y) \in \{0,1\}$.

For any $x, y \in Z_q$ such that $x - y$ is even and $||x - y|| \leq \delta$, then it holds that $\mathrm{Mod}_2(x, w) = \mathrm{Mod}_2(y, w)$, where $w \leftarrow \mathrm{Cha}(y)$.

The robust extractor is designed to guarantee the correctness of key exchange. One party sends a signal to the other party and utilize the error reconciliation mechanism (i.e., robust extractor) to agree on an identical value. Both parties compute final shared key using this robust extractor.

Lemma 1. Let $q > 8$ be an odd integer, the function $\mathrm{Mod}_2()$ defined above is a robust extractor with respect to S with error tolerance $\delta = \frac{q}{4} - 2$.

Lemma 2. For any odd $q > 2$, if x is uniformly random in Z_q, then $\mathrm{Mod}_2(x)$ is uniformly random conditioned on w, where $w \leftarrow \mathrm{Cha}(x)$.

Lemma 3. If $8n\beta^2 \leq \frac{q}{4} - 2$, then $sk_i = sk_j$ with overwhelming probability.

For detailed proofs, please refer to [5].

The reason of introducing signal function $\mathrm{Cha}()$ is to indicate which region (inner or outer) does coefficient of k_j belongs to. It serves as input for $\mathrm{Mod}_2()$ function. If $\mathrm{Cha}(y) = 0$, i.e., coefficient is in inner region, we can simply mod 2 simultaneously for both parties using $\mathrm{Mod}_2()$ with high confidence. Despite adding small error term might result in the value of k_i/k_j jumping from inner region to outer region, this only happens with very low probability. The main problem lies on error reconciliation is the case where coefficient of k_i/k_j is in outer region, i.e., $\mathrm{Cha}(y) = 1$. In this case, we need to "pull" the coefficient from outer region to inner region by adding $(q-1)/2$, then coefficients are in inner region therefore we can mod 2 simultaneously using $\mathrm{Mod}_2()$ to agree on same key with overwhelming probability. We can take outer region as buffer area for $k_i - k_j$ since as long as $|k_i - k_j|$ is smaller than half size of buffer area, reconciliation mechanism works and both parties can derive same key. Output size of signal function $\mathrm{Cha}()$ and $\mathrm{Mod}_2()$ is n bit, i.e., each coefficient of k_j generates 1 bit reconciliation and key information.

3.2 BCNS15 RLWE key exchange

3.2.1 Introduction

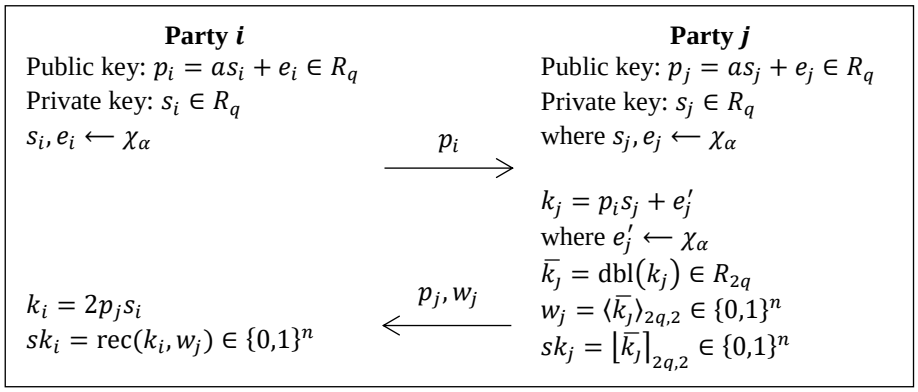
In 2015, Bos et al. instantiated PKT14 RLWE key exchange protocol [6] in [7]. BCNS15 gives parameter choice of PKT14 aiming at 128-bit security and proof-of-concept implementation. They also integrate it into TLS and constructs post-quantum TLS ciphersuite. Generally, BCNS15 (i.e., PKT14) and DING12 share same similar protocol structure, techniques and slightly different error reconciliation mechanism.

Here we recall error reconciliation mechanism of BCNS15 (i.e., PKT14):

Define cross-rounding function $\langle \cdot \rangle_{q,2}: Z_q \to Z_2, x \mapsto \langle x \rangle_{q,2} = \left\lfloor \frac{4}{q} x \right\rceil \bmod 2$, modular rounding function $\lfloor \cdot \rceil_{q,2}: Z_q \to Z_2, x \mapsto \lfloor x \rceil_{q,2} = \left\lfloor \frac{2}{q} x \right\rceil \bmod 2$. When extended to elements of R_q coefficient-wise: for $f \in R_q$, $\lfloor f \rceil_{q,2} = (\lfloor f_{n-1} \rceil_{q,2}, \lfloor f_{n-2} \rceil_{q,2}, \cdots, \lfloor f_0 \rceil_{q,2})$, $\langle f \rangle_{q,2} = (\langle f_{n-1} \rangle_{q,2}, \langle f_{n-2} \rangle_{q,2}, \cdots, \langle f_0 \rangle_{q,2})$. If modulus q is odd, it requires working in Z_{2q} instead of Z_q to avoid bias in derived bits. Randomized doubling function $\mathrm{dbl}(): Z_q \to Z_{2q}, x \mapsto \mathrm{dbl}(x) = 2x - e$, where e is sampled from $\{-1,0,1\}$ with probability $p_{-1} = p_1 = \frac{1}{4}$, $p_0 = \frac{1}{2}$. Reconciliation function $\mathrm{rec}(): Z_{2q} \times Z_2 \to Z_2$ is defined by $\mathrm{rec}(w, b) = \begin{cases} 0, \ if \ w \in I_b + E \bmod 2q \\ 1, \ otherwise \end{cases}$ where $I_0 = \{0, 1, \cdots, \lfloor \frac{q}{2} \rfloor - 1\}$ and $I_1 = \{-\lfloor \frac{q}{2} \rfloor, \cdots, -1\}$, $E = [-\frac{q}{4}, \frac{q}{4})$. For details and proofs, please refer to [6] and [7].

Figure 2 illustrates BCNS15 key exchange protocol:

Figure 2 BCNS15 (PKT14) RLWE key exchange protocol.



It is clear to see that BCNS15 and DING12 RLWE key exchange share similar paradigm, except that DING12 adds $2e$ when computing public key and k_j. Another difference is that in BCNS15, party i does not add additional error term to $2p_j s_i$.

3.2.2 Error reconciliation and comparison

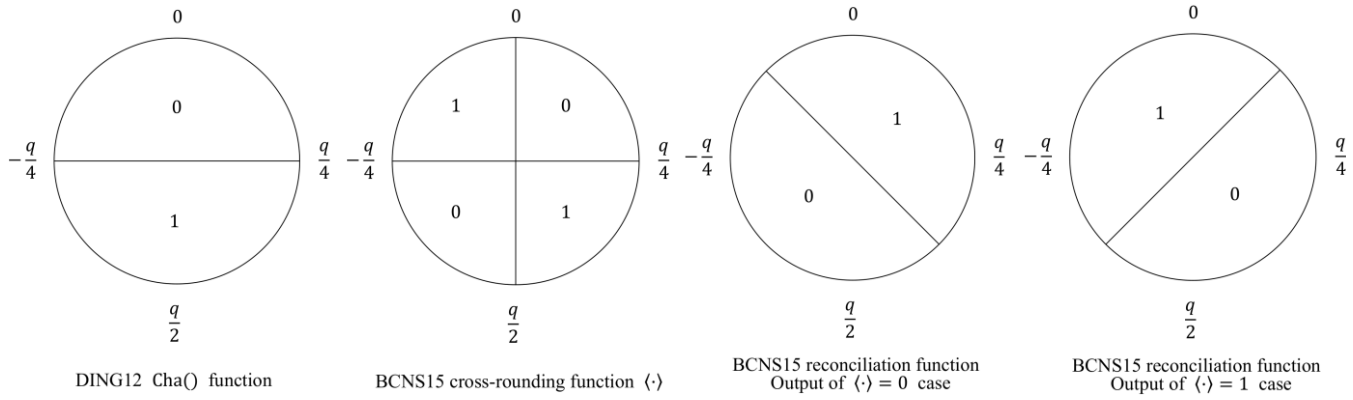
Major difference between DING12 RLWE key exchange protocol and BCNS15 is error reconciliation mechanism. As recalled in section 3.1, signal function $\mathrm{Cha}()$ is used to derive bits that indicates which region k_j lies in. Robust extractor reconciles small error for both parties and derives same shared key. We note that in the case of the signal function, we divide Z_q into two regions as E and E^c (complement of E). Suppose q is odd, then there is a bias in the output of the signal function, since there are more numbers in E than E^c. However, this bias is removed by randomized signal functions, i.e., $\sigma_0(x)$ and $\sigma_1(x)$ in DING12. Same idea is also applied using $\mathrm{dbl}()$ function in BCNS15.

Now we compare error reconciliation mechanism of DING12 and BCNS15. As defined above, cross rounding function $\langle \cdot \rangle$ provides the functionality of signal function $\mathrm{Cha}()$ as in DING12. As a result of applying the cross rounding function on each coordinate of vector k_j, we receive the bits corresponding to whether each value lies in $I_0 \cup I_0'$ or $I_1 \cup I_1'$. Sets I_0, I_1, I_0', I_1' are defined as follows: $I_0 = \left\{0,1,\dots,\left\lfloor \frac{q}{4} \right\rfloor - 1\right\}$, $I_1 = \left\{-\left\lfloor \frac{q}{4} \right\rfloor - 1,\dots,-1\right\}$ and $I_0' = \frac{q}{2} + I_0$, $I_1' = \frac{q}{2} + I_1$.

There are two cases to apply the rounding function depending on whether q is odd or even. If q is even, then there is no bias since the regions are divided equally. If q is odd, this results in a bias in generated bits. To overcome this, they switch to operations in $2q$ using $\mathrm{dbl}()$ function and then moves back to q. This technique is similar as $\sigma_0()$ and $\sigma_1()$ in DING12. Error tolerance δ is $q/4$ since if $||k_i - k_j||_\infty > q/4$, key bit will jump to other regions, therefore final shared key does not agree. Lemma of [7] explain this claim.

Figure 3 gives intuitive comparison of signal and reconciliation functions in DING12 and BCNS15:

Figure 3 Comparison of $\mathrm{Cha}()$ in DING12, $\langle \cdot \rangle_{q,2}$ and two cases of $\lfloor \cdot \rceil_{q,2}$ in BCNS15.



To sum up, $\mathrm{dbl}()$ function is designed to avoid bias in generated keys, which is a variant of hint functions $(\sigma_0(x), \sigma_1(x))$ in DING12. For generating signal, it is denoted as cross-rounding function $\langle \cdot \rangle_{q,2}$ in BCNS15, which is a modified version of $\mathrm{Cha}()$ in DING12. Figure 3 demonstrates the difference between them. We can see that compared with DING12, reconciliation mechanism in BCNS15 divides Z_q into four regions instead of two in DING12, then rotate them. For generating keys, modular

rounding function $\lfloor\cdot\rceil_{q,2}$ and rec() are variants of $\mathrm{Mod}_2()$ in DING12. More precisely, DING12 extracts least significant bit (LSB) from each coefficient while BCNS15 extracts most significant bit (MSB). Output size of cross rounding function, reconciliation function and error tolerance are the same for these two works.

3.3 Modifying error reconciliation in BCNS15 towards LWE key exchange - Frodo

Frodo LWE key exchange protocol was proposed by Bos et al. at CCS 2016 [9]. It adapts a slightly modified error reconciliation mechanism in BCNS15 to construct a LWE-based key exchange protocol. They also implement Frodo key exchange with practical parameter choice and integrate into TLS as post-quantum ciphersuite like BCNS15. Since Frodo modifies error reconciliation in BCNS15 very slightly, we would like to analyse the difference.

In Frodo, they stated that their error reconciliation mechanism is a generalized version compared with BCNS15. The major difference between Frodo and BCNS15 in terms of reconciliation is that Frodo extracts more bits from each coefficient of k_j. Note that in DING12 and BCNS15, they both extract one bit from one coefficient. In Frodo, they extract B-bit from one coefficient. According to parameters suggested in Frodo ($B = 4$), they extract four bits from one coefficient of k_j. The motivation for extracting more bits is to reduce dimension of large matrix $Z_q^{m\times n}$ while preserving same 128-bit security. Matrix dimension, B and security level is bounded by $m \cdot n \cdot B \geq 256$. Similar as BCNS15, cross rounding function $\langle\cdot\rangle_{2^B}$ is defined as $\lfloor 2^{-\bar{B}+1}v \rceil \bmod 2$, rounding function $\lfloor\cdot\rceil_{2^B} = \lfloor 2^{-\bar{B}}v \rceil \bmod 2^B$, where $\bar{B} = \log_2 q - B$. For any $v \in Z_q$ represented as $[0, q)$, $\lfloor\cdot\rceil_{2^B}$ outputs B most significant bits of $\left(v + 2^{\bar{B}-1}\right) \bmod q$. Correctness of Frodo is bounded by $|v - w| < 2^{\bar{B}-2}$.

Compared with BCNS15, Frodo's reconciliation mechanism extracts B most significant bits from one coefficient in order to reduce dimension of large matrix, compared with extracting most significant bit in BCNS15 and least significant bit in DING12. We believe Frodo's error reconciliation mechanism shares similar idea as BCNS15 and DING12.

4 DING12 instantiation and implementation

4.1 Practical parameter choices

To the best of our knowledge, there is no existing work had instantiated or implemented DING12 RLWE key exchange protocol efficiently. In original DING12 paper, they did not present parameter choice, implementation or choice of error distribution to sample error terms. In this section, we instantiate DING12 RLWE key exchange with two practical parameter choices: P30 and P14.

For security estimation of our parameter choices, we use two approaches: (1) LWE estimator in [12] to estimate classical security; (2) [8] to estimate quantum security. LWE estimator gives a thorough security estimation for both LWE and RLWE-based cryptosystems. It evaluates security level of cryptosystems by computing attack complexity of exhaustive search, BKW, lattice reduction, decoding, reducing BDD to unique-SVP and meet-in-the-middle attacks. Given any parameters, LWE estimator outputs computation and space complexity of these attacks, therefore it gives a nice security estimation result for LWE and RLWE-based cryptosystems. BCNS15 also takes this approach to estimate security of their parameter choice. For quantum security estimation, [8] presents an analysis on LWE and RLWE-based cryptosystems which covers primal and dual attack for solving underlying lattice problem on quantum computer. [8] claims that this estimation is pessimistic since they only consider core SVP hardness, therefore actual attack complexity and security level should be larger than result from estimation in [8].

4.1.1 128-bit classic and 80-bit quantum secure parameter choice - P30

Recall parameter choice of BCNS15: $R_q = Z_q[x]/(x^n + 1)$ with $n = 1024$, discrete Gaussian sampling with standard deviation $\sigma = 3.192$ and modulus $q = 2^{32} - 1$. Similarly, our parameter choice P30 for DING12 chooses $n = 1024$, discrete Gaussian sampling $\sigma = 3.192$ and modulus $q = 1073479681$ (approximately 30 bits). Our parameter choice is very close to BCNS15 which can be compared relative fairly.

With estimation approaches for both classic and quantum security from LWE estimator and [8], parameter choice P30 provides at least 128-bit classic security and 80-bit quantum security.

4.1.2 256-bit classic and >200-bit quantum secure parameter choice - P14

For parameter choice P14, we choose same $n = 1024$, but utilize centred binomial distribution Ψ_k of parameter $k = 16$ for sampling and modulus $q = 12289$. This choice is exact same as [8]. They claimed that choosing Ψ_k is to achieve better performance than sampling discrete Gaussian distribution and they proved that sample from Ψ_k as secret and error term will not affect security.

We also apply same security level estimation approaches as P30 and results show that P14 provides least 256-bit classic security and 200-bit quantum security with wide margin.

4.2 Implementations, benchmark and comparison

In this section, we introduce our efficient portable DING12 implementations for parameter choice P30 and P14. The goal of our portable implementation is to provide better compatibility, allowing more devices can take advantage of the security from post-quantum key exchange and its efficiency. We want to achieve high performance with good portability, namely, our implementation can run on most Intel and AMD processors even they were released several years ago. For parameter choice P30 and P14, we provide two

different implementations. In this section, we introduce details of both implementations, benchmark of major computation operations and performance of overall key exchange.

All implementations are tested on same server equipped with 3.4GHz Intel Xeon E5-2687W v2 processor (released in 2012) and 64GB memory. This processor was released 4 years ago with middle range single core performance and it does not support AVX2 instruction set. Server runs 64-bit Ubuntu 14.04. Implementations are compiled by g++ or gcc version 4.8.4 with '-O3 -march=native -m64' flags and only run on single core without parallel computing techniques. Implementation of BCNS15 we use is the same code as they suggested in the paper.

4.2.1 DING12-P30

Our efficient portable C++ DING12 implementation for P30 parameter choice is implemented using NFLlib library [13]. NFLlib is a very fast NTT-based C++ library dedicated to ideal lattice cryptography. It includes highly optimized algorithms and programming optimizations to achieve very high performance. NTT, inverse NTT and discrete Gaussian sampling are very efficient. NFLlib also takes advantage of SSE instruction sets to improve efficiency of NTT and inverse NTT. We make full use of bitwise operations to optimize our implementation, especially in error reconciliation. We safely set the statistical distance from sampled distribution to discrete Gaussian distribution to be 2^{-128} to preserve high statistical quality and security. More details are discussed in section 5. Note that for error reconciliation, our implementation does not utilize NFLlib library or assemble-level optimizations, but only plain C++ operations to reconcile errors. BCNS15 also does not use additional library when implementing error reconciliation. Since error reconciliation mechanism of DING12 is simpler than BCNS15, we expect it will have better performance.

For BCNS15 portable C implementation, they implement key exchange protocol and integrate into TLS as post-quantum ciphersuite. They use the approach from Nussbaumer-based on recursive negacyclic convolutions to implement FFT and inverse FFT. Sampling part adopts OpenSSL's RAND_bytes function to generate 256-bit seed and use AES-CTR as PRNG function to obtain approximately 24 KiB of data. This implementation is portable since it does not use dedicated instruction sets for specified CPUs.

BCNS15 provides constant time and non-constant time version implementations. Since NFLlib does not provide constant time implementation, therefore we compare our implementation with BCNS15 non-constant time implementation. Detailed benchmark of DING12-P30 and BCNS15 is reported in table 1:

Table 1 Benchmark of DING12-P30 and BCNS15 implementation

	DING12-P30 (ms)	BCNS15 (ms)
Sampling	0.007	0.134
Polynomial multiplication	0.023	0.207
Signal computation (Cha()/⟨·⟩)	0.004	0.008
Pre-shared key generation (Mod2()/⌊·⌉)	0.003	0.003
Public key generation	0.038	0.492
Complete key exchange execution - party *i*	0.0706	0.708
Complete key exchange execution - party *j*	0.0707	0.847

Compared with DING12-P30 implementation, BCNS15 is 19.14x, 9x and 12.95x slower on sampling, polynomial multiplication and key generation respectively. Highly efficient discrete Gaussian sampling, NTT and inverse NTT implementation in NFLlib contribute to efficiency of our implementation. NFLlib also utilizes SSE instruction sets to optimize NTT and inverse NTT computation while BCNS15 does not, therefore NFLlib-based implementation is more efficient. We believe that if these two protocols are implemented with same library, performance should be relative close since these two protocols share very similar structure, error reconciliation mechanism and parameter choices.

For error reconciliation (signal and pre-shared key generation), BCNS15 is 1.57x slower. For overall key exchange execution, BCNS15 is 10.03x and 11.98x slower than DING12-P30 for party *i* and *j* respectively.

4.2.2 DING12-P14

Our efficient portable implementation of P14 is a modified version of LatticeCrypto library [15] and it is different from P30 implementation. LatticeCrypto provides a faster C implementation of NewHope RLWE key exchange with no improvements on key exchange protocol itself. LatticeCrypto implementation protects against timing and cache-timing attacks through regular, constant-time implementation of all operations on secret key material. We work on portable C implementation rather than AVX2-optimized version for better compatibility. We fork LatticeCrypto library and take advantage major parts of the code, including sampling and polynomial computations. We replace error reconciliation mechanism, message encoding/decoding and other slight differences between DING12 and NewHope carefully to adapt to DING12 design.

Detailed benchmark of DING12-P14 and LatticeCrypto is reported in Table 2:

Table 2 Benchmark of DING12-P14 and LatticeCrypto implementation

	DING12-P14 (ms)	LatticeCrypto (ms)
Sampling	0.024	0.024
NTT	0.009	0.009
Inverse NTT	0.010	0.010
Signal computation (Cha()/HelpRec())	0.002	0.010
Pre-shared key generation (Mod2()/Rec())	0.003	0.004
Public key generation	0.092	0.092
Complete key exchange execution - party i	0.139	0.113
Complete key exchange execution - party j	0.142	0.149

For signal computation and pre-shared key generation (i.e., error reconciliation), DING12-P14 is 2.8x faster than LatticeCrypto. Efficient reconciliation algorithm and implementation contribute to this result. For overall key exchange execution, performance of DING12-P14 for party i is 1.23x slower than LatticeCrypto since DING12 adds additional error term $2e_i'$ to $p_j s_i$ while NewHope does not. Performance of DING12-P14 for party j is 1.05x faster than LatticeCrypto since error reconciliation is more efficient. One more difference between DING12 and NewHope is that NewHope uses SHAKE128 XOF to generate different public parameter a while DING12 does not. This computation is very fast therefore we ignore this minor difference.

Compared with BCNS15 implementation, total execution timing for party i and j are 3.58x and 4.29x faster respectively. Since parameter choices in P30 and P14 are very different and cannot be compared directly, this result is provided for completeness and reference purposes.

5 Discussion

There are a few similarities and differences between these protocols that we would like to summarize:

(1) Error reconciliation. We have explained this part in section 3. The fundamental idea of error reconciliation is: first generate signal value, which indicates which region does each coefficient of k_j belongs to, then reconcile the errors with another function which returns uniformly distributed binary stream as key. Fundamental idea of using signal function and reconciliation using mod operation are the same for DING12, BCNS15 and Frodo. BCNS15 (i.e., PKT14) cuts Z_q region differently as DING12 and turn it around with different angles. They also extract most significant bit while DING12 extracts least significant bit. We show that they are very similar in essence and share same error tolerance. We remark that one may also derive other variants of error reconciliation with different approaches to divide Z_q or adding multiple times of error term e on $a \cdot s$ ($a \cdot s + 2e$ in DING12) or extract bits with different approaches. These approaches share same idea generally since they use signal to assist error reconciliation. We also practically show that performance of DING12 reconciliation is simpler and 1.57x faster than BCNS15.

(2) Error sampling. DING12-P30 and BCNS15 use discrete Gaussian sampling with $\sigma = 3.192$ to sample error terms. DING12-P30 implementation uses NFLlib, which adopts precomputed Cumulative Distribution Table (CDT) sampler to sample from discrete Gaussian distribution. For fixed centre sampling over Z^n, it is the fastest alternative with reasonable memory usage (typically <100KB). We set statistical distance from sampled distribution to discrete Gaussian distribution to be 2^{-128}. BCNS15 also samples from discrete Gaussian distribution with a sampler they suggested using same standard deviation $\sigma = 3.192$. DING12-P14 implementation samples from centred binomial distribution ψ_k of parameter $k = 16$, which has very close statistical distance to discrete Gaussian distribution with standard deviation $\sigma = \sqrt{16/2} \approx 2.828$ according to security analysis in [8]. According to [8], this consideration is mainly for faster performance while maintaining similar security level.

(3) Polynomial multiplication. For DING12-P30, We use NFLlib that adopts NTT with SSE optimized implementation. Performance of DING12-P30 on NTT and inverse NTT credits to highly efficient NFLlib library with SSE optimizations while BCNS15 gives plain C implementation. This benefits efficiency of DING12-P30 significantly. DING12-P14 and LatticeCrypto also use NTT. In BCNS15, they choose Nussbaumer FFT.

(4) Overall performance. We remark that high efficiency of our P30 implementation come from efficient NFLlib library, which optimizes error sampling and polynomial multiplication. For P14 implementation, LatticeCrypto implements optimized-NTT for efficient polynomial multiplication. However, implementation of error sampling in LatticeCrypto library is not as efficient as NFLlib, therefore overall performance of P14 parameter choice is less efficient then P30. We also note that since DING12 error reconciliation is 1.57x faster than BCNS15 and this part is implemented using plain C language, this highlights error reconciliation mechanism of DING12.

We present comprehensive comparison chart of DING12-P30, BCNS15 and DING12-P14 in Table 3:

Table 3 Comparison result of DING12-P30, BCNS15 and DING12-P14.

	DING12-P30	DING12-P14	BCNS15
Party i timing (ms)	0.0706	0.139	0.708
Party j timing (ms)	0.0707	0.142	0.847
Party i to j message size (KB)	3.75	1.75	4
Party j to i message size (KB)	3.875	1.875	4.125
Signal size (KB)	0.125	0.125	0.125
Error tolerance	$q/4$	$q/4$	$q/4$
n	1024	1024	1024
Error distribution	Discrete Gaussian $\sigma = 3.192$	Centred Binomial $k = 16$	Discrete Gaussian $\sigma = 3.192$
Modulus size (bit)	30	14	32
Sampling operations	Party i: 2. Party j: 3		Party i: 2. Party j: 3
FFT/NTT	Party i: 2. Party j: 2		Party i: 2. Party j: 2
Inverse FFT/NTT	Party i: 1. Party j: 1		Party i: 1. Party j: 1
Signal computation operations (Each coefficient)	1 random hint function selection ($\sigma_0()$ or $\sigma_1()$) 1 if condition (decide value in inner or outer region)		1 subtraction 2 multiplication 1 division 1 rounding 1 mod operation
Error reconciliation computation operations (Each coefficient)	1 addition 2 mod operation		1 random region selection ($I_0 + E$ or $I_1 + E$) 1 if condition (decide value in region $I_0 + E$ or $I_1 + E$) 2 mod operation
Security level	128-bit classic >80-bit quantum	>256-bit classic >200-bit quantum	128-bit classic <80-bit quantum

Table 3 shows that DING12-P30 has smaller communication cost compared with BCNS15 since they choose 32-bit modulus ($q = 2^{32} - 1$). Size of signal value is exact same 1024 bits. DING12-P14 has more than half of message sizes compared with P30 and BCNS15 due to choosing 14-bit modulus.

We also would like to give a brief comment on NewHope RLWE key exchange [8]. NewHope improves BCNS15 with following major contributions: (1) an improved error reconciliation mechanism; (2) Randomly generated public parameter a ; (3) Security analysis; (4) Efficient implementation. Structure of protocol design and the idea of using signal and reconciliation remain the same. For reconciliation mechanism in NewHope, they adopt a more geometric idea, which divides 4-dimensional space into various parts and point coefficient vectors to different regions to generate signal and final key bits. They use four coefficients in key exchange material to extract one key bit. This is mainly for higher error tolerance. They use error correction codes on $\widetilde{D_4}$ lattice to achieve this. It is a geometric approach since signal vector in NewHope is computed as the difference between four coefficients of k_j and nearest center of Voronoi cell. For reconciliation, key bit is derived by adding signal vector to k_i/k_j and the sum points to nearest centre of Voronoi cell to decide 0 or 1 key bit is generated. Note that four coefficients generates 8-bit signal. Since they extract 1 bit from four coefficients, this allows higher tolerance than DING12 and BCNS15, which both of them extract 1 bit from one coefficient. NewHope directly generates 256-bit key, while DING12 and BCNS15 generate 1024-bit key.

6 Conclusion

With all detailed analysis, comparison, efficient implementations and benchmark we presented, we conclude that DING12 and BCNS15 are truly practical RLWE-based key exchange protocols. Error reconciliation is an important technique in RLWE-based constructions. Applications like

encryption, key exchange etc. require efficient, compact and high error tolerance reconciliation mechanism. How to reconcile errors using signal in DING12 and BCNS15 RLWE key exchange is explained. We analyse similarity and differences of these RLWE-based key exchange protocols very carefully, especially error reconciliation part. Our work is also the first to instantiate DING12 RLWE key exchange with 128-bit classic (80-bit quantum) and 256-bit classic (>200-bit quantum) secure parameter choices. Benchmarks show that our efficient implementations of P30 and P14 are truly efficient with 11x and 3.94x speed improvement over BCNS15, error reconciliation is 1.57x faster than BCNS15. RLWE-based cryptosystems are extremely efficient and have robust security, which can be applied in various applications (e.g., [16]-[25]) as post-quantum alternatives.

Acknowledgements

We would like to thank anonymous reviewers for valuable feedbacks. This work is supported by China Scholarship Council, National Natural Science Foundation of China (Grant No. 61672092) and Fundamental Research Funds for the Central Universities (Grant No. 2017YJS038). Jintai Ding is partially supported by NSF grant DMS-1565748 and US Air Force grant FA2386-17-1-4067.

References

[1] Diffie, W., Hellman, M.E., 1976. New directions in cryptography. Information Theory, IEEE Transactions on 22, 644–654.

[2] Shor, P.W., 1994. Algorithms for quantum computation: Discrete logarithms and factoring, in: Foundations of Computer Science, 1994 Proceedings., 35th Annual Symposium on. IEEE, pp. 124–134.

[3] Regev, O., 2009. On lattices, learning with errors, random linear codes, and cryptography. Journal of the ACM (JACM) 56, 34.

[4] Lyubashevsky, V., Peikert, C., Regev, O., 2013. On ideal lattices and learning with errors over rings. Journal of the ACM (JACM) 60, 43.

[5] Ding, J., Xie, X., Lin, X., 2012. A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. IACR Cryptology ePrint Archive 2012, 688.

[6] Peikert, C., 2014. Lattice cryptography for the internet, in: International Workshop on Post-Quantum Cryptography. Springer, pp. 197–219.

[7] Bos, J.W., Costello, C., Naehrig, M., Stebila, D., 2014. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem (No. 599).

[8] Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P., 2015. Post-quantum key exchange-a new hope. IACR Cryptology ePrint Archive 2015, 1092.

[9] Bos, J., Costello, C., Ducas, L., Mironov, I., Naehrig, M., Nikolaenko, V., Raghunathan, A., Stebila, D., 2016. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. ACM, pp. 1006–1018.

[10] Zhang, J., Zhang, Z., Ding, J., Snook, M., Dagdelen, Ö., 2015. Authenticated key exchange from ideal lattices, in: Advances in Cryptology-EUROCRYPT 2015. Springer, pp. 719–751.

[11] Ding, J., Alsayigh, S., Lancrenon, J., Saraswathy, R.V., Snook, M., 2017. Provably Secure Password Authenticated Key Exchange Based on RLWE for the Post-Quantum World, in: Cryptographers' Track at the RSA Conference. Springer, pp. 183–204.

[12] Albrecht, M.R., Player, R., Scott, S., 2015. On the concrete hardness of Learning with Errors. IACR Cryptology ePrint Archive 2015, 046.

[13] Aguilar-Melchor, C., Barrier, J., Guelton, S., Guinet, A., Killijian, M.-O., Lepoint, T., 2016. NFLlib: NTT-based fast lattice library, in: Cryptographers' Track at the RSA Conference. Springer, pp. 341–356.

[14] Longa, P., Naehrig, M., 2016. Speeding up the number theoretic transform for faster ideal lattice-based cryptography, in: International Conference on Cryptology and Network Security. Springer, pp. 124–139.

[15] Albano, P., Bruno, A., Carpentieri, B., Castiglione, Aniello, Castiglione, Arcangelo, Palmieri, F., Pizzolante, R., Yim, K., You, I., 2014. Secure and distributed video surveillance via portable devices. Journal of Ambient Intelligence and Humanized Computing 5, 205–213.

[16] Castiglione, Arcangelo, DAmbrosio, C., De Santis, A., Castiglione, Aniello, Palmieri, F., 2013. On secure data management in health-care environment, in: Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS), 2013 Seventh International Conference on. IEEE, pp. 666–671.

[17] Liu, Z., Gro\s sschädl, J., Hu, Z., Järvinen, K., Wang, H., Verbauwhede, I., 2017a. Elliptic curve cryptography with efficiently computable endomorphisms and its hardware implementations for the internet of things. IEEE Transactions on Computers 66, 773–785.

[18] Liu, Z., Huang, X., Hu, Z., Khan, M.K., Seo, H., Zhou, L., 2017b. On emerging family of elliptic curves to secure internet of things: Ecc comes of age. IEEE Transactions on Dependable and Secure Computing 14, 237–248.

[19] Cui, Z., Gao, G., Zhou, C., Yu, J., Deng, A., Deng, C., 2016. Efficient key management for publish/subscribe system in cloud scenarios. International Journal of High Performance Computing and Networking 9, 489–498.

[20] Shi, Y., Liu, J., Han, Z., Qiu, S., 2016. Deterministic attribute-based encryption. International Journal of High Performance Computing and Networking 9, 443–450.

[21] Zhang, J., Zhao, X., Zhen, W., 2017. OGPADSM2: oriented-group public auditing for data sharing with multi-user modification. International Journal of High Performance Computing and Networking 10, 240–249.

[22] González, S., Huguet, L., Martínez, C., Villafañe, H., 2013. Discrete logarithm like problems and linear recurring sequences. Advances in Mathematics of Communications 7, 187–195. https://doi.org/10.3934/amc.2013.7.187

[23] Jalili, R., Dousti, M.S., 2015. Forsakes: A forward-secure authenticated key exchange protocol based on symmetric key-evolving schemes. Advances in Mathematics of Communications 9, 471–514. https://doi.org/10.3934/amc.2015.9.471

[24] Micheli, G., 2015. Cryptanalysis of a noncommutative key exchange protocol. Advances in Mathematics of Communications 9, 247–253. https://doi.org/10.3934/amc.2015.9.247