NTRU-KE: A Lattice-based Public Key Exchange Protocol

Xinyu Lei[⋆] and Xiaofeng Liao

College of Computer Science, Chongqing University, Chongqing 400044, China.

Abstract. Public key exchange protocol is identified as an important application in the field of public-key cryptography. Most of the existing public key exchange schemes are Diffie-Hellman (DH)-type, whose security is based on DH problems over different groups. Note that there exists Shor's polynomial-time algorithm to solve these DH problems when a quantum computer is available, we are therefore motivated to seek for a non-DH-type and quantum resistant key exchange protocol. To this end, we turn our attention to lattice-based cryptography. The higher methodology behind our roadmap is that in analogy to the link between ElGamal, DSA, and DH, one should expect a NTRU lattice-based key exchange primitive in related to NTRU-ENCRYPT and NTRU-SIGN. However, this excepted key exchange protocol is not presented yet and still missing. In this paper, this missing key exchange protocol is found, hereafter referred to as NTRU-KE, which is studied in aspects of security and key-mismatch failure. In comparison with ECDH (Elliptic Curve-based Diffie-Hellman), NTRU-KE features faster computation speed, resistance to quantum attack, and more communication overhead. Accordingly, we come to the conclusion that NTRU-KE is currently comparable with ECDH. However, decisive advantage of NTRU-KE will occur when quantum computers become a reality.

Keywords: Key exchange, Diffie-Hellman, NTRU, quantum resistance

1 Introduction

1.1 Application Background

The first practical public key exchange protocol was introduced by Diffie and Hellman in their epoch-making paper [1], allowing two parties that have never met to establish a shared secret key by exchanging information over an untrusted and insecure channel. An adversary is unable to retrieve the key even with the ability to eavesdrop the communication channel. In practical applications, public key exchange protocols play an important role in the use of shared-key cryptography to protect transmitted data over insecure networks. For instance, they are a central building block of today's most commonly used cryptographic protocols (such as SSL [2], IPSec [3], and SSH [4]), which have enabled today's proliferation of secure electronic commerce over the Internet. Consequently, public key exchange, since its proposal, has stimulated considerable research efforts.

1.2 Related Work

Let us begin with the classic Diffie-Hellman (DH) primitive protocol. A prime p and a generator $g \in \mathbb{Z}_p^\times$ are public generated, they are known not only to all parties but also the adversary. The detailed steps of DH are as follows.

[⋆] Email: xy-lei@qq.com

– **Step 1:** entity i picks $x \in [1, p-2]$ at random and let $X = g^x (\text{mod } p)$. Then, he sends X to entity j.
– **Step 2:** entity j picks $y \in [1, p-2]$ at random and let $Y = g^y (\text{mod } p)$. Then, he sends Y to entity i.

It holds that $X^y{=}(g^x)^y{=}(g^y)^x{=}Y^x$ over group $\mathbb{Z}_p^\times$. Entity i computes $Y^x(\text{mod } p)$, and entity j computes $X^y(\bmod\ p)$, then both entities come up with the common secret key $K{=}g^{xy}(\bmod\ p)$.

After the invention of the DH protocol, most of the subsequent public key exchange protocols appear to be *DH-type protocols*. By DH-type we mean that these protocols are essentially the DH protocol, but works over different groups. These DH-type protocols are summarized below.

– **Elliptic curve-based DH.** Elliptic curve-based DH (ECDH) is a variant of the DH protocol using group that consists of points on an elliptic curve (see [5] for details). Compared with DH, the primary benefit possessed by ECDH is a smaller key size, thereby reducing storage and transmission requirements.
– **Recurrence equations-based DH.** This class of DH-type public key exchange protocols are based on recurrence equations, see e.g., GH [6], LUC [7], and Chebyshev-based [8, 9]. These cryptosystems are all based on recurrence equations, but with different forms. We explicitly note that there always exists a DH-type key exchange protocol in each cryptosystem. In these cryptosystems, the basic group operation is iteration of a certain recurrence equation, and the group members are generated by iterations of the recurrence equation from certain initial values.
– **XTRDH.** In XTR [10] cryptosystem, it follows that the corresponding key exchange protocol XTRDH is a DH-type protocol. In essence, XTRDH is also a recurrence equation-based DH. We list it separately because the significant feature of XTRDH is that it uses the trace over GF(p^2) to represent elements of a subgroup of GF$(p^6)^*$. This leads to a smaller key size and substantial savings both in communication and computation overhead in applications.

We are now concerned with the hard problems, on which the security of these DH-type protocols depends. Let g denote a generator of a group (e.g., the multiplicative group $\mathbb{Z}_p^\times$, an elliptic curve group, or a group generated by iterations of a certain recurrence equation). Let x and y be randomly chosen elements in this group.

Definition 1. *i) The* ***computational DH (CDH) problem*** *is informally stated as follows: given an generator g and the values of g^x and g^y in a group, compute the value of g^{xy}.*

ii) The ***CDH assumption****: for appropriate parameters, it is computational intractable to solve CDH problem.*

iii) The ***discrete logarithm (DL) problem*** *is informally stated as follows: given an generator g and the value of g^x in a group, compute the value of x.*

iv) The ***DL assumption****: for appropriate parameters, it is computational intractable to solve DL problem.*

The security of these DH-type protocols relies on the CDH assumption (in stronger semantic security notion, decisional DH (DDH) assumption is required). Notice that we can break CDH assumption if we already know an algorithm to break DL assumption. We denote this relationship by CDH assumption>DL assumption. To date, the most efficient means known to solve the DH problem is to solve the DL problem, and there is a strong heuristic argument showing that DL problem and DH problem are very likely to be equivalent [11]. Accordingly, we may also say that the security of these DH-type protocols relies on the DL assumption.

1.3 Research Motivations

Two motivations of our research are identified.

- **Motivation 1.** It is disappointed to observe that over three decades after the discovery of the DH protocol, the cryptographer's toolbox still contains very few public key exchange primitive schemes. Despite the fact that a tremendous amount of key exchange protocols have been proposed and analyzed, many of these protocols are dependent on DH problem, see, e.g., MTI [12], Unified Model [13], MQV [14], HMQV [15], to just list a few. This motivates us to seek for high-performance non-DH-type key exchange primitives.
- **Motivation 2.** Shor in [16,17] showed that quantum computers, when they become a reality, will render DH-type protocols insecure. It is shown that the quantum computer is developed rapidly [18], leading to that quantum resistant key exchange protocols are urgently needed.

Target at designing a non-DH-type and quantum resistant key exchange primitive, we should turn our attention to new hard problems. Ironically, over the past years, very few convincingly efficient key exchange protocols were well designed by using other hard problems despite of considerable research efforts. The seek for new schemes appears sometimes hopeless as new schemes are immediately broken or, if they survive, are compared with the DH-type protocols, which are obviously elegant, simple, and efficient. Consequently, the search for new key exchange protocols still remains a major challenge.

1.4 Roadmap

In the complexity-theoretic-based modern cryptography, the security of a public-key protocol should depend on a hard problem. To design a non-DH-type scheme, we start by review the existing cryptosystems and the underlying hard problems. Three of the most successful practical cryptosystems are taken into account.

1. DL cryptosystem that is based on DL problem in the multiplicative group $\mathbb{Z}_p^\times$;
2. ECDL cryptosystem that is based on ECDL problem in an elliptic curve group; and
3. NTRU cryptosystem that is related to *shortest vector problem* (SVP) and *closest vector problem* (CVP) in a NTRU lattice.

Generally speaking, a public cryptosystem primarily consists of three important members: 1) public key encryption, 2) digital signature, and 3) public key exchange. The representative members in the above three cryptosystems are displayed in Table. 1. It is clearly observed from Table. 1 that the public key exchange protocol in the NTRU cryptosystem is missing. *In analogy to the link between ElGamal, DSA, and DH, one should expect a NTRU lattice-based key exchange primitive in related to NTRU-ENCRYPT and NTRU-SIGN.* Following the naming rule, this key exchange protocol, if exists, should be named as NTRU-KE. Our roadmap is now clear, namely exploring this potentially existed NTRU-KE.

Table 1. Representative Members in DL, ECDL, and NTRU Cryptosystems

hard problem / protocol	DL Problem	ECDL Problem	SVP and CVP in a NTRU lattice
public key encryption	ElGamal [19]	ECElGamal	NTRU-ENCRYPT [20]
digital signature	DSA [21]	ECDSA [22]	NTRU-SIGN [23, 24]
public key exchange	DH	ECDH	**?**

Note: the prefix "EC" denotes the corresponding elliptic curves-based hard problem or protocols. The symbol "**?**" denotes the item is missing currently.

1.5 Main Contributions

We regard our main contributions as three-fold:

1. contributing towards a successful construction of a non-DH-type and quantum resistant key exchange primitive scheme NTRU-KE, whose security is related to hard problems in a NTRU lattice;
2. estimating the key-mismatch failure of NTRU-KE theoretically; and
3. performing a detailed comparative study between NTRU-KE and ECDH.

1.6 Paper Organization

The remainder of this paper is organized as follows. Section 2 introduces technical and mathematical preliminaries. In Section 3, we describe NTRU-KE, formalize the underlying hard problem, and analyze the lattice attacks. In Section 4, the probability of key-mismatch failure of NTRU-KE is estimated. Then, the performance of NTRU-KE is theoretically compared with ECDH in Section 5. Finally, some conclusions are drawn in Section 6.

2 Notions, Mathematical and NTRU-ENCRYPT Background

A generic execution of a key exchange protocol between two entities is called a *run* of the protocol. Any particular run of a protocol is called a *session*. The keying information exchanged in the course of a protocol run is referred to as a *session key*. The individual messages that form a protocol run are called *flows*.

We denote by $\mathbb{Z}$ the integer ring and by $\mathbb{Z}_q$ the residue class ring $\mathbb{Z}/\mathbb{Z}_q$. The truncated polynomial ring $R_q = \mathbb{Z}_q[x]/(X^N - 1)$ consists of all polynomials with degree less than N and coefficients in $\mathbb{Z}_q$. An element $f \in R_q$ can be written as a polynomial or a vector,

$$f = \sum_{i=0}^{N-1} f_i x^i = [f_0, \cdots, f_{N-1}].$$

Elements in R_q are written in bold type, if the vector description is needed. Two polynomials $f, g \in R_q$ are multiplied by the ordinary convolution,

$$(f * g)_k \equiv \sum_{i+j \equiv k (\bmod N)} (f_i \cdot g_j),\ k = 0, \cdots, N-1,$$

which is commutative and associative. The convolution product is presented by $*$ to distinguish it from the multiplication in $\mathbb{Z}_q$. We define a center l_2-norm of an element $f \in R_q$ by

$$|f|_2 = (\sum_{i=0}^{N-1} (f_i - \overline{f}))^{1/2},$$

where $\overline{f} = \frac{1}{N} \sum_{i=0}^{N-1} f_i$. Let $a \overset{\mathrm{r}}{\leftarrow} A$ denote the process of picking an element a from the set A uniformly at random. A real-valued function $\epsilon(k) < k^{-c}$ is negligible if for every $c > 0$ there exists $k_c > 0$ such that $\epsilon(k) < k^{-c}$ for all $k > k_c$.

The NTRU-ENCRYPT depends on three global integers (N, p, q) and four set $\mathcal{L}_f$, $\mathcal{L}_g$, L_r, L_m of polynomials of degree $N-1$ with small integer coefficients. The previous parameter choices for NTRU-ENCRYPT often take the small integer coefficients of polynomials to be binary $\{0, 1\}$ or trinary $\{-1,0,1\}$. We adopt the trinary polynomials because they are better to resist the recent hybrid meet-in-the-middle (MITM) and lattice reduction attack [25]. We set the notation:

$$\mathcal{L}(d_1,d_2)=\left\{\begin{matrix} F\in R_q : F \text{ has } d_1 \text{ coefficients equal to } 1, \\ d_2 \text{ coefficients equal to } -1, \text{ the rest } 0. \end{matrix}\right\}$$

With this notation, we proceed to set

$$\mathcal{L}_f = \mathcal{L}(d_f, d_f - 1),\ \mathcal{L}_g = \mathcal{L}(d_g, d_g),\ L_r = \mathcal{L}(d_r, d_r). \quad (1)$$

We take $\mathcal{L}_f = \mathcal{L}(d_f, d_f - 1)$ instead of $\mathcal{L}_f = \mathcal{L}(d_f, d_f)$ because it is required that f is invertible and a polynomial satisfying $f(1) = 0$ can never be invertible.

The NTRU-ENCRYPT algorithm starts by setting up three global integers (N,p,q), such that:

- N is prime,
- p and q are coprime, $\gcd(p,q) = 1$, and
- q is considerably larger than p.

Key Generation.

Step 1: Choose $f \overset{r}{\leftarrow} \mathcal{L}_f$ and $g \overset{r}{\leftarrow} \mathcal{L}_g$ such that there exist $F_q, F_p \in R_q$ satisfying

$$F_q * f \equiv 1 (\text{mod } q),\ F_p * f \equiv 1 (\text{mod } p).$$

Step 2: Let

$$h \equiv f_q^{-1} * g (\text{mod } q).$$

public key: h,   private key: f, F_p.

Encryption.

Step 1: To encrypt $m \in \mathcal{L}_m$, we first choose an $r \overset{r}{\leftarrow} \mathcal{L}_r$, then compute the ciphertext as

$$c \equiv pr * h + m (\text{mod } q).$$

Decryption.

Step 1: We first compute

$$\begin{aligned} a &= f * c && (\text{mod } q) \\ &= pr * g + f * m && (\text{mod } q). \end{aligned}$$

For appropriate parameter choices, the absolute value of coefficients in $(pr*g+f*m)$ is small, together with the fact that q is chosen to be large, we almost derive

$$a = pr * g + f * m.$$

Step 2: We then recover the message m by computing

$$m = F_p * a (\text{mod } p).$$

Note that, the latest parameters selection for NTRU-ENCRYPT always set $f = 1+pF$, where $F \in \mathcal{L}_f$ in the key generation process, then the Step 2 in the decryption process is eliminated since $F_p \equiv 1 (\text{mod } p)$.

The security of NTRU-ENCRYPT depends on the following computational assumption.

Definition 2. *([26]) i) The **NTRU-ENCRYPT inversion problem**: For a given security parameter k, which specifies (N,p,q) and the spaces $\mathcal{L}_f$, $\mathcal{L}_g$, $\mathcal{L}_m$, and $\mathcal{L}_r$, as well as a random public key h and ciphertext $c \equiv h * r + m (\text{mod } q)$, where $m \in \mathcal{L}_m$ and $r \in \mathcal{L}_r$, find m.*

*ii) The **NTRU-ENCRYPT assumption**: for appropriate parameters, it is computational intractable to solve the NTRU-ENCRYPT inversion problem.*

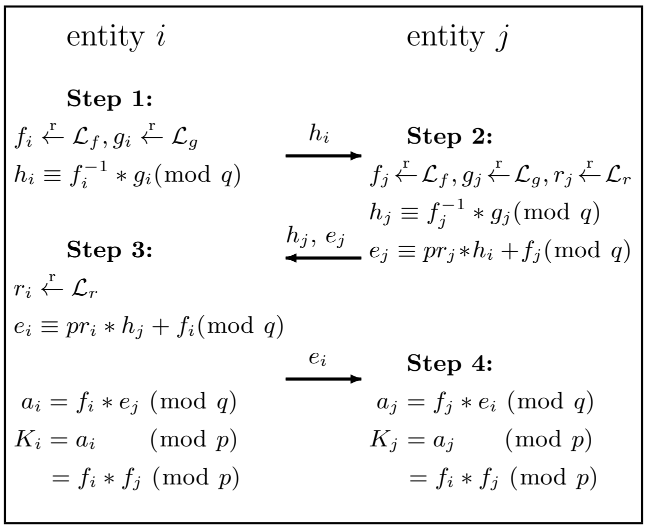
entity i
entity j
Step 1:
$f_i \overset{r}{\leftarrow} \mathcal{L}_f, g_i \overset{r}{\leftarrow} \mathcal{L}_g$
$h_i \equiv f_i^{-1} * g_i (\bmod\ q)$
h_i
Step 2:
$f_j \overset{r}{\leftarrow} \mathcal{L}_f, g_j \overset{r}{\leftarrow} \mathcal{L}_g, r_j \overset{r}{\leftarrow} \mathcal{L}_r$
$h_j \equiv f_j^{-1} * g_j (\bmod\ q)$
$h_j,\ e_j$
$e_j \equiv pr_j * h_i + f_j (\bmod\ q)$
Step 3:
$r_i \overset{r}{\leftarrow} \mathcal{L}_r$
$e_i \equiv pr_i * h_j + f_i (\bmod\ q)$
e_i
Step 4:
$a_i = f_i * e_j\ (\bmod\ q)$
$K_i = a_i\ (\bmod\ p)$
$= f_i * f_j\ (\bmod\ p)$
$a_j = f_j * e_i\ (\bmod\ q)$
$K_j = a_j\ (\bmod\ p)$
$= f_i * f_j\ (\bmod\ p)$

Fig. 1. Data Flows in NTRU-KE

3 NTRU-KE Construction

The classic DH protocol and the subsequent DH-type protocols have given cryptographers a strong impression that a key exchange primitive protocol typically consists of two flows. However, it seems impossible to design a two-flow-based key exchange primitive in related to NTRU-ENCRYPT, this is because the special structure of the public key in NTRU-ENCRYPT. This, from our point of view, is the main reason why the NTRU-KE protocol has been missing up till now. Notably, there is no restriction that a key exchange primitive must consist of two flows, and indeed, most of applicable key exchange protocols consist of more than two flows. In view of this consideration, the three-flow-based NTRU-KE primitive protocol is proposed.

The NTRU-KE protocol starts by setting up three global integers (N, p, q) at first, this process is the same with that of NTRU-ENCRYPT. The data flows of NTRU-KE are depicted in Fig. 1. Here is a detailed verbal description of these steps.

– **Step 1**: entity i picks $f_i \overset{r}{\leftarrow} \mathcal{L}_f$ such that there exists the inverse of f_i in R_q, and picks $g_i \overset{r}{\leftarrow} \mathcal{L}_g$. Then, he computes $h_i \equiv f_i^{-1} * g_i (\bmod\ q)$ and sends h_i to j.
– **Step 2**: upon receipt of h_i, entity j picks $f_j \overset{r}{\leftarrow} \mathcal{L}_f$ such that there exists the inverse of f_j in R_q, and picks $g_j \overset{r}{\leftarrow} \mathcal{L}_g$, $r_j \overset{r}{\leftarrow} \mathcal{L}_r$. Then, he computes $h_j \equiv f_j^{-1} * g_j (\bmod\ q)$ and $e_j \equiv pr_j * h_i + f_j (\bmod\ q)$. Afterwards, he sends h_j and e_j to i.
– **Step 3**: upon receipt of h_j and e_j, entity i picks $r_i \overset{r}{\leftarrow} \mathcal{L}_r$ and computes $e_i \equiv pr_i * h_j + f_i (\bmod\ q)$. Then, he sends e_i to j and computes $a_i = f_i * e_j (\bmod\ q)$, $K_i = a_i (\bmod\ p) = f_i * f_j (\bmod\ p)$.
– **Step 4**: upon receipt of e_i, entity j computes $a_j = f_j * e_i (\bmod\ q)$, $K_j = a_j (\bmod\ p) = f_i * f_j (\bmod\ p)$.

It can be easily checked that the resultant common session key is

$$K_i = K_j = f_i * f_j (\bmod\ p). \tag{2}$$

The NTRU-KE protocol consists of three flows, which is one more flow than these DH-type protocols. Compared with NTRU-ENCRYPT, it does not need to compute the inverse of f modulo p in NTRU-KE. Like DH-type protocols, the NTRU-KE protocol features elegant and simple.

3.1 Security

When a new public key protocol is proposed, in very few instances is there a rigorous proof that breaking this public key protocol is equivalent to solving an existing well studied hard problem. What frequently occurs is a definition of a new hard problem, which may relate to a previously well-defined hard problem. In the same manner, the proposed NTRU-KE follows this routine. The security of NTRU-KE depends on the following NTRU-KE assumption.

Definition 3. *i) The **NTRU-KE problem**: For a given security parameter k, which specifies (N, p, q) and the spaces $\mathcal{L}_f$, $\mathcal{L}_g$, $\mathcal{L}_m$, and $\mathcal{L}_r$, as well as h_i, h_j, e_i, and e_j, where these variables are clearly defined in Fig. 1, find $f_i * f_j (\text{mod } p)$.*

*ii) The **NTRU-KE assumption**: for appropriate parameters, it is computational intractable to solve the NTRU-KE problem.*

Despite it is hard to prove this assumption, the relationship between the NTRU-ENCRYPT assumption and the NTRU-KE assumption can be investigated. Suppose that there exists an efficient algorithm $\mathcal{A}$ to solve the NTRU-ENCRYPT inversion problem. Then, $\mathcal{A}$ can be reduced to an efficient algorithm to find f_j from e_j and find f_i from e_i. Finally, $\mathcal{A}$ can derive the session key by computing $K = f_i * f_j (\text{ mod } p)$. This implies NTRU-KE assumption>NTRU-ENCRYPT assumption (in analogy to CDH assumption>DL assumption).

3.2 Lattice Attacks

The security of NTRU-ENCRYPT is related to CVP and SVP in a *convolution modular lattice* (CML) $\mathcal{L}_{\mathbf{h}}$ (also called as a *NTRU lattice*). $\mathcal{L}_{\mathbf{h}}$ is the $2N$ dimensional lattice with basis given by the rows of the matrix,

$$\mathcal{L}_{\mathbf{h}} = \left[\begin{array}{c|c} \mathbf{I} & \mathbf{H} \\ \hline \mathbf{0} & q\mathbf{I} \end{array}\right], \tag{3}$$

where $\mathbf{I}$ is an $N \times N$ identity matrix, $\mathbf{0}$ is an $N \times N$ zero matrix, and $\mathbf{H}$ is a circulant matrix generated from h,

$$\mathbf{H}_{i,j} = h_{j-i(\text{ mod } N)}.$$

Another way to describe CML $\mathcal{L}_{\mathbf{h}}$ (3) is the set of vectors

$$\mathcal{L}_{\mathbf{h}} = \{[\mathbf{f}, \mathbf{g}] \in \mathbb{Z}^{2N} \mid \mathbf{f} * \mathbf{h} \equiv \mathbf{g} (\text{mod } q)\}. \tag{4}$$

For appropriate parameters and a padding scheme, the best known attack on NTRU-ENCRYPT appears to be lattice attacks. Likewise, based on current knowledge, these lattice attacks are also the best known attack on NTRU-KE. Two lattice attacks on NTRU-KE are discussed below.

1) **Lattice attack on $\mathbf{h_i}$ (or $\mathbf{h_j}$)**. Suppose that a passive adversary $\mathcal{A}$ tries to recover secret information $[\mathbf{f}_i, \mathbf{g}_i]$ from $\mathbf{h}_i$. We here drop the subscript i, because it is the same case for entity j. By $\mathbf{h} \equiv \mathbf{f}^{-1} * \mathbf{g} (\text{mod } q)$, there exists $\mathbf{u} \in R_q$ with

$$\mathbf{u} = \frac{-\mathbf{f} * \mathbf{h} + \mathbf{g}}{q}, \tag{5}$$

such that

$$[\mathbf{f}, \mathbf{u}] \cdot \mathcal{L}_{\mathbf{h}} = [\mathbf{f}, \mathbf{g}].$$

Therefore, $\mathcal{L}_{\mathbf{h}}$ contains the vector $[\mathbf{f}, \mathbf{g}]$. By choosing appropriate parameters, then the vector $[\mathbf{f}, \mathbf{g}]$ is quite short, so it can be found by solving SVP (or approximate SVP) in $\mathcal{L}_{\mathbf{h}}$. Indeed, this lattice attack corresponds to the lattice attack on a NTRU-ENCRYPT private key.

2) **Lattice attack on $\mathbf{e_i}$ (or $\mathbf{e_j}$)**. In the light of

$$e_i \equiv pr_i * h_j + f_i (\text{mod } q), \tag{6}$$

we can rewrite this relation in the vector form as

$$[\mathbf{0}, \mathbf{e}_i] \equiv [\mathbf{r}_i, \mathbf{r}_i * (p\mathbf{h}_j)(\text{mod } q)] + [-\mathbf{r}_i, \mathbf{f}_i]. \tag{7}$$

Recall the description of CML in (4), we can deduce from (7) that $[\mathbf{r}_i, \mathbf{r}_i * (p\mathbf{h}_j)(\text{mod } q)]$ is in the lattice $\mathcal{L}_{p\mathbf{h}_j}$. The point $[\mathbf{0}, \mathbf{e}_i]$ is only $[-\mathbf{r}_i, \mathbf{f}_i]$ away from a lattice point of $\mathcal{L}_{p\mathbf{h}_j}$. By choosing appropriate parameters, then the vector $[-\mathbf{r}_i, \mathbf{f}_i]$ is quite short, so $[-\mathbf{r}_i, \mathbf{f}_i]$ can be recovered if we can find a vector in $\mathcal{L}_{p\mathbf{h}_j}$ that is closest to the vector $[\mathbf{0}, \mathbf{e}_i]$. Indeed, this lattice attack corresponds to the lattice attack on a NTRU-ENCRYPT message.

It can be easily found that either of the above two lattice attacks, if succeed, will lead to a recovery of the session key. They indicate how NTRU-KE relate to SVP and CVP. The ideal lattice basis reduction algorithms can be used to solve SVP and CVP in CML, and therefore, they can be applied to carry out the above two lattice attacks. However, the lattice attacks do not necessarily imply that NTRU-KE is insecure, as currently known lattice reduction algorithms (such as LLL [27] algorithm or its improved variants, e.g., BKZ-LLL [28]) still remain exponential time algorithms in terms of security parameters.

4 Key-mismatch Failure Analysis

It is well-known that NTRU-ENCRYPT algorithm has a probability of description failure. A similar failure is also found in NTRU-KE, hereafter referred to as *key-mismatch failure*. We now proceed to estimate the probability of key-mismatch failure in NTRU-KE. We first set

$$\begin{cases} P_i = \text{Prob}[(f_i * e_j (\text{mod } q)) \neq f_i * e_j], \\ P_j = \text{Prob}[(f_j * e_i (\text{mod } q)) \neq f_j * e_i]. \end{cases}$$

Because of symmetry, $P_i = P_j$ holds immediately. Some parameter choices for NTRU-KE may cause occasional key-mismatch failure. The probability of key-mismatch failure P_{KM} is formally defined as

$$P_{\text{KM}} = \text{Prob}[K_i \neq K_j].$$

For appropriate parameters, P_i and P_j can be very small quantities, this leads to

$$P_{\text{KM}} \approx 2P_i = 2P_j. \tag{8}$$

The probability P_i can be conservatively bounded by the probability that one or more coefficients of

$$a_i = f_i * e_j = pr_j * g_i + f_i * f_j$$

have an absolute value greater than $q/2$. We set

$$P(c) = \text{Prob}\begin{bmatrix}\text{a given coefficient of } pr_j * g_i + \\ f_i * f_j \text{ has absolute value} \geq c.\end{bmatrix} \tag{9}$$

Let X_j denote a coefficient of $pr_j * g_i + f_i * f_j$, then X_j is a sum of N terms,

$$X_j = \sum_{i=1}^{N} (py_i + z_i),$$

with each $y_i = r_k g_l$ and $z_i = f_s f_t$ for some k, l, s, t. Because of (1) and (6), the mathematical expectation of X_j can be obtained. For sufficiently large N, it holds that $E(X_j) = 0$. Then, the variance of X_j is

$$\begin{aligned} \sigma^2 = D(X_j) &= E(X_j^2) - E^2(X_j) \\ &= E(X_j^2) = \sum_{i=1}^{N} (p^2 E(y_i^2) + E(z_i^2)) \\ &= \frac{4}{N}(p^2 d_r d_g + d_f^2). \end{aligned} \tag{10}$$

Combining (6) and (10) yields

$$\sigma = \sqrt{\frac{4}{N}(p^2 d_r^2 + d_f^2)}. \tag{11}$$

Suppose that N is sufficiently large and X_j are independent variables. Note that X_j are identically distributed, Central Limit Theorem is applied to X_j, one has

$$\text{Prob}[X_j \geq c] = \text{Prob}[X_j \leq -c] < \frac{1}{\sqrt{2\pi}} \int_{c/\sigma}^{\infty} e^{-x^2/2} dx. \tag{12}$$

We can rewrite (12) as a single inequality,

$$\text{Prob}[|X_j| \geq c] < \frac{2}{\sqrt{2\pi}} \int_{c/\sigma}^{\infty} e^{-x^2/2} dx. \tag{13}$$

Translating this into the notation of the complementary error function, we arrive at

$$P(c) = \text{Prob}[|X_j| \geq c] < \text{erfc}(c/(\sqrt{2}\sigma)). \tag{14}$$

Recall that one or more X_j have an absolute value greater than $c = q/2$ will lead to $f_i * e_j (\text{mod } q) \neq f_i * e_j$, it holds that

$$P_i = \sum_{i=1}^{N} \binom{N}{i} P^i(c)(1 - P(c))^{N-i}, \ c = q/2. \tag{15}$$

We can always ensure that $P(c)$ is a very small quantity via choosing appropriate parameters, therefore ignoring the high order small quantities in (15) yields

$$P_i \approx N \cdot P(c). \tag{16}$$

Combining (8), (14), (16), and ignoring approximation errors leads to

$$P_{\text{KM}} < U, \tag{17}$$

where

$$U = 2N \cdot \text{erfc}(c/(\sqrt{2}\sigma)), \tag{18}$$

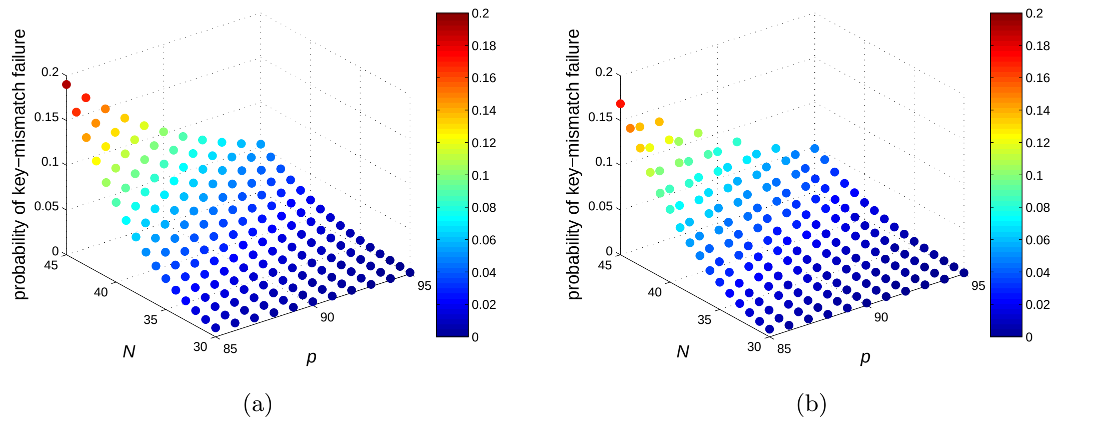

Fig. 2. (a) Data points are given by (18). (b) Data points are obtained by average over 10^5 simulations.

with $c = q/2$, and σ is given by (11). It is indicated from (17) that the probability of key-mismatch failure in NTRU-KE is approximately bounded above by U.

Since several assumptions and approximations are made in deriving (17), a simulation is carried out to validate the correctness of (17). In this simulation, we let

$$p = 3,\ d_r = \lfloor N/3 \rfloor,\ d_f = \lfloor N/4 \rfloor,$$

with p and N being x- and y-coordinates in Fig. 2. Simulation results in Fig. 2 show that the bound in (18) is a little conservative estimation of reality. Accordingly, (18) can be applied to have an estimation of key-mismatch failure in NTRU-KE.

5 Comparison

It is currently impossible to have an experimental comparison of performance for NTRU-KE and ECDH (the state-of-the-art benchmark). This is because how to choose secure parameters of NTRU-KE remains unsolved and this is a nontrivial task. According to the performance report of NTRU-ENCRYPT [20, 29], NTRU-KE shares an identical performance with that of NTRU-ENCRYPT since they have an identical underlying basic operation, i.e., convolution multiplication in R_q. Several merits and drawbacks are summarized as follows.

Merits:

1. *Computation speed.* NTRU-KE is faster than ECDH.
2. *Space cost of domain parameters.* NTRU-KE is smaller than ECDH.
3. *Domain parameters selection.* NTRU-KE is easier than ECDH. This is because the selection of secure elliptic curves is always cumbersome [30].
4. *Hardware implementation.* The convolution multiplication in the NTRU cryptosystem offers better performance in hardware implementation than the basic operation in ECDH [31].
5. *Resistance to parallelized and distributed attack.* NTRU-KE is better resistance to parallelized and distributed attack. This is because lattice reduction algorithms (e.g., LLL) are highly sequential.
6. *Resistance to quantum attack.* NTRU-KE is considered to be quantum resistant, whereas ECDH is not. This is because lattice-based cryptography is currently quantum resistant [32].

Drawbacks:

1. *Communication Overhead.* NTRU-KE is heavier than ECDH. Moreover, NTRU-KE needs one more communication flow.
2. *Space cost for private key.* NTRU-KE is larger than ECDH.
3. *Key-mismatch failure.* NTRU-KE has a probability of key-mismatch failure. This probability can be restricted to be negligible via choosing appropriate parameters.

According to the above summary, we come to the conclusion that NTRU-KE is comparable with ECDH based on current knowledge. The decisive advantage of NTRU-KE, however, will occur when a quantum computer is available.

6 Conclusions

In this paper, we have helped the NTRU cryptosystem to find its long missing member NTRU-KE, which has been studied in aspects of security and key-mismatch failure. It worth noting that the proposed NTRU-KE is only an anonymous key exchange primitive, and therefore, it does not provide authentication of the entities, making it vulnerable to man-in-the-middle attacks. A wide variety of cryptographic authentication schemes and protocols have been developed to provide authenticated key agreement to prevent man-in-the-middle and related attacks. These methods generally mathematically bind the agreed key to other agreed-upon data, such as public/private key pairs, shared secret keys, and passwords. We will not elaborate how to enable NTRU-KE to provide authentication. One meaningful future work is to investigate the parameter choices of NTRU-KE.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61170249).

References

1. W. Diffie, M. Hellman, New directions in cryptography, Information Theory, IEEE Transactions on 22 (6) (1976) 644–654.
2. A. Freier, P. Karlton, P. Kocher, The secure sockets layer (ssl) protocol version 3.0.
3. S. Kent, R. Atkinson, Security architecture for the internet protocol (1998).
4. T. Ylonen, C. Lonvick, The secure shell (ssh) protocol architecture.
5. D. Hankerson, A. Menezes, S. Vanstone, Guide to Elliptic Curve Cryptography, Springer-Verlag New York, Inc, 2004.
6. G. Gong, L. Ham, Public-Key Cryptosystems Based on Cubi Finite Field Extensions, IEEE Trans. Inform. Theory 45 (7) (1999) 2601–2605.
7. P. Smith, M. Lennon, LUC: A New Public Key System, in: Proc. Ninth IFIP Int. Symp. Computer Security, 1999, pp. 103–117.
8. L. Kocarev, Z. Tasev, Public-Key Encryption Based on Chebyshev Maps, in: Proc. 2003 IEEE Int. Symp. Circuit and Systems, Vol. 3, 2003, pp. 28–31.
9. X. Liao, F. Chen, K. W. Wong, On the security of public-key algorithm based on chebyshev polynomials over the finite field $\mathbb{Z}_n$, IEEE Trans. Computers 59 (10) (2010) 1392–1401.
10. A. Lenstra, E. Verheul, The XTR Public Key System, in: Proceedings of Crypto 2000, LNCS 1880, Springer-Verlag, 2000, pp. 1–19.
11. U. Maurer, S. Wolf, The relationship between breaking the Diffie-Hellman protocol and computing discrete logarithm, SIAM Journal of Computing 28 (5) (1999) 1689–1721.

12. T. Matsumoto, Y. Takashima, On seeking smart public-key-distribution systems, IEICE TRANSACTIONS (1976-1990) 69 (2) (1986) 99–106.
13. R. Ankney, D. Johnson, M. Matyas, The unified model, Contribution to X9F1.
14. L. Law, A. Menezes, M. Qu, J. Solinas, S. Vanstone, An efficient protocol for authenticated key agreement, Designs, Codes and Cryptography 28 (2) (2003) 119–134.
15. H. Krawczyk, Hmqv: A high-performance secure diffie-hellman protocol, in: Advances in Cryptology–CRYPTO 2005, Springer, 2005, pp. 546–566.
16. P. Shor, Algorithms for Quantum Computation: Discrete Logarithms and Factoring, in: Proc. 35th Annual IEEE Symposium on Foundations of Computer Science, IEEE Press, Piscataway, 1994, pp. 124–134.
17. P. Shor, Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer, SIAM J. Comput. 26 (5) (2006) 1484–1509.
18. A. Daskin, A. Grama, G. Kollias, S. Kais, Universal programmable quantum circuit schemes to emulate an operator, The Journal of chemical physics 137 (2012) 234112.
19. T. ElGamal, A public key cryptosystem and a signature scheme based on discrete logarithms, in: Advances in Cryptology, Springer, 1985, pp. 10–18.
20. J. Hoffstein, J. Pipher, J. H. Silverman, Ntru: A ring-based public key cryptosystem, in: Algorithmic number theory, Springer, 1998, pp. 267–288.
21. NIST, Announcing the Standard for DIGITAL SIGNATURE STANDARD (DSS)Federal Information Processing Standards Publication 186.
22. D. Johnson, A. Menezes, S. Vanstone, The elliptic curve digital signature algorithm (ecdsa), International Journal of Information Security 1 (1) (2001) 36–63.
23. J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. Silverman, W. Whyte, NTRUSign: Digital signatures using the NTRU lattice, in: J. Buhler (Ed.), Proceedings of CT-RSA2003, Lecture Notes in Computer Science, Vol. 2612, Berlin, Germany: Springer-Verlag, 2003, pp. 112–140.
24. Y. Hu, B. Wang, W. He, NTRUSign With a New Perturbation, IEEE Trans. Inform. Theory 54 (7) (2008) 3216–3221.
25. N. Howgrave-Graham, A Hybird Meet-in-the-Middle and Lattice Reduction Attack on NTRU, in: CRYPTO, 2007, pp. 150–169.
26. P. Nguyen, D. Pointcheval, Analysis and Improvements of NTRU Encryption Paddings, in: Crypto'02, LNCS, Vol. 2442, Springer-Verlag, Berlin, 2002, pp. 210–225.
27. A. Lenstra, H. W. Lenstra, L. Lovasz, Factoring polynomials with rational coefficients 261 (1982) 515–534.
28. C. P. Schnorr, M. Euchner, Lattice basis reduction: improved practical algorithms and solving subset problems, Math. Programming 66, Ser. A (1994) 181–199.
29. J. Hermans, F. Vercauteren, B. Preneel, Speed records for ntru, in: Topics in Cryptology-CT-RSA 2010, Springer, 2010, pp. 73–88.
30. M. Lochter, J. Merkle, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation.
31. C. O'Rourke, B. Sunar, Achieving NTRU with Montgomery Multiplication, IEEE. Trans. Comput. 52 (4) (2003) 440–448.
32. R. Perlner, D. Cooper, Quantum resistant public key cryptography: a survey, in: Proceedings of the 8th Symposium on Identity and Trust on the Internet, ACM New York, NY, USA, 2009, pp. 85–93.