Key Reuse Attack on NewHope Key Exchange Protocol

Chao Liu[1], Zhongxiang Zheng[2], and Guangnan Zou[3]

[1] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, P.R. China
`liu_chao@mail.sdu.edu.cn`

[2] Department of Computer Science and Technology Tsinghua University, P.R. China
`zhengzx13@mails.tsinghua.edu.cn`

[3] Space Star Technology co., LTD, P.R. China
`zouguangnan@spacestar.com.cn`

Abstract. In recent years, Ring Learning with Errors (RLWE) key exchange has been recognized for its efficiency and is considered a potential alternative to Diffie-Hellman (DH) key exchange protocol. We focus on RLWE key exchange protocols in the context of key reuse. In 2016 (ePrint 085), Fluhrer firstly presented an attack aiming at the case where party B reuse his secret key. In key reuse attack, the adversary plays the role of A with the abilities to initiate any number of key exchange sessions with party B. The adversary initiates a sequence of key exchange sessions with a malformed key, then looks for the *signal* variations sent by party B. In this work, we describe a new key reuse attack against the NewHope key exchange protocol proposed by Alkim et al. in 2016. We give a detailed analysis of the *signal* function of the NewHope and describe a new key recovering technique based on the special property of NewHope's *signal*.

Keywords: RLWE· key exchange · post quantum· key reuse· active attack.

1 Introduction

In 1994, Shor [16] devised that the discrete log problem can be solved in polynomial time by quantum computers. The fact that currently used Diffie-Hellman (DH) key exchange algorithms are mainly based on the hardness of the discrete log problem, leads to the search for cryptographic protocols with resistance to all known quantum algorithms. Lattice based constructions which appear to be resistant to attack by both classical and quantum computers, have been regarded as an important candidate for post-quantum cryptography. Recent progress in the development of Learning With Errors (LWE) and its variants puts lattice cryptography in an excellent position for use in practice. The Learning With Errors (LWE) problem was first introduced by Regev in [15] and then Lyubashevsky, Peikert and Regev [11] proposed an algebraic variant of LWE called Ring Learning With Errors (RLWE), which is of better efficiency.

Key exchange is a fundamental cryptographic primitive where cryptographic keys are allowed to exchanged between two or more participants. LWE and RLWE based key exchange protocols are considered to be a desirable replacement for DH protocols. Ding et al. firstly introduced such key exchange in [8] and several recent key exchange variants [2–5, 14, 18] that rely on the hardness of the LWE problem or RLWE problem have been proposed and implemented. In RLWE based key exchange, the two parties A and B firstly compute approximate values using the public key of the other's, then party B sends an additional information about the interval in which the approximately equal value lies, to party A. Finally, both two parties compute a shared secret using this additional information which we refer to as the *signal*. In this paper, we consider the case that party B reuses his secret key. Since key reuse is widely adopted for efficiency reasons (see [7], Section 4), attack analysis on such case will reveal some potential danger of the key exchange protocol.

In this work, we focus on NewHope key exchange proposed by Alkim et al. in [2]. NewHope implementation is very rapid, and is tested in Google Chrome Canary browser for its post quantum experiment [9]. And it was awarded the 2016 Internet Defense Prize [17].

Previous Work. The first attack on RLWE key exchange for reused public keys was described by Fluhrer in [10]. A detailed description of key reuse attack is presented by Ding et al. in [6] on key exchange proposed in [8]. The idea of their attacks is to deviate from the key exchange in generating the adversary's public key, then the adversary uses the variations of the *signal* to extract information about the party B's secret key.

Ding et al. [7] also described a new attack on Ding's one pass case key exchange [8]. This attack doesn't rely on the *signal* function output but use only the information of whether the final key of two parties agree.

Our Contribution. An error-reconciliation mechanism allows two parties with two "approximately agreeing" secret values to reach exact agreement. NewHope's error-reconciliation mechanism is more complex than Ding's key exchange's error-reconciliation mechanism presented in [8]. For details, its *signal* function is constructed based on a special lattice $\tilde{D}_4$ and the *signal* doesn't change regularly as Ding's key exchange does. Hence Fluhrer or Ding's attack can't be adopted directly. We analysis the *signal* function of NewHope and find that for every *signal*, there is one special vector corresponding to it. Let $\mathbf{a}_{k,l}[i]$ be one of dimension of this special vector, then we find that the sequence $(\mathbf{a}_{k,l}[i])_{k=0,1,\dots,q-1}$ have "periodic" property. We show that the number of period of $(\mathbf{a}_{k,l}[i])_{k=0,1,\dots,q-1}$ equals to the absolute value of one coefficient of the secret key. We introduce a technique to select sequence $(\mathbf{a}_{k,l}[i])_{k=0,1,\dots,q-1}$ and construct algorithm to compute the number of the period of this sequence, also we use a new way to eliminate the ambiguity of the $\pm$ sign of the coefficients. Experiments have been conducted to verify the correctness of our attack.

Organization. In Section 2. we discuss some notations used in this paper and background on RLWE. The NewHope key exchange and key reuse attack is reviewed in Section 3. And our attack is described in Section 4. The conclusion is presented in Section 5.

2 Preliminaries

2.1 Notation

Let n be an integer which is a power of 2. Define the ring of integer polynomials $R := \mathbb{Z}[x]/(x^n+1)$. For any positive integer q, define $R_q := \mathbb{Z}_q[x]/(x^n+1)$ as the ring of integer polynomials modulo x^n+1 where every coefficient is reduced modulo q. We define for $x \in \mathbb{R}$ the rounding function $\lfloor x \rceil = \lfloor x + \frac{1}{2} \rfloor \in \mathbb{Z}$. For an even (resp. odd) positive integer α, we define $r' = r \bmod^{\pm} \alpha$ to be the unique element r' in the range $-\frac{\alpha}{2} < r' \leq \frac{\alpha}{2}$ (resp. $-\frac{\alpha-1}{2} \leq r' \leq \frac{\alpha-1}{2}$) such that $r' = r \bmod \alpha$. Sometimes for $v = v_0 + v_1 x + v_2 x^2 + \ldots v_{n-1}x^{n-1} \in R$, we write $v[i]$ as v_i. Suppose χ is a probability distribution over R, $x \xleftarrow{\$} \chi$ means the sampling of $x \in R$ according to χ. For a probabilistic algorithm $\mathcal{A}$, we write $y \xleftarrow{\$} \mathcal{A}$ to represent that the output of $\mathcal{A}$ is assigned to y randomly.

We write (column) vectors in bold face as $\mathbf{v} = (v_0, v_1, \ldots, v_{n-1})^T$, where $\mathbf{v}^T$ denotes the transpose of the vector, and matrices in bold face as $\mathbf{A}$. For a vector $\mathbf{v} = (v_0, \ldots, v_{n-1})^T$ in $\mathbb{R}^n$, define the l_1 norm as $||\mathbf{v}||_1 = \sum_{i=0}^{n-1} |v_i|$ and the l_2 norm as $||\mathbf{v}||_2 = (\sum_{i=0}^{n-1} |v_i|^2)^{1/2}$. In this paper $||\cdot||$ denote the l_2 norm.

For any positive real $s \in \mathbb{R}$, we write $\rho_s(x) = exp(-\pi \frac{||x||^2}{s^2})$ as the Gaussian function which is scaled by a factor s. Let $\rho_s(\mathbb{Z}^n) = \sum_{\mathbf{x} \in \mathbb{Z}^n} \rho_s(\mathbf{x})$. Then for $\mathbf{x} \in \mathbb{Z}^n$, we let $D_{\mathbb{Z}^n,s}(\mathbf{x}) = \frac{\rho_s(\mathbf{x})}{\rho_s(\mathbb{Z}^n)}$ to indicate the n-dimensional discrete Gaussian distribution.

In applying the norms, we assume the coefficient embedding of elements from R to $\mathbb{R}^n$. For any element $y = \sum_{i=0}^{n-1} y_i x^i$ of R, we can embed this element into $\mathbb{R}^n$ as the vector $(y_0, \ldots, y_{n-1})$. Also we define the ring of integer polynomials $\bar{R} := \mathbb{Z}[x]/(x^4+1)$. For any $y(x) = y_0 + y_1 x + \cdots + y_{n-1}x^{n-1} \in R$, define a mapping:

$$\begin{aligned} \phi : R &\to \bar{R}^{\frac{n}{4}} \\ y(x) &\mapsto (\bar{y}_0(x), \bar{y}_1(x), \cdots, \bar{y}_{\frac{n}{4}-1}(x)) \end{aligned} \tag{1}$$

where $\bar{y}_i(x) = y_i + y_{i+\frac{n}{4}}x + y_{i+2\cdot\frac{n}{4}}x^2 + y_{i+3\cdot\frac{n}{4}}x^3 \in \bar{R}$. Assume the coefficient embedding of $\bar{y}_i$ to $\mathbf{v}_i = (y_i, y_{i+\frac{n}{4}}, y_{i+2\cdot\frac{n}{4}}, y_{i+3\cdot\frac{n}{4}})^T$, then we call vector $\mathbf{v}_i$ a *split vector* of y.

2.2 Ring Learning with Errors

The learning with Errors (LWE) problem is first introduced by Oded Regev [15] who shows that under a quantum reduction, solving LWE in the average cases is as hard as solving certain Lattice problems in the worst cases. But LWE based cryptosystems is not efficient for practical applications for its large key sizes of $O(n^2)$. In 2010, Lyubashevsky, Peikert, and Regev [11] introduced the Ring Learning with Errors(RLWE), which is the version of LWE in the ring setting and can drastically improve the efficiency. For a uniform random $a, s \overset{\$}{\leftarrow} R_q$ and error distribution χ, let $A_{s,\chi}$ denote the distribution of the RLWE pair $(a, as+e)$, where error $e \overset{\$}{\leftarrow} \chi$. Given $(a, as+e)$ for polynomial number of samples, the search version of RLWE is to find a secret s in R_q, and the decision version of the RLWE problem, denote R-DLWE$_{q,\chi}$ is to distinguish $A_{s,\chi}$ from the uniform distribution on $R_q \times R_q$. Like LWE, RLWE enjoys a worst case hardness guarantee, and we state in the following:

Theorem 1. ([11], **Theorem 3.6**) *Let $R = \mathbb{Z}[x]/(x^n + 1)$ where n is a power of 2. Let $\alpha = \alpha(n) < \sqrt{logn/n}$, and $q = 1 \mod 2n$ be a* ploy(n)*-bounded prime such that $\alpha q \geq \omega(\sqrt{logn})$. There exists a* ploy(n)*-time quantum reduction from $\tilde{O}(\sqrt{n}/\alpha)$-SIVP (Short Independent Vectors Problem) on ideal lattices in R to solving R-DLWE$_{q,\chi}$ with $l-1$ samples, where $\chi = D_{\mathbb{Z}^n,s}$ is discrete Gaussian distribution with $s = \alpha q \cdot (nl/log(nl))^{1/4}$.*

3 The Protocol and Key Reuse Attack

The Newhope key exchange. The NewHope key exchange protocol proposed by Alkim in [2] is an instantiation of Peikert's RLWE based passively secure key-exchange protocol [14]. Firstly, we recall NewHope protocol. Let n be a power of 2, $R = \mathbb{Z}[X]/(X^n + 1)$ and $R_q = R/qR$. In NewHope key exchange parameter $(n, q) = (1024, 12289)$. NewHope protocol is listed in table 1.

The RLWE secret and error is sampled from ψ_k which is a centered binomial. We note that one can sample from ψ_k by computing $\sum_{i=0}^{k} b_i - b'_i$ where the $b_i, b'_i \in \{0, 1\}$ are uniform independent bits, hence this way of sampling is very easy and efficient. NewHope uses the parameter $k = 16$.

HelpRec and Rec. In NewHope, party B firstly mapping $\phi(v_B) = (v_0, v_1, \ldots, v_{\frac{n}{4}-1})$, then coefficient embed v_i to $\mathbf{x}_i$. NewHope's error reconciliation is based on finding the closest vector in $\tilde{D}_4$ with basis

$$\mathbf{B} = \begin{pmatrix} 1 & 0 & 0 & 1/2 \\ 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & 1/2 \\ 0 & 0 & 0 & 1/2 \end{pmatrix}$$

and we define $\mathbf{g}^t := (1/2, 1/2, 1/2, 1/2)$. HelpRec$(\mathbf{x}_i; b)$ function to compute 2-bit reconciliation information as:

$$\text{HelpRec}(\mathbf{x}_i; b) = \text{CVP}_{\tilde{D}_4}\left(\frac{4}{q}(\mathbf{x}_i + b\mathbf{g})\right) \bmod 4, \tag{2}$$

Parameter: $q = 12289, n = 1024$
Error distribution: ψ_{16}

Party A		**Party** B
$seed \xleftarrow{\$} \{0,1\}^{\frac{n}{4}}$		
$a \leftarrow$ Parse(SHAKE-128($seed$))$\in R_q$		
$s_A, e_A \xleftarrow{\$} \psi_{16}^n$		$s_B, e_1, e_2 \xleftarrow{\$} \psi_{16}^n$
$p_A \leftarrow as_A + e_A \in R_q$	$\xrightarrow{(p_A, seed)}$	$a \leftarrow$ Parse(SHAKE-128($seed$))
		$p_B \leftarrow as_B + e_1 \in R_q$
		$v_B \leftarrow p_A s_B + e_2 \in R_q$
$v_A \leftarrow p_B s_A \in R_q$	$\xleftarrow{(p_B, \mathbf{r})}$	$\mathbf{r} \xleftarrow{\$}$FullHelpRec$(v_B) \in \mathbb{Z}_4^n$
$c \leftarrow$FullRec$(v_A, \mathbf{r})$		$c \leftarrow$FullRec$(v_B, \mathbf{r}) \in \{0,1\}^{\frac{n}{4}}$
$\mu \leftarrow$SHA3-256(c)		$\mu \leftarrow$SHA3-256(c)

Table 1. NewHope Scheme from [2].

where $b \in \{0, 1\}$ is a uniformly chosen random bit. Then compute $\mathbf{r} \xleftarrow{\$}$FullHelpRec$(v_B)$ where $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})^T$ by computing $\mathbf{r}_i \xleftarrow{\$}$ HelpRec$(\mathbf{x}_i)$ where $\mathbf{r}_i = (r_i, r_{i+\frac{n}{4}}, r_{i+2\cdot\frac{n}{4}}, r_{i+3\cdot\frac{n}{4}})^T \in \{0, 1, 2, 3\}^4$.

We call the output vector $\mathbf{r} \xleftarrow{\$}$FullHelpRec$(v_B)$ a *signal*. Vector $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})^T$ is split into vectors $\mathbf{r}_i = (r_i, r_{i+\frac{n}{4}}, r_{i+2\cdot\frac{n}{4}}, r_{i+3\cdot\frac{n}{4}})^T \in \{0, 1, 2, 3\}^4$, and Rec function

$$\text{Rec}(\mathbf{x}_i, \mathbf{r}_i) = \text{Decode}\left(\frac{1}{q}\mathbf{x}_i - \frac{1}{4}\mathbf{B}\mathbf{r}_i\right) \tag{3}$$

computes one key bit from a vector $\mathbf{x}_i$ and a reconciliation vector $\mathbf{r}_i$. To compute $c \leftarrow$FullRec$(v_B, \mathbf{r})$ where $c = (c_0, \ldots, c_{\frac{n}{4}-1})^T$, one need compute $c_i \leftarrow$ Rec$(\mathbf{x}_i, \mathbf{r}_i)$. CVP$_{\tilde{D}_4}$ and Decode are listed as Algorithm 1 and Algorithm 2, respectively.

Algorithm 1 CVP$_{\tilde{D}_4}$

```
Require: $\mathbf{v} := (v_0, v_1, v_2, v_3) \in \mathbb{R}^4$
Ensure: An vector $\mathbf{z} \in \mathbb{Z}^4$ such that $\mathbf{Bz}$ is a closest vector to $\mathbf{v}$
1: if $(||\mathbf{v} - \lfloor \mathbf{v} \rceil||_1) < 1$ then
2:     return $(\lfloor v_0 \rceil, \lfloor v_1 \rceil, \lfloor v_2 \rceil, 0)^T + \lfloor v_3 \rceil \cdot (-1, -1, -1, 2)^T$
3: else
4:     return $(\lfloor v_0 \rceil, \lfloor v_1 \rceil, \lfloor v_2 \rceil, 1)^T + \lfloor v_3 \rceil \cdot (-1, -1, -1, 2)^T$
5: end if
```

Key reuse attack. For a key exchange protocol, suppose that an active adversary $\mathscr{A}$ has the ability to initiate any number of key exchange sessions with party B who reuses his secret key s_B. An adversary plays the role of party A in the

Algorithm 2 Decode

```
Require: $\mathbf{v} \in \mathbb{R}^4/\mathbb{Z}^4$
Ensure: A bit $c$ such that $c\mathbf{g}$ is a closest vector to $\mathbf{v} + \mathbb{Z}^4$
1: $\mathbf{x} = \mathbf{v} - \lfloor \mathbf{v} \rceil$
2: return 0 if $\|\mathbf{x}\|_1 \leq 1$ and 1 otherwise
```

protocol and aims to recover s_B. The adversary $\mathscr{A}$ can set p_A adaptively by deviating from the protocol. Then in a key reuse attack on NewHope, once $\mathscr{A}$ has collected enough sequences of $(p_B^{(i)}, \mathbf{r}^{(i)})$ from party B, he can recover s_B.

4 Key Reuse Attack on NewHope

In this Section, we describe our key reuse attack on NewHope. We firstly give a general overview of the attack in Section 4.1. In Section 4.2, we introduce a special sequence which will be used in our attack techniques. In Section 4.3, we describe the sequence selecting and period counting techniques which are aimed to recover the secret.

4.1 General overview of our attack

We denote the deviated public key of the adversary as $p_{\mathscr{A}}$, and the secret and error terms of the adversary as $s_{\mathscr{A}}$ and $e_{\mathscr{A}}$ respectively. To explain our attack strategy, we firstly consider the simpler case when the error term e_2 is not added to the key computation of v_B of party B.

Choice of $s_{\mathscr{A}}$ and $e_{\mathscr{A}}$: The attacker chooses $s_{\mathscr{A}}$ to be 0 in R_q. The attacker chooses $e_{\mathscr{A}}$ to be the identity element 1 in R_q, and computes $p_{\mathscr{A}} = ke_{\mathscr{A}} = k$, where k takes values in $\mathbb{Z}_q$.

If we look at the key computation of B, we have $v_B = p_{\mathscr{A}} s_B = ks_B$. This results in the *signal* $\mathbf{r} \overset{\$}{\leftarrow} \text{FullHelpRec}(v_B)$ sent by the party B leaks the information of s_B, which is explained in the Section 4.2.

Oracle $\mathcal{S}$: There exists an oracle $\mathcal{S}$ which can be used to simulate party B's response. $\mathcal{S}$ performs the action of B and the adversary $\mathscr{A}$ has access to this oracle to make multiple queries. Assume that s_B is fixed for party B and $\mathcal{S}$ has access to the secret s_B. The input of $\mathcal{S}$ is $p_{\mathscr{A}}$ from $\mathscr{A}$. Oracle $\mathcal{S}$ computes $v_B = p_{\mathscr{A}} s_B$ and outputs $\mathbf{r} = \text{FullHelpRec}(v_B) \in \mathbb{Z}_4^n$.

We give the attack steps in the following.

Key reused attack by the adversary $\mathscr{A}$:

1. For every $k \in \{0, 1, \ldots, q-1\}$:
   a. Set $p_{\mathscr{A}} = k$.
   b. Invoke the oracle $\mathcal{S}$ with $p_{\mathscr{A}}$, and obtain $\mathcal{S}$'s output $\mathbf{r}_k = (r_{k,0}, \ldots, r_{k,n-1})$.

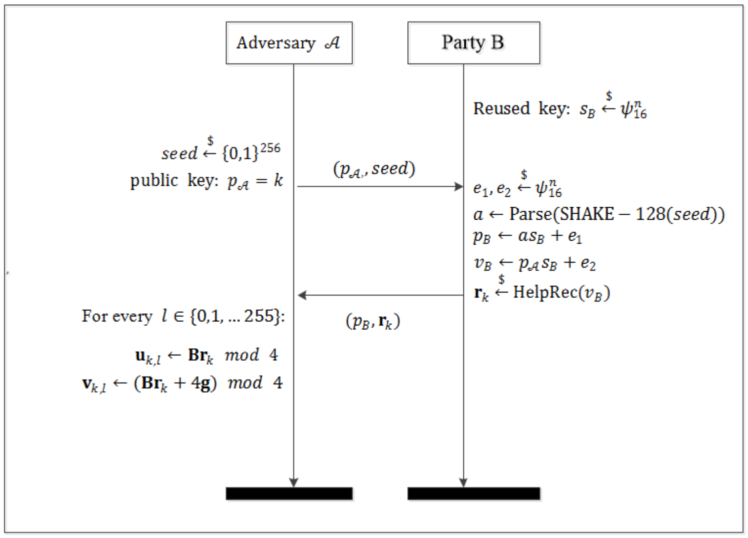
Adversary $\mathscr{A}$
Party B
Reused key: $s_B \overset{\$}{\leftarrow} \psi_{16}^n$
$seed \overset{\$}{\leftarrow} \{0,1\}^{256}$
public key: $p_{\mathscr{A}} = k$
$(p_{\mathscr{A}}, seed)$
$e_1, e_2 \overset{\$}{\leftarrow} \psi_{16}^n$
$a \leftarrow \text{Parse}(\text{SHAKE}-128(seed))$
$p_B \leftarrow a s_B + e_1$
$v_B \leftarrow p_{\mathscr{A}} s_B + e_2$
$\mathbf{r}_k \overset{\$}{\leftarrow} \text{HelpRec}(v_B)$
$(p_B, \mathbf{r}_k)$
For every $l \in \{0,1, \ldots 255\}$:
$\mathbf{u}_{k,l} \leftarrow \mathbf{Br}_k \ mod \ 4$
$\mathbf{v}_{k,l} \leftarrow (\mathbf{Br}_k + 4\mathbf{g}) \ mod \ 4$

Fig. 1. One key exchange session of our key reused attack on NewHope key exchange.

2. For every $l \in \{0, 1, \ldots, \frac{n}{4} - 1\}$:
   a. For every $k \in \{0, 1, \ldots, q-1\}$, set $\mathbf{r}_{k,l} := (r_{k,l}, r_{k,l+\frac{n}{4}}, r_{k,l+2\cdot\frac{n}{4}}, r_{k,l+3\cdot\frac{n}{4}})$ and set vectors: $\mathbf{u}_{k,l} := \mathbf{Br}_{k,l} \bmod 4$; $\mathbf{v}_{k,l} := (\mathbf{Br}_{k,l} + 4\mathbf{g}) \bmod 4$.
   b. For every $i \in \{0, 1, 2, 3\}$:
      i. Set two sequences: $U_{l+\frac{n}{4}\cdot i} := \{\mathbf{u}_{0,l}[i], \mathbf{u}_{1,l}[i], \ldots, \mathbf{u}_{q-1,l}[i]\}$; $Z_{l+\frac{n}{4}\cdot i} := \{\mathbf{v}_{0,l}[i], \mathbf{v}_{1,l}[i], \ldots, \mathbf{v}_{q-1,l}[i]\}$.
      ii. Use $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$ to compute secret coefficient $s_B[l + \frac{n}{4} \cdot i]$.

Figure 1 is one instance of query of our attack. The most important step of our attack is in step (2.b.ii). In next several sections, we focus on how to compute secret's coefficient $s_B[l + \frac{n}{4} \cdot i]$ using sequences $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$.

4.2 Preparation

For fixed $l \in \{0, 1 \ldots, \frac{n}{4} - 1\}$ and $i \in \{0, 1, 2, 3\}$. In this section we describe that a special sequence can be selected from two sequences $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$. We show that this special sequence has "periodic" property, which is the key of our attack. And we define two periodic function to analysis the property of this special sequence.

The special sequence. Here, we describe a special sequence. If we look at oracle $\mathcal{S}$'s computation of the key v_B, we have $v_B = p_{\mathscr{A}} s_B = k \cdot s_B \bmod q$ where $k \in \mathbb{Z}$. This results in the *split vector* of v_B to be: $\mathbf{x}_{k,l} = (ks_B[l], ks_B[l + \frac{n}{4}], ks_B[l + 2 \cdot \frac{n}{4}], ks_B[l + 3 \cdot \frac{n}{4}])^T \bmod q$. For function $\text{HelpRec}(\mathbf{x}_{k,l}; b)$ in oracle $\mathcal{S}$, we firstly consider the case when $b = 0$ (The case when b is randomly selected from $\{0, 1\}$

is described in Section 4.4). Then suppose $\mathbf{r}_{k,l} = \mathrm{HelpRec}(\mathbf{x}_{k,l}; 0) \in \mathbb{Z}_4^4$, we can set

$$\hat{\mathbf{r}}_{k,l} := \mathrm{CVP}_{\tilde{D}_4}(\frac{4}{q}\mathbf{x}_{k,l}) \in \mathbb{Z}^4 \text{ and } \mathbf{a}_{k,l} := \mathbf{B}\hat{\mathbf{r}}_{k,l} \in \mathbb{R}^4.$$

By the definition of algorithm 1, it's easy to find that $\mathbf{a}_{k,l}$ is the closest vector of vector $\frac{4}{q}\mathbf{x}_{k,l}$ in lattice $\tilde{D}_4$ and there is equation:

$$\mathbf{a}_{k,l}[i] = \begin{cases} \lfloor \frac{4}{q}(ks_B[l+\frac{n}{4}\cdot i] \bmod q) \rceil & ||\frac{4}{q}\mathbf{x}_{k,l} - \lfloor \frac{4}{q}\mathbf{x}_{k,l} \rceil ||_1 < 1 \\ \lfloor \frac{4}{q}(ks_B[l+\frac{n}{4}\cdot i] \bmod q) - \frac{1}{2} \rceil + \frac{1}{2} & \text{others.} \end{cases} \tag{4}$$

Obviously, the sequence $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ has some "periodic" property.

And here we describe the relationships between $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ and the two sequences $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$. Since $\mathbf{r}_{k,l} = \hat{\mathbf{r}}_{k,l} \bmod 4$, it's easy to verify that:

$$\mathbf{a}_{k,l} \bmod 4 = \mathbf{B}\mathbf{r}_{k,l} \bmod 4 \text{ or } \mathbf{a}_{k,l} \bmod 4 = (\mathbf{B}\mathbf{r}_{k,l} + 4\mathbf{g}) \bmod 4.$$

Note that in the step (2.a) of the attack, the adversary sets $\mathbf{u}_{k,l} := \mathbf{B}\mathbf{r}_{k,l} \bmod 4$; $\mathbf{v}_{k,l} := (\mathbf{B}\mathbf{r}_{k,l} + 4\mathbf{g}) \bmod 4$. Thus, one of vectors $\mathbf{u}_{k,l}$ and $\mathbf{v}_{k,l}$ is $\mathbf{a}_{k,l} \bmod 4$ and the other is $(\mathbf{a}_{k,l} + 4\mathbf{g}) \bmod 4$. Hence every element in $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1\ldots,q-1}$ is in one of sequences $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$.

Two periodic function. Here we say that $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ reveals some information of the secret s_B. To analysis the "periodic" property of $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, we can define the following two functions:

$$f_{0,x}(h) := \lfloor \frac{4}{q} \cdot (h \cdot x \bmod q) \rceil \bmod 4;$$
$$f_{1,x}(h) := \left(\lfloor \frac{4}{q} \cdot (h \cdot x \bmod q) - \frac{1}{2} \rceil + \frac{1}{2} \right) \bmod 4.$$

where $x \in \mathbb{Z}$, $h \in \mathbb{R}$. When $x = 0$, $f_{0,0}(h) = 0$ and $f_{1,0}(h) = 0.5$. When $x \neq 0$, $f_{0,x}(h)$ and $f_{1,x}(h)$ are periodic functions with the fundamental period $N := \frac{q}{|x \bmod^{\pm} q|}$. We denote the number of period for function $f(h)$ in domain $[0,q)$ as $P_{f,[0,q)}$. Then $P_{f_{0,x},[0,q)} = P_{f_{1,x},[0,q)} = |x \bmod^{\pm} q|$. One instance of function $f_{0,x}(k)$ and $f_{1,x}(k)$ is illustrated by figure 2. For sequence $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, equation

$$\mathbf{a}_{k,l}[i] \bmod 4 = f_{0,s_B[l+\frac{n}{4}\cdot i]}(k) \text{ or } \mathbf{a}_{k,l}[i] \bmod 4 = f_{1,s_B[l+\frac{n}{4}\cdot i]}(k) \tag{5}$$

holds for every $k \in \{0, 1, \ldots, q-1\}$. Then for such sequence $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, we define its number of period as $P_{\mathbf{a}_{k,l}[i] \bmod 4,[0,q)} := P_{f_{0,s_B[l+\frac{n}{4}\cdot i]},[0,q)} = |s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q|$. Thus, the attacker can compute $|s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q|$ by computing $P_{f_{0,s_B[l+\frac{n}{4}\cdot i]},[0,q)}$ or $P_{f_{1,s_B[l+\frac{n}{4}\cdot i]},[0,q)}$ once he has obtained $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$.

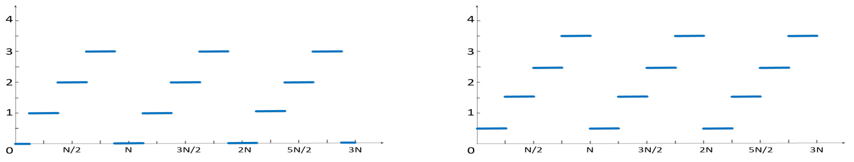

Fig. 2. Left figure represent $f_{0,3}(k)$ and the right figure represent $f_{1,3}(k)$, where $k \in [0, 12289]$, $N = \frac{12289}{3}$ which is defined in Section 4.2. The Vertical Axis for the value of function, and the Horizontal Axis for k.

4.3 Recover the secret key

In this section, we describe how to compute every coefficient of s_B using sequence $U_{l+\frac{n}{4}\cdot i}$ and $Z_{l+\frac{n}{4}\cdot i}$. This section is divided into two parts. The first part aims to select the sequence $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ described in above section, and the second part aims to compute every coefficient of s_B using the sequence we select.

Select sequence. Here, we describe how to obtain $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ using $2q$ values $(\mathbf{u}_{k,l}[i])_{k=0,1,\ldots,q-1}$ and $(\mathbf{v}_{k,l}[i])_{k=0,1,\ldots,q-1}$.

In order to obtain $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, we have to find the relationships between the two adjacent elements $\mathbf{a}_{k,l}[i] \bmod 4$ and $\mathbf{a}_{k+1,l}[i] \bmod 4$. By equation (5), if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q > 0$, there may exists some $k = \lfloor \frac{nN}{8} \rfloor$ where $n \in \mathbb{Z}$, such that:

$$(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = 1; \tag{6}$$

and if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q < 0$, there may exists some $k = \lfloor \frac{nN}{8} \rfloor$ where $n \in \mathbb{Z}$ such that:

$$(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = -1; \tag{7}$$

and for others k in above two cases, there are:

$$-0.5 \leq (\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod^{\pm} 4 \leq 0.5. \tag{8}$$

We can select the sequence $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$ using relations (6), (7) and (8). Firstly, to ensure the sequence we select is $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, we set the first number of the sequence:

$$w_0 = \begin{cases} \mathbf{u}_{0,l}[i] & \text{if } |\mathbf{u}_{0,l}[i] - 2| > |\mathbf{v}_{0,l}[i] - 2|; \\ \mathbf{v}_{0,l}[i] & \text{else.} \end{cases}$$

Here, we will select two sequences - the sequence W for the case when $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q > 0$ and the other sequence T for the case when $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q < 0$. Define a mapping:

$$\begin{aligned} \psi : \mathbb{R}^{2q} &\rightarrow \mathbb{R}^{2q}, \\ (U_{l+\frac{n}{4}\cdot i}, Z_{l+\frac{n}{4}\cdot i}) &\mapsto (W, T) \end{aligned} \tag{9}$$

where the elements in sequence $W = (w_k)_{k=0,\ldots,q-1}$ satisfy:

$$w_k = \begin{cases} w_0 & \text{when } k = 0; \\ \mathbf{u}_{k,l}[i] & \begin{array}{l}\text{when } k > 0 \text{ and } (-0.5 \le (\mathbf{u}_{k,l}[i] - w_{k-1}) \bmod^{\pm} 4 \le 0.5 \text{ or} \\ (\mathbf{u}_{k,l}[i] - w_{k-1}) \bmod 4 = 1);\end{array} \\ \mathbf{v}_{k,l}[i] & \text{others.} \end{cases}$$

and the elements in sequence $T = (t_k)_{k=0,\ldots,q-1}$ satisfy:

$$t_k = \begin{cases} w_0 & \text{when } k = 0; \\ \mathbf{u}_{k,l}[i] & \begin{array}{l}\text{when } k > 0 \text{ and } (-0.5 \le (\mathbf{u}_{k,l}[i] - t_{k-1}) \bmod^{\pm} 4 \le 0.5 \text{ or} \\ (\mathbf{u}_{k,l}[i] - t_{k-1}) \bmod 4 = -1);\end{array} \\ \mathbf{v}_{k,l}[i] & \text{others.} \end{cases}$$

Note that if equation (8) holds for every $k \in [0, q)$, $W = T = (\mathbf{a}_{k,l}[i] \bmod$

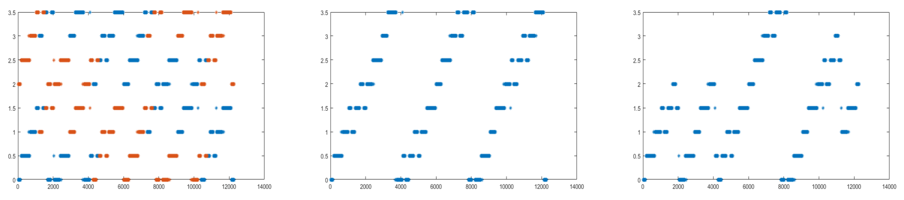
(a) (b) (c)

Fig. 3. An experimental data when one split vector of v_B is $\mathbf{x}_{k,l} = k \cdot (5, -7, 3, 4) \bmod q$. We put all values of $(\mathbf{u}_{k,l}[2])_{k=0,1,\ldots,q-1}$ and $(\mathbf{v}_{k,l}[2])_{k=0,1,\ldots,q-1}$ in the figure (a). Our selected sequence W is illustrated in the figure (b). And selected sequence T is illustrated in the figure (c). We can note that W has "period" property, while T is "out-of-order". The Vertical Axis for the value of the elements in sequence, and the Horizontal Axis for k.

$4)_{k=0,1,\ldots,q-1}$. If $W \neq T$, one of the two sequences W and T is $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$. And if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q > 0$, W is $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, while if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q < 0$, T is $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$. One instance of experimental data of our selecting technique is illustrated by figure 3, and we note that in this instance $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q = 3$, so $W = (\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$.

Compute the coefficients. Now, we show how to compute $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q$ using sequences W and T. In Section 4.2, we know that $P_{\mathbf{a}_{k,l}[i] \bmod 4, [0,q)} = |s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q|$. Construct an algorithm Counting to compute the number of period of sequence W (or T). Algorithm Counting is listed as Algorithm 3.

We note that computing $P_{\mathbf{a}_{k,l}[i] \bmod 4,[0,q)}$ means that to compute $P_{f_{0,s_B[l+\frac{n}{4}\cdot i]},[0,q)}$ or $P_{f_{1,s_B[l+\frac{n}{4}\cdot i]},[0,q)}$. Since $|s_B[l+\frac{n}{4}\cdot i]| \leq 16$, for periodic functions $f_{0,s_B[l+\frac{n}{4}\cdot i]}$ and $f_{1,s_B[l+\frac{n}{4}\cdot i]}$, there exists h' in every period, such that $f_{0,s_B[l+\frac{n}{4}\cdot i]}(h') = 0$ or $f_{1,s_B[l+\frac{n}{4}\cdot i]}(h') = 0.5$ or 3.5. In algorithm 3, the count number c plus 1 at the point when the element in sequence W firstly equals to 0 or 0.5 or 3.5 in one period interval. Thus every period of function $f_{0,s_B[l+\frac{n}{4}\cdot i]}$ or $f_{1,s_B[l+\frac{n}{4}\cdot i]}$ in domain $[0,q)$ can be counted.

Algorithm 3 Counting

```
Require: $W = \{w_0, w_1, \ldots, w_{q-1}\} \in \mathbb{R}^q$
Ensure: The number of period of sequence $W$.
1: $k \leftarrow 0$; $c \leftarrow 0$
2: while ($w_k \neq 0$ or $w_k \neq 0.5$ or $w_k \neq 3.5$) do
3:     $k \leftarrow k+1$
4: end while
5: $c \leftarrow c+1$
6: while ($w_k \neq 2$ or $w_k \neq 2.5$) do
7:     $k \leftarrow k+1$
8: end while
9: if $k < q-1$ then
10:     goto step 2
11: end if
12: return $c$
```

Suppose $c_1 \leftarrow$ Counting(W) and $c_2 \leftarrow$ Counting(T). If $W = T$, $c_1 = c_2 = |s_B[l+\frac{n}{4}\cdot i] \bmod^{\pm} q|$. Then to determine the sign of $s_B[l+\frac{n}{4}\cdot i] \bmod^{\pm} q$, define:

$$sign = \sum_{\substack{k \in [0,\, q/c]; \\ |w_{k+1} - w_k| < 3}} (w_{k+1} - w_k).$$

where c is the output of Counting(W) and $w_k \in W$. Note that $[0, q/c]$ is one periodic interval. If $s_B[l+\frac{n}{4}\cdot i] \bmod^{\pm} q > 0$, it is easy to verified that $sign > 0$, and if $s_B[l+\frac{n}{4}\cdot i] \bmod^{\pm} q < 0$, there is $sign < 0$.

When $W \neq T$, we want to figure out that which one of c_1 and c_2 equals to $|s_B[l+\frac{n}{4}\cdot i] \bmod^{\pm} q|$. We need a parameter to measure the degree of approximation of the sequence we select and $(\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$. Suppose the input sequence of algorithm 3 is W. Let the value of k be k_j at the point when the algorithm 3 loops to steps 5 for the j-th times $(0 < j \leq c_1)$. Then the domina size of j-th periodic interval of W is $N_j = |k_{j+1} - k_j|$ $(j \in \{1, 2, \ldots, c_1 - 1\})$. Define parameter

$$var_W = \frac{\sum_{j=1}^{c_1-1} |N_j - \frac{\sum_{j=1}^{c_1-1} N_j}{c_1-1}|}{c_1 - 1}, \tag{10}$$

which is the variance of $N_1, \ldots, N_{c_1-1}$. Similarly, we can define var_T for sequence T. If $W = (\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, var_W should be very small because every two number in $\{N_1, \ldots, N_{c_1-1}\}$ are almost equal. Meanwhile sequence T hasn't "periodic" property (one instance is illustrated by figure 3.c), which results in $var_T > var_W$. Similarly, the case when $T = (\mathbf{a}_{k,l}[i] \bmod 4)_{k=0,1,\ldots,q-1}$, $var_T < var_W$. This property is verified in our experiments. Then if $var_W < var_T$, the sign of coefficient $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q$ is "+" and $|s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q| = c_1$, otherwise the sign of $s_B[l + \frac{n}{4} \cdot i]$ is "−" and $|s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q| = c_2$. Thus we can compute every coefficient of s_B.

4.4 Effect of $\mathbf{e_2}$ and parameter b

In our experiments, for the case where e_2 is added to v_B and b is chosen randomly from $\{0, 1\}$ in function HelpRec($\mathbf{x}_{k,l}$;b), we can still get the secret key correctly using algorithm described above. And the following analysis is only theoretical. Firstly, we analysis the case when e_2 is added to v_B. The key computation of B is $v_B = p_{\mathscr{A}} s_B + e_2$. Let $\tilde{\mathbf{x}}_{k,l} = (ks_B[l] + g_0, ks_B[l + \frac{n}{4}] + g_1, ks_B[l + 2 \cdot \frac{n}{4}] + g_2, ks_B[l + 3 \cdot \frac{n}{4}] + g_3)^T \bmod q$ where $g_i \overset{\$}{\leftarrow} \psi_{16}$ is one of *split vector* of v_B. Then if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q > 0$, for equation (4), there may exists k such that

$$||\frac{4}{q}\tilde{\mathbf{x}}_{k,l} - \lfloor \frac{4}{q}\tilde{\mathbf{x}}_{k,l} \rceil||_1 < 1; \; ||\frac{4}{q}\tilde{\mathbf{x}}_{k+1,l} - \lfloor \frac{4}{q}\tilde{\mathbf{x}}_{k+1,l} \rceil||_1 < 1$$

and

$$\lfloor \frac{4}{q}(ks_B[l + \frac{n}{4} \cdot i] + g_i \bmod q) \rceil - \lfloor \frac{4}{q}((k+1)s_B[l + \frac{n}{4} \cdot i] + g_i' \bmod q) \rceil = 1 \quad (11)$$

where $g_i, g_i' \overset{\$}{\leftarrow} \psi_{16}$. Equation (11) is equivalent to: $(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = -1$. Thus the two sequences selected by mapping (9) will be both wrong. Similarly, if $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q < 0$, equation $(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = 1$ may occurs. We hope to eliminate such wrong points in mapping (9). Our idea of improvement is to detect such wrong point and "delete" it. We add the following two steps to the mapping (9): 1) when selecting the sequence W, for k such that $|(\mathbf{u}_{k,l}[i] - w_{k-1}) \bmod^{\pm} 4| = 1$ and $|(\mathbf{u}_{k,l}[i] - \mathbf{u}_{k+1,l}[i]) \bmod^{\pm} 4| = 1$, let $w_k = w_{k-1}$; 2) when selecting the sequence T, for k such that $|(\mathbf{u}_{k,l}[i] - t_{k-1}) \bmod^{\pm} 4| = 1$ and $|(\mathbf{w}_{k,l}[i] - \mathbf{w}_{k+1,l}[i]) \bmod^{\pm} 4| = 1$, let $t_k = t_{k-1}$. Since such wrong points are sparse in $[0, q)$ (actually in our experiments we failed to found such wrong points), the "periodic" property of sequence we select by this way doesn't change, hence this improvement hasn't influence to the algorithm Counting. Other steps of the attack is all the same with the case when e_2 is not added to v_B.

Similarly, we consider the case when b is chosen randomly from $\{0, 1\}$ in function HelpRec($\mathbf{x}_{k,l}$;b). Firstly consider the case when $s_B[l + \frac{n}{4} \cdot i] \bmod^{\pm} q > 0$. Suppose two inputs for function HelpRec($\mathbf{x}$;b) is $(\mathbf{x}_{k,l}, 1)$ and $(\mathbf{x}_{k+1,l}, 0)$. Then for equation (4), there may exists k such that:

$$||\frac{4}{q}(\mathbf{x}_{k,l} + \mathbf{g}) - \lfloor \frac{4}{q}(\mathbf{x}_{k,l} + \mathbf{g}) \rceil||_1 < 1; ||\frac{4}{q}\mathbf{x}_{k+1,l} - \lfloor \frac{4}{q}\mathbf{x}_{k+1,l} \rceil||_1 < 1$$

and

$$\lfloor\frac{4}{q}((ks_B[l+\frac{n}{4}\cdot i]+2) \bmod q))\rceil - \lfloor\frac{4}{q}((k+1)s_B[l+\frac{n}{4}\cdot i] \bmod q)\rceil = -1$$

hold. This equals to $(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = -1$. Similarly for the case when $s_B[l+\frac{n}{4}\cdot i]\mathrm{mod}^{\pm} q < 0$, $(\mathbf{a}_{k+1,l}[i] \bmod 4 - \mathbf{a}_{k,l}[i] \bmod 4) \bmod 4 = 1$. Note that such wrong points is same with the case when e_2 is added to v_B described in above paragraph, and we can use the same way to eliminate such wrong points in mapping (9).

4.5 Adversary time complexity

From the above description of our attack, it is clear that the adversary needs q queries to recover every coefficient of s_B. The time complexity of selecting the sequence W and T for every coefficient of s_B is $2q$ and there needs about $2q$ times to compute the exact value of the coefficient using sequences W and T. Suppose times of once querying is t. Since q is $O(n)$, the query complexity of the complete attack is $q = O(n)$ and times complexity is $qt + n\cdot(2q+2q) = O(n^2)$.

5 Conclusion

In this work, we have presented an detailed key reused attack on NewHope key exchange in recovering the secret of a reused key with q queries. We show that for NewHope key exchange, when the public key is fixed for a long term, an active adversary can collect a sequence of the *signal* and construct a sequence with "periodic" property, which reveals the information of the secret. The adversary can exploits the "periodic" property of the sequence and recovers the secret key of the honest party. We believe that such strategy of the key reuse attack can also be adapted to NISTPQC submission NewHope IND-CPA KEM [1]. But the NewHope IND-CCA KEM would stop the attack. This version of key exchange applies the Fujisaki-Okamoto transform and achieves CCA security.

Acknowledgments. This article is supported by The National Key Research and Development Program of China (Grant No. 2017YFA0303903). Authors thank Yang Yu for discussions and the anonymous ICISC'18 reviewers for helpful comments.

References

1. Alkim, E. Avanzi, R. Bos, J. W. Ducas, L. NewHope, Algorithm Specifcations and Supporting Documentation. Version 1.0 - November 30, 2017. Submission to NIST, https://newhopecrypto.org/data/NewHope_2017_12_21.pdf.
2. Alkim, E. Ducas, L. Pöppelmann, T. Schwabe, P. Post-quantum key exchange - a new hope, in *Proceedings Of the 25th Usenix Security Symposium.* USENIX Association, pp. 327-343.

3. Bos, J. W. Costello, C. Ducas, L. Mironov, I. Naehrig, M. Nikolaenko, V. Raghunathan, A. Stebila, D. 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 Press, pp. 1006-1018.
4. Bos, J. W. Costello, C. Naehrig, M. Stebila, D. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem, in *2015 Ieee Symposium on Security And Privacy*, pages 553-570. IEEE Computer Society Press, May 2015.
5. Bos, J. Ducas, L. Kiltz, E. Lepoint, T. Lyubashevsky, V. Schanck, J. M. Schwabe, P. Stehlé, D. CRYSTALS-Keyber: a CCA-secure module-lattice-base KEM. Cryptology ePrint Archive, Report 2017/634, 2017, http://eprint.iacr.org/2017/634.
6. Ding, J. T. Alsayigh, S. Saraswathy, R. V. Fluhrer, S. Lin, X. D. Leakage of Signal function with reused keys in RLWE key exchange, in *2017 Ieee International Conference on Communications (Icc).*
7. Ding, J. T. Fluhrer, S. Saraswathy, R. V. Complete attack on RLWE key exchange with reused keys, without signal leakage. In *Information Security And Privacy, ACISP 2018*, pp. 467-486.
8. Ding, J. T. Xie, X. X, L. A simple provably secure key exchange scheme based on the learning with errors problem, Cryptology ePrint Archive, Report 2012/688, 2012, http://eprint.iacr.org//2012/688.pdf.
9. Experimenting with post-quantum cryptography (July 2016), https://security/googleblog.com/2016/07/experimenting-with-post-quantum.html.
10. Fluhrer, S. Cryptanalysis of ring-lwe based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085, 2016, http://eprint.iacr.org/2016/085.
11. Lynbashevsky, V. Peikert, C. Regev, O. On Ideal Lattices and Learning with Errors over Rings, in *Advances in Cryptology - Eurocrypt 2010*, ser. LNCS, H. Gilbert, Ed. Springer Berlin / Heidelberg, 2010, vol. 6110, pp. 1-23.
12. Micciancio, D. Regev, O. Worst-case to average-case reductions based on Gaussian measures, in *Siam Journal on Computing*, vol. 37, pp. 267-302, April 2007.
13. Peikert, C. Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem, in *Proceedings Of the 2009 Acm Symposium on Theory Of Computing*, ser. STOC '09, New York, NY, USA: ACM, 2009, pp. 333-342.
14. Peikert, C. Lattice Cryptography for the Internet, in *Post-Quantum Cryptography, Pqcrypto 2014*, pp. 197-219.
15. Regev, O. On Lattices, Learning with Errors, Random Linear Codes, and Cryptography, in *Proceedings of the thirty-seventh annual ACM symposium on Theory of computing.* pp. 84-93. STOC'05, CM, New York, NY, USA (2005).
16. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, in *Siam Journal on Computing,* pp. 1484-1509 (Oct 1997).
17. The Internet Defense Prize, https://internetdefenseprize.org/.
18. Zhang, J. Zhang, Z. F. Ding, J. T. Snook, M. Dagdelen, O. Authenticated Key Exchange from Ideal Lattices, in *Advances In Cryptology - Eurocrypt 2015*, pp. 719-751. Springer(2015).