Light the Signal: Optimization of Signal Leakage Attacks against LWE-Based Key Exchange

Yue Qin[1], Ruoyu Ding[1], Chi Cheng[1], Nina Bindel[2], Yanbin Pan[3], and Jintai Ding[4]

[1] China University of Geosciences, Wuhan, 430074, China
chengchi@cug.edu.cn
[2] IQC and University of Waterloo, Waterloo, ON, Canada
nlbindel@uwaterloo.ca
[3] Key Laboratory of Mathematics Mechanization, Academy of Mathematics and Systems Science, Chinese Academy of Sciences
[4] Tsinghua University, Beijing

Abstract. Key exchange protocols from the learning with errors (LWE) problem share many similarities with the Diffie–Hellman–Merkle (DHM) protocol, which plays a central role in securing our Internet. Therefore, there has been a long time effort in designing authenticated key exchange directly from LWE to mirror the advantages of DHM-based protocols. In this paper, we revisit signal leakage attacks and show that the severity of these attacks against LWE-based (authenticated) key exchange is still underestimated.

In particular, by converting the problem of launching a signal leakage attack into a coding problem, we can significantly reduce the needed number of queries to reveal the secret key. Specifically, for DXL-KE we reduce the queries from 1,266 to only 29, while for DBS-KE, we need only 748 queries, a great improvement over the previous 1,074,434 queries. Moreover, our new view of signals as binary codes enables recognizing vulnerable schemes more easily. As such we completely recover the secret key of a password-based authenticated key exchange scheme by Dabra et al. with only 757 queries and partially reveal the secret used in a two-factor authentication by Wang et al. with only one query. The experimental evaluation supports our theoretical analysis and demonstrates the efficiency and effectiveness of our attacks. Our results caution against underestimating the power of signal leakage attacks as they are applicable even in settings with a very restricted number of interactions between adversary and victim.

Keywords: Post-quantum cryptography · Key exchange · Learning with errors · Signal leakage attack

1 Introduction

The past decades have seen the rapid developments in *post-quantum* (PQ) cryptography, i.e., cryptographic primitives that are secure even against attackers having access to a quantum computer. Examples for such PQ cryptography are primitives based on Regev's learning with errors (LWE) problem [34]. Interestingly, LWE-based key exchange protocols share many similarities with the famous and elegant Diffie–Hellman–Merkle (DHM) protocol [15]. In a nutshell, Alice computes and sends $\mathbf{P}_A = \mathbf{a}\mathbf{s}_A + \mathbf{e}_A$ to Bob, while Bob responds with $\mathbf{P}_B = \mathbf{a}\mathbf{s}_B + \mathbf{e}_B$. In contrast to the shared secret resulting from a DHM key exchange, Alice and Bob only agree on an approximately equal value $\mathbf{a}\mathbf{s}_A\mathbf{s}_B$. To enable establishing exactly the same key, Ding, Xie, and Lin [19] introduced a *signal* function, which indicates whether an element belongs to a fixed interval or not and that has been used to construct an LWE-based key exchange, called DXL-KE. Similarly, in 2014, Peikert [30] suggested a *reconciliation* function that has been used to construct key encapsulation mechanisms (KEMs), which were then instantiated and tested in the transport layer security (TLS) protocol by Bos, Costello, Naehrig, and Stebila [9]. While the progress regarding LWE-based key exchange seems promising, in practice we need efficient *authenticated* key exchange (AKE), as well as more advanced protocols, such as password or two-factor authentication.

LWE-based AKEs can be achieved by instantiating generic constructions from public-key encryption (PKE) or KEMs. For example, recently quantum-safe AKEs from lattice-based KEMs for the TLS [35,36,22] and for the Signal protocol [10,23] have been constructed. However, most classical AKEs avoid generic transforms and construct them directly from DHM, e.g., [26,27,25]. The only AKE constructed directly from LWE (inspired by [25]), has been presented by Zhang, Zhang, Ding, Snook, and Dagdelen [40] in 2015. Mirroring the ideas of DHM-based protocols for LWE-based ones is challenging, because protocols using signal or reconciliation functions are often vulnerable to *key reuse* attacks.

Key reuse attacks have a long history starting with Bleichenbacher's remarkable attack against RSA PKCS#1 [8] and the key reuse attacks against the DHM key exchange proposed by Menezes and Ustaoglu [28]. There are essentially two types of key reuse attacks against LWE-based key exchange schemes.

The first one is called the *key mismatch* attack, which aims to recover the secret by checking whether the shared keys of both parties match or not when Alice's key is reused. Ding, Fluhrer and Saraswathy first proposed a key mismatch attack against DXL-KE [18]. Recently, key mismatch attacks have been adopted to analyze candidates[1] of NIST's PQ cryptography project, such as NewHope [5,31,29], Kyber [32], LAC [21], NTRU-HRSS [41], and others [4,24,33].

Another example of key reuse attack against LWE-based key exchange is the *signal leakage* attack. Fluhrer [20] has been the first to show that the signal function reveals secret key information. In a follow-up work, Ding, Alsayigh, Saraswathy, Fluhrer, and Lin [16] attacked DXL-KE using signal leakage. The idea of the attack is that the adversary sends $\mathbf{P}_A = k$ for increasing k instead of an honestly generated $\mathbf{P}_A = \mathbf{a}\mathbf{s}_A + \mathbf{e}_A$. From Bob's honestly generated response including the signal, the adversary can determine the absolute value of the secret

[1] It is important to point out that these attacks are against candidates designed to resist passive adversaries. Hence, security claims are not invalidated by these attacks.

coefficients by counting how often the signal switches between 0 and 1. For DXL-KE, about 98,310 interactions between the adversary and Bob (also called *queries*) are required to successfully launch the attack. Recently, Bindel, Stebila and Veitch [7], proposed a *sparse signal collection* method to reduce the number of needed queries to 1,266.

To counter signal leakage attacks, Ding, Branco, and Schmitt [17], proposed a *pasteurization* technique to construct a key exchange called DBS-KE, which is claimed to be robust against key reuse attacks. They also introduced an authenticated key exchange (DBS-AKE) and proved its security in the Bellare–Rogaway (BR) model [6]. Similar pasteurization techniques have been used in other authentication/key exchange protocols such as the password-based AKE called LBA-PAKE [13], Seyhan, Nguyen, Akleylek, Cengiz, and Islam's key exchange [37], and Akleylek and Seyhan's AKE [2]. There also exist other techniques to thwart signal leakage attacks, for example a smart card based two factor authenticated key exchange for mobile devices, called Quantum2FA [39]. The basic idea of Quantum2FA is to resist signal leakage attacks by putting the public key shares in the smart card in advance.

While signal leakage attacks have been widely known and considered for LWE-based key exchanges, we argue that the severity of this kind of attack is still underestimated. Just recently, Bindel, Stebila and Veitch [7] used their sparse signal leakage attack to reveal the secret key used in DBS-KE, showing that the protocol is not robust against key reuse. Their attack against DBS-KE, needs about 1,074,434 queries.

In this paper, we further caution against underestimating the power of signal leakage attacks. In particular, we present a new view on the attack by representing the signals as binary codes. This novel perspective enables our *targeted signal extraction* approach which decreases the number of needed queries drastically. More concretely, we are able to reveal Bob's secret used in the DXL-KE from 1,266 to only 29 needed queries (i.e., we reduced the number by a factor of 43). For DBS-KE, the improvement is even stronger, namely we reduce the number of queries from 1,074,434 to only 748 (i.e., an improvement of factor 1,436). This makes the attack feasible in settings where the attacker is only able to run a very limited number of sessions. At the same time, our results caution strongly against, e.g., allowing key reuse for a restricted number of times, as further improvements might be possible. For example, in our analysis against Quantum2FA, we are able to recover part of the secret key without key reuse. That is, revealing about 50% of the secret key used in the two-factor authentication Quantum2FA with just a single query. While this does not reveal the entire secret, it decreases the bit security considerably.

Furthermore, signal representation as binary codes and the resulting targeted signal extraction, enables to recognize vulnerable schemes more easily. As such, we successfully carry out a signal leakage attack against the password-based authentication LBA-PAKE using 757 queries. Since these schemes are claimed to be secure against signal leakage attacks, our results show that although signal leakage attacks are well-known, it seems challenging to construct robust schemes and to spot their vulnerabilities.

All our attacks are supported by experimental evaluation that matches our theoretical analysis well. To recover a complete secret key, the average time needed for our proposed attacks on DXL-KE is 0.44 seconds, while our proposed attack on DBS-KE costs 6.53 seconds. As a comparison, the attacks presented in [7] need more than 24.1 and 582.08 seconds against DXL-KE and DBS-KE, respectively. Meanwhile, it costs less than 8 seconds to completely recover the long-term secret key of LBA-PAKE. For Quantum2FA, on average we can successfully recover 54.57% of 512 coefficients using one query.

In addition and on a more theoretical level, viewing the signals used in LWE-based key exchange as binary codes, supports the strong connection between these two field of research. Similarities between lattices and binary codes have gained more attention recently and have been systematically analyzed by Debris-Alazard, Ducas, and van Woerden [14].

For the remainder of this paper, we first recall signal leakage attacks in Section 2. We continue in Section 3, with describing how to define the binary codes and our targeted signal extraction. In Section 4, we give the details on how to apply the attack to DXL-KE, and in Section 5 to DBS-KE, LBA-PAKE, and Quantum2FA. We explain our experimental results in Section 6.

2 Background

Notations. For a prime q and a dimension n, let the polynomial ring $\mathcal{R}_q$ be $\mathbb{Z}_q[x]/\langle x^n+1\rangle$ with $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z} = \{-\frac{q-1}{2}, \cdots, \frac{q-1}{2}\}$. All polynomials are in bold lower-case letters, and we use $\mathbf{c}[i]$ $(0 \le i \le n-1)$ to represent the i-th coefficient of the polynomial $\mathbf{c} \in \mathcal{R}_q$. We denote by $\mathbf{p} \leftarrow \chi_\alpha$ sampling every coefficient of $\mathbf{p}$ from a discrete Gaussian distribution over $\mathbb{Z}$ with standard deviation α. The operation $\lfloor x \rfloor$ represents the maximum integer not exceeding x, while $\lceil x \rceil$ represents the minimal integer greater than or equal to x; we define $\lfloor x \rceil = \lfloor x + \frac{1}{2} \rfloor$.

For prime $q > 2$, set $E = \{-\lfloor \frac{q}{4} \rfloor, \cdots, \lfloor \frac{q}{4} \rceil\}$. The aim of a signal function $Sig(x)$ is to tell whether the coefficients of the shared key belong to E or not. Specifically, we define

$$Sig(x) = \begin{cases} 0 & \text{if } x \in E, \\ 1 & \text{otherwise,} \end{cases} \tag{1}$$

for $x \in \mathbb{Z}_q$. Further, $Sig(\mathbf{p}) = (Sig(\mathbf{p}[i]))_{i=0,\ldots,n-1} \in \{0,1\}^n$ is a natural extension to each of the polynomial coefficients.

Moreover, we define

$$Mod_2(x, \omega) = \left(x + \omega \cdot \frac{q-1}{2} \right) \mod q \mod 2, \tag{2}$$

and its coefficient-wise extension to $Mod_2(\mathbf{p}, \omega)$ to polynomials.

DXL-KE. We depict the details of DXL-KE in Fig. 1, which has been one of

Alice		**Bob**	Oracle $\mathcal{O}_{\mathbf{s}_B}(\mathbf{P}_A)$:
$\mathbf{s}_A, \mathbf{e}_A \leftarrow \chi_\alpha$			$\mathbf{e}_B, \mathbf{g}_B \leftarrow \chi_\alpha$
$\mathbf{P}_A \leftarrow \mathbf{a}\mathbf{s}_A + 2\mathbf{e}_A$	$\xrightarrow{\mathbf{P}_A}$	$\mathbf{s}_B, \mathbf{e}_B, \mathbf{g}_B \leftarrow \chi_\alpha$	$\mathbf{P}_B \leftarrow \mathbf{a}\mathbf{s}_B + 2\mathbf{e}_B$
		$\mathbf{P}_B \leftarrow \mathbf{a}\mathbf{s}_B + 2\mathbf{e}_B$	$\mathbf{K}_B \leftarrow \mathbf{P}_A\mathbf{s}_B + 2\mathbf{g}_B$
$\mathbf{g}_A \leftarrow \chi_\alpha$		$\mathbf{K}_B \leftarrow \mathbf{P}_A\mathbf{s}_B + 2\mathbf{g}_B$	$\omega_B \leftarrow Sig(\mathbf{K}_B)$
$\mathbf{K}_A \leftarrow \mathbf{P}_B\mathbf{s}_A + 2\mathbf{g}_A$	$\xleftarrow{(\mathbf{P}_B, \omega_B)}$	$\omega_B \leftarrow Sig(\mathbf{K}_B)$	Return $(\mathbf{P}_B, \omega_B)$
$SK_A \leftarrow Mod_2(\mathbf{K}_A, \omega_B)$		$SK_B \leftarrow Mod_2(\mathbf{K}_B, \omega_B)$	

Fig. 1: Pseudo-code description of DXL-KE (left) and oracle $\mathcal{O}$ (right)

the first schemes to be attacked by signal leakage attacks [16]. In this kind of attacks, it is assumed that Bob's private key $\mathbf{s}_B$ is reused. An active adversary $\mathcal{A}$ impersonating Alice, deliberately chooses appropriate $\mathbf{P}_A$ and tries to recover the secret key of Bob. To formalize this, we define an oracle $\mathcal{O}$ that reuses Bob's secret key $\mathbf{s}_B$ as also shown in Fig. 1. The parameters for DXL-KE are $n = 1024$, $\alpha = 3.197$, $q = 2^{14} + 1 = 16,385$.

Signal Leakage Attacks. The signal leakage attack can be divided into two steps. In Step 1, the adversary $\mathcal{A}$ recovers the absolute value of each $\mathbf{s}_B[i]$ for $i = 0, 1, \cdots, n-1$. To launch the attack, $\mathcal{A}$ queries $\mathbf{P}_A = k$ to the oracle $\mathcal{O}_{s_B}$, which returns $\mathbf{P}_B$ and ω_B. Increasing k from 0 to $q-1$, yields q signals for each $\mathbf{s}_B[i]$. The oracle computes $\mathbf{K}_B = \mathbf{P}_A\mathbf{s}_B + 2\mathbf{g}_B = k\mathbf{s}_B + 2\mathbf{g}_B$ to get ω_B. Ignoring the error $2\mathbf{g}_B$ for simplicity, when $\mathbf{K}_B[i]$ enters or leaves the interval $[-\lfloor \frac{q}{4} \rfloor, \lfloor \frac{q}{4} \rceil]$, the corresponding signal flips. For example, if $\mathbf{s}_B[i] = \pm 1$, $|\mathbf{K}_B[i]| = |(\mathbf{P}_A\mathbf{s}_B)[i]| = k|\mathbf{s}_B[i]| = k$. As k changes from 0 to $q-1$, the signal $\omega_B[i]$ changes as $0 \to 1 \to 0$, i.e., the signal changes 2 times. As $|\mathbf{s}_B[i]|$ grows, the signal changes more frequent, to be exact the signal changes $2|\mathbf{s}_B[i]|$ times. Counting the signal changes, $\mathcal{A}$ can determine the absolute value of $\mathbf{s}_B[i]$. Since the range of $\mathbf{g}_B$ is small, it may affect only some small regions of the signals when $k\mathbf{s}_B[i]$ is approximate to $\pm\lfloor \frac{q}{4} \rfloor$. Therefore, when counting the number of signals changes, the small (fluctuated) regions where the signal changes frequently, have been mostly ignored in [16]. Bindel, Stebila, and Veitch [7] formalized this by dividing the signals into stable and noisy regions as visualized for absolute values 1, 2, and 3 of $s_B[i]$ in Fig. 2. The blue and yellow bars represent the stable regions consisting of 0s and 1s, respectively; the wavy lines represent the fluctuated (or noisy) regions. When k increases from 0 to $q-1$, fluctuated regions occur when $K_B[i]$ approaches $\pm\ \frac{q}{4}$ due to the error terms. During stable regions, the error terms have no impact on the signal.

In Step 2, the adversary tries to determine the sign of each $\mathbf{s}_B[i]$. Querying $\mathbf{P}_A = (1+x)k$ to $\mathcal{O}_{s_B}$ for $k = 0, \ldots, q-1$, the adversary can recover the pairs $s_B[0] - s_B[n-1], s_B[1] + s_B[2], \ldots, s_B[n-2] + s_B[n-1]$. However, there may be the case that 1 or more consecutive 0s occur in $\mathbf{s}_B$, which prevents deciding the relative sign of two non-zero coefficients. To eliminate this, the adversary needs to set $\mathbf{P}_A = (1 + x^{z+1})k$ to collect enough signals, where z represents the

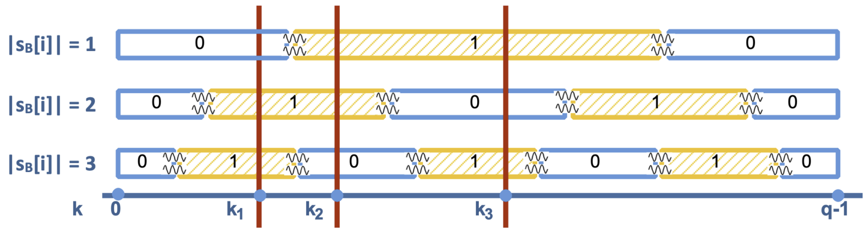

Fig. 2: Alternate stable and fluctuated regions

maximum number of consecutive 0s. To be specific, setting $z = 4$ is sufficient to successfully launch the attack [7]. During the first attack against DXL-KE [16], the adversary queries the oracle $q = 16,385$ times in Step 1 and $(1+z)q = 81,925$ times during Step 2 *for each coefficient of* s_B, which is very inefficient.

An improvement presented in [18] leads to only $\lceil \frac{q}{4} + 2 \rceil = 4,099$ queries for determining the absolute values, and $(1+z)(\lceil \frac{q}{4} + 2 \rceil) = 20,495$ queries to recover each $\mathbf{s}_B[i]$. Since the signal flips when $\mathbf{K}_B[i]$ changes from $\lfloor \frac{q}{4} \rceil$ to $\lfloor \frac{q}{4} \rceil + 1$, and when $\mathbf{s}_B[i] = \pm 1$ the maximum number of queries is $\frac{q}{4} + 2$. Therefore, in this case the adversary needs $(2 + z)(\frac{q}{4} + 2) = 24,594$ queries to recover the secret.

Bindel, Stebila, and Veitch [7] further decreased the number of needed queries by using a *sparse signal collection* method. The idea of their *sparse signal* attack is to collect at least one signal from a stable and at most one from a noisy region. Totally, the adversary needs $36(3 + 2z)\alpha \approx 1,266$ queries. While the number of needed queries is reduced drastically, they still need to sample the signals periodically to count the number of times the signal changes. In what follows, we reduce the number of needed queries even further. This is enabled by viewing the received signals as codewords as explained next.

3 The Targeted Signal Extraction Method

This section presents our new view on LWE-based signal leakage attacks by considering the collected signal as binary codes. This enables a variant of the signal leakage attack that needs only very few queries.

3.1 Description of codewords and Lower bound

Denote by $\mathcal{S} = \{\mathcal{S}_0, \mathcal{S}_1, \cdots, \mathcal{S}_{m_1-1}\}$ the set of all the possible values of $\mathbf{s}_B[i]$, also called the alphabet of $\mathbf{s}_B[i]$ of size m_1. Let $\mathcal{C} = \{\mathcal{C}_0, \mathcal{C}_1, \cdots, \mathcal{C}_{m_2-1}\}$ represent the alphabet of signals $\omega_B[i]$ with $m_2 > 1$ symbols. Taking DXL-KE as an example, $\mathbf{s}_B[i]$ is sampled from discrete Gaussian distribution with standard deviation $\alpha = 3.197$. Hence, the probability of $|\mathbf{s}_B[i]| < 5\alpha = 15.985$ is 99.9999%. Therefore, we choose $\mathcal{S} = \{0, \cdots, 14, 15\}$ with $m_1 = 16$ for absolute value recovery. The alphabet of signal function is $\mathcal{C} = \{0, 1\}$ with $m_2 = 2$.

Assume that the adversary $\mathcal{A}$ accesses the oracle $\mathcal{O}_{s_B}$ (see Fig. 1) t times with t different $\mathbf{P}_{A_j} = k_j,\ j = 1, 2, \cdots, t$. The oracle returns corresponding signals

$$
\begin{array}{ccccccccc}
 & & & & & & \Omega_i & & \\
\mathbf{P}_{A_1} & = k_1 & \Rightarrow \omega_{B_1} & = (\omega_{B_1}[0], & \omega_{B_1}[1], & \cdots, & \omega_{B_1}[i] & , \cdots, & \omega_{B_1}[n-1]) \\
\mathbf{P}_{A_2} & = k_2 & \Rightarrow \omega_{B_2} & = (\omega_{B_2}[0], & \omega_{B_2}[1], & \cdots, & \omega_{B_2}[i] & , \cdots, & \omega_{B_2}[n-1]) \\
\vdots & \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots \\
\mathbf{P}_{A_t} & = k_t & \Rightarrow \omega_{B_t} & = (\omega_{B_t}[0], & \omega_{B_t}[1], & \cdots, & \omega_{B_t}[i] & , \cdots, & \omega_{B_t}[n-1])
\end{array}
$$

Fig. 3: $\mathbf{P}_{A_j}$ and its corresponding $\omega_{B_j} (j = 1, 2, \cdots, t)$

ω_{B_j} where every $\omega_{B_j} = (\omega_{B_j}[0], \omega_{B_j}[1], \cdots, \omega_{B_j}[n-1])$ consists of n bits. It is important to note that for the recovery of $\mathbf{s}_B[i]$, it is sufficient to extract the i-th coefficient from each signal ω_{B_j} $(j = 1, 2, \cdots, t)$. We denote this signal sequence by $\Omega_i = (\omega_{B_1}[i], \omega_{B_2}[i], \cdots, \omega_{B_t}[i]) \in \mathcal{C}^t$ as shown in Fig. 3 and will refer to it as the *(targeted) signal sequence*. We can regard $\Omega_i \in \mathcal{C}^t$ as a codeword and establish a map $\mathcal{M} : \mathcal{C}^t \to \mathcal{S}$ which maps a codeword to the absolute value of a coefficient of s_B. Since, we would like to be able to determine every possible value $|s_B[i]|$ with high probability, the map $\mathcal{M}$ needs to be surjective with high probability. Surjectivity of $\mathcal{M}$ implies that there is at least one codeword corresponding to every element in $\mathcal{S}$, which immediately implies that

$$|\mathcal{C}^t| = m_2^t \geqslant m_1 = |\mathcal{S}| \Leftrightarrow t \geqslant \log_{m_2} m_1. \tag{3}$$

As t corresponds to the number of queries to the oracle $\mathcal{O}_{s_B}$, t must be a positive integer. Therefore, the lower bound of the number of queries for recovering an *entire* secret key t_{bounds} in our attack is

$$t_{\mathsf{bounds}} = \left\lceil \log_{m_2} m_1 \right\rceil. \tag{4}$$

The challenge is to find values $k_1, ..., k_{t_{\mathsf{bounds}}}$ such that the resulting codewords determine the absolute values *uniquely* with very high probability. We explain concrete values in Sections 4 and 5.

Taking the absolute value recovery of DXL-KE as an example again with $m_1 = 16$ and $m_2 = 2$, it holds that $t_{\mathsf{bounds}} = \left\lceil \log_{m_2} m_1 \right\rceil = \lceil \log_2 16 \rceil = 4$. Since the best signal leakage attack against DXL-KE needs 1,266 queries [7], there is a large gap between the theoretical lower bound and existing state-of-art results, indicating that there may be more efficient signal leakage attacks.

3.2 Description of our Targeted Signal Extraction Method

In this section, we introduce a generic method to improve the signal leakage attacks, dubbed the *targeted signal extraction* method.

As stated before, there exists a surjective map $\mathcal{M}$, mapping any codeword $\Omega_i = (\omega_{B_1}[i], \omega_{B_2}[i], \cdots, \omega_{B_t}[i])$ obtained by the response of the oracle, to an element in $\mathcal{S}$. Our key observation is that if there exists some component $\omega_{B_j}[i]$ that falls into some fluctuated region, then it can be either 0 or 1 due to the randomness of $\mathbf{g}_B$, which means that it contributes very little to determine $|\mathbf{s}_B[i]|$. It is important to note that this does not necessarily mean we can remove this j-th query directly, since it may help determine other elements in $\mathcal{S}$. However, the more queries in the attack help determine all the elements in $\mathcal{S}$, the smaller the number of queries. The above observation inspires us to improve the known signal leakage attacks by carefully selecting $k_1, ..., k_t \in [0, ..., q-1]$ such that as many $\omega_{B_j}[i]$ (for $j = 1, ..., t$ and $i = 0, ..., n-1$) as possible fall into stable regions. For example, in Figure 2, all signals corresponding to $\mathbf{P}_A \in \{k_1, k_2, k_3\}$ fall into stable regions.

In more detail, for $\mathbf{P}_A = k_1$, the signal corresponding to $|\mathbf{s}_B[i]| = 1$ is in the stable region of 0s, while both $|\mathbf{s}_B[i]| = 2$ and $|\mathbf{s}_B[i]| = 3$ correspond to the stable regions of 1s. Likewise, for $\mathbf{P}_A = k_2$, the signals corresponding to $|\mathbf{s}_B[i]| = 1, 2, 3$ are $(1, 1, 0)$, respectively. Similarly, the signals corresponding to $\mathbf{P}_A = k_3$ are $(1, 0, 1)$, respectively. In this way, the codewords $\Omega_1 = (0, 1, 1)$, $\Omega_2 = (1, 1, 0)$ and $\Omega_3 = (1, 0, 1)$ uniquely determine $|\mathbf{s}_B[i]| = 1, 2$ and 3.

Hence, designing a signal leakage attack with fewer queries can be reduced to finding a sequence of k_j $(j = 1, \cdots, t)$ with small t such that the corresponding codewords Ω_i's satisfy two conditions: every Ω_i determines an element in $\mathcal{S}$ uniquely and all the components of every Ω_i are in as many stable regions as possible. Next we present a heuristic way to find such k_j's.

Finding codewords. We first associate every $\mathbf{s}_B[i] \in \mathcal{S}$ with a unique codeword such that the length of the codeword approaches our lower bound as closely as possible. For example, in DXL-KE, since $\mathcal{C} = \{0, 1\}$, we use the strategy of dichotomy to assign codewords uniquely and of minimal length, see Table 1 in Section 4.[2]

Finding values $k_1, ..., k_t$. Next, we find the appropriate values for k which results in the signal sequence we select above. To achieve this goal, we need to calculate the stable regions of each symbol in $\mathcal{S}$, and determine a set of inequalities related to k whose solution defines the range of k. The selection of k is not unique, and for each targeted signal, we simply select one of them. In case there is no solution for the set of inequalities, we go to Step 1 to select another target signal sequence and then compute the corresponding k.

4 Improved Signal Leakage Attacks on DXL-KE

In this section, we show how to apply the targeted signal extraction to improve signal leakage attacks such that very few queries to the oracle $\mathcal{O}_{\mathbf{s}_B}$ are needed to reveal the entire secret key. The process of the improved attacks consist of two steps: First, each $|\mathbf{s}_B[i]|$ is recovered using targeted signal extraction. Secondly, each $|\mathbf{s}_B[i] + \mathbf{s}_B[i+1]|$ is recovered using targeted signal extraction and our *Weighted Sign Recovery* method.

[2] Interestingly, assigning corresponding binary values as codewords fails because we fail to find suitable values in the next step.

Table 1: Signals $\omega_{B_j}[i]$ for k_j and $|\mathbf{s}_B[i]|$ in DXL-KE with $i \in [0, n-1]$

$\lvert\mathbf{s}_B[i]\rvert$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$k_1 = 550$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
$k_2 = 1,050$	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
$k_3 = 4,000$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
$k_4 = 8,192$	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

Recovering the Absolute Value of $\mathbf{s}_B[i]$. We choose the alphabet of $|\mathbf{s}_B[i]|$ to be $\mathcal{S} = \{0, 1, \cdots, 15\}$ with $m_1 = 16$, see Section 3.1. We recall that the core idea is to determine the symbols in $\mathcal{S}$ by collecting codes $\mathcal{C}^t = \{0, 1\}^t$ of fixed length t. Moreover, according to the previous section we know $t \geq \lceil \log_{m_2} m_1 \rceil = 4$. Our strategy is to use dichotomic search to identify four values, namely $k_1 = 550$, $k_2 = 1,050$, $k_3 = 4,000$, and $k_4 = 8,192$, whose corresponding codes *uniquely* identify $|\mathbf{s}_B[i]|$ for every $i = 0, ..., n-1$. The columns in Table 1 show the corresponding codewords of length 4 for each absolute value. For example, if we collect the the codeword $(0, 0, 1, 1)$, we know that $|\mathbf{s}_B[i]| = 3$ with very high probability. We give the details on how to choose appropriate k_1, k_2, k_3, and k_4 in Appendix A. It is important to note that the choice is not unique.

Recovering the Sign of $\mathbf{s}_B[i]$. In this step, the adversary queries the oracle $\mathcal{O}_{\mathbf{s}_B}$ with different $\mathbf{P}_A = (1 + x)k$ to recover each $|\mathbf{s}_B[i] + \mathbf{s}_B[i+1]|$. The corresponding alphabet is $\mathcal{S} = \{0, 1, \cdots, 30\}$ and $m_1 = 31$, thus $t \geq \lceil \log_{m_2} m_1 \rceil = 5$.

We again identify values for $k_1, ..., k_5$ such that the corresponding codewords uniquely determine the absolute values. We explain our choice of the k_j's in Appendix A and present Table 3 which shows the resulting codewords for our selected values $k_1 = 260$, $k_2 = 525$, $k_3 = 1,050$, $k_4 = 4,000$, and $k_5 = 8,192$.

In order to recover each $|\mathbf{s}_B[i] + \mathbf{s}_B[i+1]|$, the adversary queries $\mathbf{P}_A = (1+x)k_j$ to the oracle to collect the corresponding signal ω_{B_j}. Next, the adversary combines each $\omega_{B_j}[i]$ to get the codeword corresponding $|\mathbf{s}_B[i] + \mathbf{s}_B[i+1]|$ and determines its value according to Table 3. Then the relative sign of $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$ can be determined as described in Section 2. Finally, the adversary needs to repeat this step with $(1 + x^{z+1})k$ to recover the relative sign of two non-zero coefficients separated by z consecutive zeros.

Query Complexity. During absolute value recovery and since $m_1 = 16$, we need $\lceil \log_2 16 \rceil = 4$ queries per coefficient of $\mathbf{s}_B$. During sign recovery, $m_1 = 31$, we need $\lceil \log_2 31 \rceil = 5$ queries to recover the complete secret key. Since $z \approx 4$, this step needs $(1 + z) \lceil \log_2 31 \rceil = 25$ (expected) queries. Therefore, our targeted signal attack needs $\lceil \log_2 16 \rceil + (1 + z) \lceil \log_2 31 \rceil$=29 queries per coefficient of $\mathbf{s}_B$.

Success Probability of Attacking DXL-KE. In DXL-KE, all coefficients of the secret key $\mathbf{s}_B$ and errors $\mathbf{g}_B$ are sampled from discrete Gaussian distribution.

Thus, the probability that $\mathbf{s}_B[i]$ or $\mathbf{g}_B[i]$ greater than h with $\alpha = 3.197$ is

$$\Pr[\mathbf{s}_B[i] > h] = \sum_{x=h+1}^{\infty} \frac{1}{\sqrt{2\pi\alpha^2}} e^{\left(\frac{-x^2}{2\alpha^2}\right)} = \frac{1}{2}\mathrm{erfc}\left(\frac{h}{\sqrt{2}\alpha}\right) \approx 1.35 \cdot 10^{-6}, \tag{5}$$

for DXL-KE and with $\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^{\infty} \exp\left(-t^2\right) dt$ being the complementary error function [12]. For the targeted signal attack to be successful, we need that all n coefficients of $\mathbf{s}_B$ lie in $[-15, 15]$. Since we can always select the appropriate $\mathbf{P}_A = k$ near the middle of the range of k, we can ignore the influence of the errors $\mathbf{g}_B$ on the success probability. Based on Equation (5), the probability that there exists some coefficient of $\mathbf{s}_B$ out of $[-15, 15]$ is at most

$$P_{failure1} = n\left(\Pr[|\mathbf{s}_B[i]| > h]\right) \approx 0.002772.$$

Similarly, we can get $\Pr[\mathbf{s}_B[i] + \mathbf{s}_B[i+1] > 2h] = \frac{1}{2}\mathrm{erfc}\left(\frac{2h}{\sqrt{2}\alpha}\right) \approx 3.182 \cdot 10^{-21}$. And the failure probability of recovering the sign of $\mathbf{s}_B[i]$ as $P_{failure2} = (1 + z)\lceil \log_2 30 \rceil n \Pr[|\mathbf{s}_B[i] + \mathbf{s}_B[i+1]| > 2h] \approx 6.5166 \cdot 10^{-18}$, which can be ignored. Hence, the probability of recovering a complete secret key $\mathbf{s}_B$ for our targeted signal attack is $P_{\text{success}} \approx (1 - P_{\text{failure}_1} - P_{\text{failure}_2}) \times 100\% = 99.7228\%$.

5 Our Targeted Signal Leakage Attack on KEs and AKEs

In this section, we apply our targeted signal extraction and *Weighted Sign Recovery* methods to give an improved signal leakage attack against DBS-KE [17], and then we show that our attack can also be migrated directly to LBA-PAKE [13] and Quantum2FA [39].

5.1 Improved Attack Against DBS-KE

Description of DBS-KE. The DBS-KE [17] proposed by Ding, Branco, and Schmitt is presented in Fig. 4. $H_1 : \{0, 1\} \to \chi_\alpha$ is a hash function whose outputs are sampled from the discrete Gaussian distribution χ_α. DBS-KE is designed to provide robustness for key reuse using the *pasteurization* technique. The key point of this technique is that Bob does not use Alice's public key $\mathbf{P}_A$ to multiply his private key directly, but transforms $\mathbf{P}_A$ to $\overline{\mathbf{P}}_A$ as

$$\overline{\mathbf{P}}_A = \mathbf{P}_A + \mathbf{a}H_1(id_A, id_B, \mathbf{P}_A) + 2\mathbf{g}_B. \tag{6}$$

DBS-KE is instantiated with $\alpha = 4.19, n = 512$, and $q = 26,038,273$.

Unfortunately, Bindel, Stebila and Veitch show that the scheme is in fact not robust against key re-use [7]. In their proposed attack, the adversary selects $\mathbf{P}_A = k$ for some $k \in [0, q-1]$. Upon input $\mathbf{P}_A$, the oracle computes

$$\begin{aligned} \mathbf{K}_B &= \mathbf{P}_A\mathbf{s}_B + (\mathbf{P}_A\mathbf{d} + \mathbf{c}\mathbf{P}_B + \mathbf{acd}) + (2\mathbf{g}_B\mathbf{s}_B + 2\mathbf{g}_B\mathbf{d} + 2\mathbf{g}'_B - 2\mathbf{c}\mathbf{e}_B) \\ &= \mathbf{P}_A\mathbf{s}_B + \mathit{\Delta} + \varepsilon, \end{aligned} \tag{7}$$

Alice		**Bob**
$\mathbf{s}_A, \mathbf{e}_A \leftarrow \chi_\alpha$		
$\mathbf{P}_A = \mathbf{a}\mathbf{s}_A + 2\mathbf{e}_A$	$\xrightarrow{\mathbf{P}_A}$	$\mathbf{s}_B, \mathbf{e}_B, \mathbf{g}_B, \mathbf{g}'_B \leftarrow \chi_\alpha$
$\mathbf{c} \leftarrow H_1(id_A, id_B, \mathbf{P}_A)$		$\mathbf{P}_B = \mathbf{a}\mathbf{s}_B + 2\mathbf{e}_B$
		$\mathbf{c} \leftarrow H_1(id_A, id_B, \mathbf{P}_A)$
		$\mathbf{d} \leftarrow H_1(id_A, id_B, \mathbf{P}_A, \mathbf{P}_B)$
$\mathbf{d} \leftarrow H_1(id_A, id_B, \mathbf{P}_A, \mathbf{P}_B)$		$\overline{\mathbf{P}_A} \leftarrow \mathbf{P}_A + \mathbf{a}\mathbf{c} + 2\mathbf{g}_B$
$\mathbf{g}_A, \mathbf{g}'_A \leftarrow \chi_\alpha$		$\mathbf{K}_B = \overline{\mathbf{P}_A}(\mathbf{s}_B + \mathbf{d}) + 2\mathbf{g}'_B$
$\overline{\mathbf{P}_B} \leftarrow \mathbf{P}_B + \mathbf{a}\mathbf{c} + 2\mathbf{g}_A$	$\xleftarrow{(\mathbf{P}_B, \omega_B)}$	$\omega_B = Sig(\mathbf{K}_B) \in \{0,1\}^n$
$\mathbf{K}_A = \overline{\mathbf{P}_B}(\mathbf{s}_A + \mathbf{c}) + 2\mathbf{g}'_A$		
$SK_A \leftarrow Mod_2(\mathbf{K}_A, \omega_B) \in \{0,1\}^n$		$SK_B \leftarrow Mod_2(\mathbf{K}_B, \omega_B) \in \{0,1\}^n$

Fig. 4: Pseudo-code description of DBS-KE

with $\mathit{\Delta} = \mathbf{P}_A\mathbf{d} + \mathbf{c}\mathbf{P}_B + \mathbf{acd}$ and $\varepsilon = 2\mathbf{g}_B\mathbf{s}_B + 2\mathbf{g}_B\mathbf{d} + 2\mathbf{g}'_B - 2\mathbf{c}\mathbf{e}_B$.

The adversary knows $\mathbf{a}$, $\mathbf{P}_A$, $\mathbf{P}_B$, hence, $\mathit{\Delta}$. That means in particular, that the adversary can choose id_A and $\mathbf{P}_A$ such that $\mathit{\Delta}[i] = 0$, i.e., $\mathbf{K}_B[i] = k\mathbf{s}_B[i] + \varepsilon[i]$. Furthermore, the adversary can calculate a bound for $\varepsilon[i] = (2\mathbf{g}_B\mathbf{s}_B + 2\mathbf{g}_B\mathbf{d} + 2\mathbf{g}'_B - 2\mathbf{c}\mathbf{e}_B)[i]$, since all of these terms are sampled from discrete Gaussian distribution with standard deviation α. This circumvents the pasteurization and allows for the signal leakage attack.

Our Improved Signal Leakage Attack on DBS-KE. We improve the sparse signal attack against DBS-KE further by using our targeted signal extraction, reducing the number of queries drastically. For DBS-KE, the oracle $\mathcal{O}_{\mathbf{s}_B}$ computes $\mathbf{K}_B[i]$ as $\mathbf{P}_A\mathbf{s}_B[i] + \mathit{\Delta}[i] + \varepsilon[i]$. We regard $\mathit{\Delta}[i] + \varepsilon[i]$ as the cause of the fluctuated region just like the errors $2\mathbf{g}_B[i]$ do in DXL-KE. As such, $\mathbf{K}_B[i]$ has the same form in DBS-KE and DXL-KE. Therefore, we can use our targeted signal extraction to launch an improved attack against DBS-KE. We present the details on how to choose k_j and the bounds on $\mathit{\Delta}$ in Appendix C.

For recovering $|\mathbf{s}_B[i]|$, the same targeted signals as in Table 1 can be used since $|\mathbf{s}_B[i]| \in [0, 15]$. However, the values for k will be different, namely $k_1 =$ 868,000, $k_2 =$ 1,735,800, $k_3 =$ 6,076,000, and $k_4 =$ 13,019,136. We let $\mathit{\Delta}_j[i]$ where $j = 1, 2, 3, 4$ denote the term $\mathit{\Delta}[i]$ corresponding to each k_j. Difficulties during signal collection might occur if $\mathit{\Delta}[i]$ is large (while $\mathbf{g}_B[i]$ is small with high probability), and hence, disturb the signal. Since the adversary is able to calculate the value of $\mathit{\Delta}[i]$, they are able to only collect signals where the corresponding $\mathit{\Delta}_j[i]$ is small enough. More concretely, signals are only collected when $|\mathit{\Delta}_j[i]| \leqslant$ 426,000 ($j = 1, 2, 3$), and $|\mathit{\Delta}_4[i]| \leqslant$ 6,500,000. It is important to note that this requirement is much less restrictive than the one in the sparse signal attack [7], where $\mathit{\Delta}[i]$ need to be exactly 0. We compute the probability of $\mathit{\Delta}_j[i]$ being sufficiently small in Section 5.1. The adversary will keep querying the oracle until enough signals $\omega_{B_j}[i]$ to determine the absolute values are collected. Afterwards, $\mathit{\Omega}_i = (\omega_{B_1}[i], \omega_{B_2}[i], \omega_{B_3}[i], \omega_{B_4}[i])$ are taken and used to recover $|\mathbf{s}_B[i]|$ according to Table 1.

To determine the sign of $\mathbf{s}_B[i]$, we also use targeted signal extraction with $\mathcal{S} = \{+, -\}$ and the corresponding codewords being 1 and 0. The adversary first finds a small enough k to make its corresponding signal $Sig(k|\mathbf{s}_B[i]|) = 0$. Then, they need to find a $\Delta[i]$ which is approximately equal to $\frac{q}{4}$. If $\mathbf{s}_B[i] < 0$, $k\mathbf{s}_B[i] + \Delta[i] < \lfloor \frac{q}{4} \rceil - |\varepsilon[i]|$, and the corresponding signal $Sig(k\mathbf{s}_B[i] + \Delta[i]) = 0$. Otherwise (i.e., if $\mathbf{s}_B[i] > 0$), $k\mathbf{s}_B[i] + \Delta[i] > \lfloor \frac{q}{4} \rceil + |\varepsilon[i]|$, and $Sig(k\mathbf{s}_B[i] + \Delta[i]) = 1$. That is, the positive $\mathbf{s}_B[i]$ corresponds to the signal $\omega_B[i] = 1$, while $\omega_B[i] = 0$ represents the negative $\mathbf{s}_B[i]$. Specifically, the adversary selects the parameter $\mathbf{P}_A = k = 813,000$ to access the oracle, when $5,710,000 \leqslant \Delta[i] \leqslant 7,310,000$, and collects the corresponding signal $\omega_B[i]$. This process is repeated for every $\omega_B[i]$ ($i \in [0, n-1]$). If $\omega_B[i] = 0$, the adversary determines that $\mathbf{s}_B[i]$ is negative, otherwise $\mathbf{s}_B[i]$ is positive.

Query Complexity. In our improved attack on DBS-KE it is clear that the number of queries is related to the range of our bound $\Delta[i]$. More concretely, the larger the range of the bound $\Delta[i]$ is, the more signals the adversary gets after each query, and thus fewer queries are required to complete the attack. We use b to denote the bound on $|\Delta[i]|$. Since the distribution of $\Delta[i]$ is close to uniformly random, the probability of $|\Delta[i]| \leqslant b$ is approximately $2b/q$. Consequently, in our improved attack, the adversary approximately collects $2nb/q$ signals after the first query, while $n(1 - 2b/q)$ signals remain to be collected. Let t denote the number of queries. After t queries, there are still $n(1 - 2b/q)^t$ signals left to be collected by the adversary. Therefore, the number of signals that the adversary has collected after t queries is

$$n - \left\lfloor n \left(1 - \frac{2b}{q}\right)^t \right\rceil, \tag{8}$$

where the collected signals are all integers in practice. When the result of Equation (8) is n, the adversary stops collecting signals, which means

$$\left\lfloor n \left(1 - \frac{2b}{q}\right)^t \right\rceil = 0. \tag{9}$$

Or equivalently,

$$n \left(1 - \frac{2b}{q}\right)^t < \frac{1}{2}. \tag{10}$$

That is,

$$t > \log_{\frac{q-2b}{q}} \frac{1}{2n}. \tag{11}$$

According to Equation (11), we directly take the minimum value of t, namely

$$t = \lceil \log_{\frac{q-2b}{q}} \frac{1}{2n} \rceil. \tag{12}$$

For the improved attack against DBS-KE, with $q = 26,038,273$, $n = 512$, and the bounds $b_1 = b_2 = b_3 = 426,000$ and $b_4 = 6,500,000$ for the absolute value recovery, we can compute the respective number of queries $t_1 = t_2 = t_3 = \lceil 208.35 \rceil = 209$, and $t_4 = \lceil 10.02 \rceil = 11$. Thus, the total number of required queries in Step 1 is 638. In the second step (i.e., sign recovery), $\Delta_j[i]$ should be in the range of $[5\,710\,000, 7\,310\,000]$ over $\mathbb{Z}_q$, thus $b = 1,600,000$. Similarly, we can get the needed queries t as

$$t = \left\lceil \log_{\frac{q-b}{q}} \frac{1}{2n} \right\rceil = \lceil 109.30 \rceil = 110. \tag{13}$$

Therefore, the total number of needed queries to recover the key of DBS-KE is 748.

Success Probability. Recall that in our improved signal leakage attack against DBS-KE, the failure probability to recover the secret key only depends on its bound. More precisely, based on Equation (5), the failure probability of recovering all $\mathbf{s}_B[i] \in [-15, 15]$ is related to the fixed bound $h = 15$ of $\mathbf{s}_B$ and α.

In case of DBS-KE with $\alpha = 4.19$, the failure probability is $P_{\text{failure}_1} \approx 0.1760$. Hence, the success probability of our improved signal leakage attack against DBS-KE is $P_{\text{success}} = (1 - P_{\text{failure}_1}) \times 100\% \approx 82.40\%$.

5.2 Application to DBS-AKE

DBS-AKE is an AKE based on DBS-KE using a similar pasteurization technique. Bindel, Stebila, and Veitch [7] extended their attack against DBS-KE to DBS-AKE under the extended Canetti-Krawczyk (eCK) model [11]. Since DBS-AKE also uses the *pasteurization* technique, they analyze the components of $\mathbf{K}_B$ in DBS-AKE, which can be formalized as $\mathbf{K}_B = \mathbf{y}_A \mathbf{s}_B + \Delta + \varepsilon$. Here $\mathbf{y}_A$ is an ephemeral public key of Alice. Similar to DBS-KE, the value of Δ is approximately uniform over R_q, and ε follows a discrete Gaussian distribution. In the eCK model, the adversary is able to calculate the value of Δ, which can be exploited similarly to the attack against DBS-KE to recover the long-term key $\mathbf{s}_B$ in DBS-AKE.

Similar to the result in Section 5.1, our attack can be applied to DBS-AKE in the eCK model. Compared to the *sparse signal collection*, our targeted signal extraction requires much fewer signals. Specifically, DBS-KE and DBS-AKE share the same parameters, thus the needed queries against DBS-AKE are almost the same as that against DBS-KE, which is 745. However, note that in the BR security model, the adversary does not have the ability to obtain the value of Δ, hence DBS-AKE can resist the various signal leakage attacks above in accordance with the BR model.

5.3 Improved Attack Against LBA-PAKE

Description of LBA-PAKE. LBA-PAKE [13] is a password-based authenticated key exchange, which integrates the conventional password authentication to the RLWE-based key exchange. In LBA-PAKE, Bob stores the hash value of Alice's password and id_A. When Alice initiates a key exchange with Bob using her password, id_A, and public key $\mathbf{P}_A$. Bob first checks that the hash over $\mathbf{P}_A$ is the same as the stored value, and computes

$$\overline{\mathbf{P}}_A = \mathbf{P}_A + \mathbf{a}H_1(\mathbf{P}_A). \tag{14}$$

The transformation from $\mathbf{P}_A$ to $\overline{\mathbf{P}}_A$ can be seen as a simplified variant of *pasteurization*, which uses only $\mathbf{P}_A$ as the input to H_1, but without the employment of identity id_A, id_B, and the errors. Then, Bob computes a temporary public key $\mathbf{y}_B = \mathbf{a}\mathbf{r}_B + 2\mathbf{g}_B$, with $\mathbf{r}_B, \mathbf{g}_B \leftarrow \chi_\alpha$. He then uses his long-term secret key $\mathbf{s}_B$ to compute $\mathbf{K}_B$ as

$$\mathbf{K}_B = \mathbf{P}_A\mathbf{s}_B + (\mathbf{P}_A\mathbf{d} + \mathbf{c}\mathbf{P}_B + \mathbf{acd}) + (2\mathbf{g}'_B - 2\mathbf{c}\mathbf{e}_B), \tag{15}$$

where $\mathbf{g}'_B, \mathbf{e}_B \leftarrow \chi_\alpha$, $\mathbf{c} = H_1(\mathbf{P}_A)$, $\mathbf{d} = H_1(\mathbf{y}_B)$. Finally, Bob computes the signal $\omega_B = Sig(\mathbf{K}_B)$, and sends $\mathbf{y}_B$ and ω_B to Alice. The protocol is claimed to be secure in the Real-or-Random model that has been introduced by Abdalla, Fouque, and Pointcheval [1]. As we show, this claim is unfortunately not true. More concretely, every registered user with a password (i.e., after an honest registration phase) can recover the server's long-term key by launching signal leakage attacks. The following instantiation is proposed $n \in \{512, 256, 128\}$, $q = 7,557,773$, $\alpha = 3.192$.

Our Signal Leakage Attack on LBA-PAKE. The designers of LBA-PAKE claimed that LBA-PAKE is secure and robust for reusing the long-term key $\mathbf{s}_B$. However, we discover that any registered user could recover the long-term key under the key reuse setting. As a registered user, an adversary can pass the verification of the server Bob using their own password and identity. Then they are able to launch the key exchange with the server/Bob. Similar to the case in DBS-KE, the adversaries are able to calculate $\Delta = \mathbf{P}_A d + \mathbf{c}\mathbf{P}_B + \mathbf{acd}$, since they know the long-term public key $\mathbf{P}_B$, and receive $\mathbf{y}_B$ from Bob. As before, Δ is close to uniformly distributed over R_q, and the error term $\varepsilon = 2\mathbf{g}'_B - 2\mathbf{c}\mathbf{e}_B$ follows a discrete Gaussian distribution. Based on the above discussion, it is easy to see that targeted signal extraction against DBS-KE can also be directly applied to LBA-PAKE. Therefore, the adversary performs the same operations as in Section 5.1 with the following attack parameters.

For absolute value recovery, the adversary queries $\mathbf{P}_A = k_i$, with $k_1 = 252,000$, $k_2 = 503,800$, $k_3 = 1,764,000$, and $k_4 = 3,778,886$. The corresponding $|\Delta_j[i]|$ is bounded as $|\Delta_j[i]| \leqslant 122,000$ when $j \in [1,3]$, and $|\Delta_4[i]| \leqslant 1,887,000$. For sign recovery, the adversary only queries $\mathbf{P}_A = 236,000$, and bounds $\Delta[i]$ as $1,656,000 \leqslant \Delta[i] \leqslant 2,120,000$.

Following [7, Section 5.3], we calculate the standard deviation of $\varepsilon[i]$ to be $\sqrt{4n\alpha^2 + 4\alpha^2}$. Since $4.5\sqrt{4n\alpha^2 + 4\alpha^2} \approx 2075.13$, we assume that $|\varepsilon[i]| \leq 2100$. We give details on how to choose the values as described above in Appendix C.

Query Complexity. Following Section 5.1 closely, in the attack against LBA-PAKE, we choose the bounds $b_1 = b_2 = b_3 = 122,000$ and $b_4 = 1,887,000$, during absolute value recovery. Hence, the total number of needed queries is $t_1 + t_2 + t_3 + t_4 = 647$. Similarly, the number of required queries t in Step 2 with $\Delta_j[i]$'s bound $b = 464,000$, is $t = 110$. Thus, the total number of queries for our attack against LBA-PAKE is 757.

Success Probability. Similar to the case of DBS-KE, we can write $h \approx 4.6992\alpha$, and the failure probability is $P_{\text{failure}_1} \approx 0.0013$. Therefore, the success probability against LBA-PAKE is $P_{\text{success}} = (1 - P_{\text{failure}_1}) \times 100\% \approx 99.87\%$.

5.4 Application to Quantum2FA

Quantum2FA [39] is a password-based authentication that uses a modified version of NewHope-Simple [3] to establish shared keys. Specifically, Quantum2FA is instantiated with $q = 12289$, $n = 512$. The secret $\mathbf{s}$ and error $\mathbf{e}$ are sampled from the centered binomial distribution ψ_8, i.e., they are integers in [-8, 8].

In Quantum2FA, the server A computes the long-term public key $\mathbf{P}_A = \mathbf{a}\mathbf{s}_A + \mathbf{e}_A$, where $\mathbf{s}_A, \mathbf{e}_A \leftarrow \psi_8$. It is important to point out that A stores $\mathbf{P}_A$ in a smart card B in advance. When B is used to log into the server to complete the password-based authenticated key exchange, B samples the ephemeral secret $\mathbf{s}_B$ to compute $\mathbf{P}_B = \mathbf{a}\mathbf{s}_B + \mathbf{e}_B$, where $\mathbf{s}_B, \mathbf{e}_B \leftarrow \psi_8$. Then B chooses a random key m to compute $\mathbf{c} = \mathbf{P}_A\mathbf{s}_B + \mathbf{e}'_B + \mathsf{Encode}(m)$, where $m \leftarrow \{0,1\}^{128}$, $\mathbf{e}'_B \leftarrow \psi_8$, and $\mathsf{Encode}(m)$ is a polynomial $\mathbf{f}$ with $\mathbf{f}[i + j \cdot 128] = \lfloor q/2 \rfloor \cdot m[i]$ for $i \in \{0, \ldots, 127\}$ and $j = 0, 1, 2, 3$. After that, B computes $\bar{\mathbf{c}} = \mathsf{Compress}(\mathbf{c})$, where $\bar{\mathbf{c}}[i] = \lfloor (\mathbf{c}[i] \cdot 8)/q \rceil \bmod 8$ and sends $\mathbf{P}_B, \bar{\mathbf{c}}$ to server A.

To thwart the signal leakage attack in Quantum2FA, the server A needs to pre-embed the public key $\mathbf{P}_A$ into B, which means that even a malicious A cannot deliberately select more than one $\mathbf{P}_A$ to launch attacks. The question is whether it is possible to launch the attack with only one query.

Since the signal $\bar{\mathbf{c}}[i] \in [0, 7]$ and $\mathbf{s}_B[i] \in [-8, 8]$, from Equation (4) we need $t_{\mathsf{bounds}} = \lceil \log_8 17 \rceil = 2$ queries to fully recover the secret. However, by restricting $\mathbf{s}_B[i] \in [-1, 1]$, we can successfully recover part of $\mathbf{s}_B$ with one query. Specifically, we assume that server A is malicious and launches the following attack.

Step 1. A chooses $\mathbf{P}_A = 1260$[3] and embeds it into the smart card B in advance.
Step 2. After receiving the authentication information and the signal $\bar{\mathbf{c}}$ sent from B, A checks whether $\bar{\mathbf{c}}[i]$ is equal to the targeted signal. Specifically, A determines that $\mathbf{s}_B[i] = 0$ if $\bar{\mathbf{c}}[i] \in \{0, 4\}$, $\mathbf{s}_B[i] = 1$ if $\bar{\mathbf{c}}[i] \in \{1, 5\}$, and $\mathbf{s}_B[i] = -1$ if $\bar{\mathbf{c}}[i] \in \{3, 7\}$.

According to the distribution of ψ_8, the probability that $\mathbf{s}_B[i] \in [-1, 1]$ is 54.55%. Hence, A can recover about 1/2 of all coefficients of $\mathbf{s}_B$. Although this is not a complete key recovery, it decreases the bit security drastically.

[3] Other values than 1260 are possible but at this time, our attack needs $\mathbf{P}_A$ to be a constant polynomial

Table 2: Comparison of the experimental results on DXL-KE and DBS-KE

Protocols	Attacks	n	α	q	**Average #Queries**	**Average Time (s)**
DXL-KE	Sparse Signal Attack [7]	1024	3.197	$2^{14}+1$	824.13	24.14
	Ours				24.23	1.04
DBS-KE	Sparse Signal Attack [7]	512	4.19	$\leq 2^{7.5}$	390,597.15	582.08
	Ours				737.45	6.53
LBA-PAKE	Ours	512	3.192	$\leq 2^{6.9}$	742.53	7.67

6 Experimental Evaluation

In this section, we perform experimental validation of our improved attacks against the above mentioned (authenticated) key exchange protocols. Furthermore, we compare the results of our proposed attacks on DXL-KE and DBS-KE with the sparse signal attack [7] in terms of average queries and time, which shows that our attacks are more efficient. Our implementations are publicly available on `https://github.com/frostry/improved-signal-leakage-attack`.

6.1 Experimental Setup

We implement our proposed improved attacks against DXL-KE and DBS-KE on the basis of the publicly available implementation of the sparse signal attack [38,7]. It is important to note that the implementation of the sparse signal attack is designed to recover $\mathbf{s}_B[i] \in [-13, 13]$, while our attack is designed for the range $[-15, 15]$. Thus, for consistency, we modify the sparse signal parameters for attacking DBS-KE in their implementation by reducing the limit h_2 (resp., h'_2) for Δ from 220,000 (resp., 110,000) to $210,000$ (resp., $100,000$). Moreover, we set the sampling parameter t_1 (resp., t_2) from $465,000$ (resp., $230,000$) to $434,000$ (resp., $220,000$). Moreover, in the implementation of [38], two polynomial multiplication functions, namely `poly_mul_mont` and `poly_mul`, are implemented. In our experiments, we use `poly_mul` as it is experimentally faster in our setting. Furthermore, we follow [38] in collecting signals during the attack in parallel to ensure a fair comparison. In addition, we also simulate the attack against Quantum2FA by implementing the part of authenticated key exchange.

All implementations are run on a computer with two 3 GHz Intel Xeon E5-2620 CPUs and a 64 GB RAM. We run each attack 1000 times, and record the average queries and the average time. For each attack, we generate a unique secret key. We present the procedures of our improved attacks against DXL-KE and DBS-KE in Algorithm 1 and 2, respectively.

6.2 Results and Comparison

The experimental results of our proposed attacks in comparison with our re-run of the sparse signal attacks are presented in Table 2. As shown in the table, our

Algorithm 1 Pseudocode of our attack against DXL-KE

Input: $\mathbf{P}_A$
Output: $\mathbf{s}_B$
$k = [550, 1050, 4000, 8192]$
Set $n = 512$, $queries = 0$, $c = 0$
for i from 0 to 3 **do**
  Set $\mathbf{P}_A = k[i]$
  $(\mathbf{P}_B, \omega_{B_i}) \leftarrow \mathcal{O}(\mathbf{P}_A)$
end for
for i from 0 to $n-1$ **do**
  for j from 0 to 3 **do**
    Extract W_i from $\omega_{B_j}[i]$
  end for
  Recover each $|\mathbf{s}_B[i]|$ from W_i according to Table 1
end for
Set $z =$ the maximum number of consecutive 0s
$k = [260, 525, 1050, 4000, 8192]$
for c from 1 to $z+1$ **do**
  for i from 0 to 4 **do**
    Set $\mathbf{P}_A = k[i](x+c)$
    $(\mathbf{P}_B, \omega_{B_i}) \leftarrow \mathcal{O}(\mathbf{P}_A)$
  end for
  for i from 0 to $n-1$ **do**
    for j from 0 to 4 **do**
      Extract W_i from $\omega_{B_j}[i]$
    end for
    Recover each $|\mathbf{s}_B[i] + \mathbf{s}_B[i+c]|$ from W_i according to Table 3
  end for
end for
Recover the relative sign of $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$
return $\mathbf{s}_B$ or $-\mathbf{s}_B$

improved attacks using targeted signal and weighted sign recovery reduce the number of queries by 97.1% and 97.7% (about 33 and 42 times), and reduce the run time by 95.7% and 98.2% (about 22 and 54 times), respectively.

For attacks against DBS-KE, our improved attack significantly reduces the queries and time by 99.8% and 98.9%, respectively, which means our attack is nearly 100 times more efficient than the sparse signal attack [7]. Additionally, our attack against Quantum2FA successfully recovers 54.57% of 512 coefficients in each session on average.

7 Conclusion

In this paper, we have proposed a systematic method to determine the lower bounds for the number of queries required in the signal leakage attacks against LWE-based key exchange. Further, we follow the lower bounds' guide to propose a generic technique called *Targeted Signal Extraction* and apply it to improve the existing signal leakage attacks. As a result, our improved attacks are capable of reducing the number of queries by tens or even hundreds of times compared to previous attacks. Due to the recent attacks and our improved attacks against LWE-based AKE, as well as the fact that FO transform only help protect one side, we conclude that it is not an easy task to design LWE-based AKE. Therefore, more attention should be paid to the security of AKE under key reuse attacks.

Algorithm 2 Pseudocode of our attack on DBS-KE

```
Input: P_A
Output: s_B
k = [868000, 1735800, 6076000, 13019136]
t = [426000, 426000, 426000, 6500000]
Set n = 512, queries = 0, c = 0
Select id_A and id_B
for i from 0 to 3 do
    Set P_A = k[i]
    while c < n do
        (P_B, ω_{B_i}) ← O(P_A)
        c ← H_1(id_A, id_B, P_A)
        d ← H_1(id_A, id_B, P_A, P_B)
        Δ = P_A d + c P_B + acd
        for j from 0 to n − 1 do
            if |Δ[j]| ⩽ t[i] then
                if ω_{B_i}[j] has not been collected then
                    Collect ω_{B_i}[j]
                    c = c + 1
                end if
            end if
        end for
        Change id_A
    end while
    c = 0
end for
for i from 0 to n − 1 do
    for j from 0 to 3 do
        Extract W_i from ω_{B_j}[i]
    end for
    Recover each |s_B[i]| from W_i according to Table 1
end for
P_A = 813 000
Set t_1 = 5 710 000, t_2 = 7 310 000
while c < n do
    (P_B, ω_B) ← O(P_A)
    c ← H_1(id_A, id_B, P_A)
    d ← H_1(id_A, id_B, P_A, P_B)
    Δ = P_A d + c P_B + acd
    for i from 0 to n − 1 do
        if t1 ⩽ Δ[i] ⩽ t2 then
            if ω_B[i] has not been collected then
                Collect ω_B[j]
                c = c + 1
                if ω_B[i] == 0 then
                    s_B[i] is negative
                else
                    s_B[i] is positive
                end if
            end if
        end if
    end for
    Change id_A
end while
return s_B
```

References

1. Abdalla, M., Fouque, P.A., Pointcheval, D.: Password-based authenticated key exchange in the three-party setting. In: Public Key Cryptography - PKC 2005. pp. 65–84. Springer (2005)
2. Akleylek, S., Seyhan, K.: A probably secure bi-gisis based modified AKE scheme with reusable keys. IEEE Access **8**, 26210–26222 (2020)
3. Alkim, E., Ducas, L., Pöppelmann, T., Schwabe, P.: NewHope without reconciliation. Cryptology ePrint Archive, Report 2016/1157 (2016)
4. Băetu, C., Durak, F.B., Huguenin-Dumittan, L., Talayhan, A., Vaudenay, S.: Misuse Attacks on Post-quantum Cryptosystems. In: EUROCRYPT 2019. pp. 747–776. Springer (2019)
5. Bauer, A., Gilbert, H., Renault, G., Rossi, M.: Assessment of the Key-Reuse Resilience of NewHope. In: CT-RSA 2019. pp. 272–292. Springer (2019)
6. Bellare, M., Rogaway, P.: Entity Authentication and Key Distribution. In: CRYPTO 1993. pp. 232–249. Springer (1994)
7. Bindel, N., Stebila, D., Veitch, S.: Improved attacks against key reuse in learning with errors key exchange. In: LATINCRYPT 2021. LNCS, Springer (2021)
8. Bleichenbacher, D.: Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS# 1. In: CRYPTO 1998. pp. 1–12. Springer (1998)
9. 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: S&P 2015. pp. 553–570. IEEE (2015)
10. Brendel, J., Fiedler, R., Günther, F., Janson, C., Stebila, D.: Post-quantum Asynchronous Deniable Key Exchange and the Signal Handshake. Cryptology ePrint Archive, Report 2021/769 (2021)
11. Canetti, R., Krawczyk, H.: Analysis of Key-Exchange Protocols and Their Use for Building Secure Channels. In: EUROCRYPT 2001. pp. 453–474. Springer (2001)
12. Chang, S.H., Cosman, P.C., Milstein, L.B.: Chernoff-type bounds for the Gaussian error function. IEEE Transactions on Communications **59**(11), 2939–2944 (2011)
13. Dabra, V., Bala, A., Kumari, S.: LBA-PAKE: Lattice-Based Anonymous Password Authenticated Key Exchange for Mobile Devices. IEEE Systems Journal **15**(4), 5067–5077 (2021)
14. Debris-Alazard, T., Ducas, L., van Woerden, W.P.: An algorithmic reduction theory for binary codes: Lll and more. Cryptology ePrint Archive, Report 2020/869 (2020), `https://ia.cr/2020/869`
15. Diffie, W., Hellman, M.: New directions in cryptography. IEEE transactions on Information Theory **22**(6), 644–654 (1976)
16. Ding, J., Alsayigh, S., Saraswathy, R., Fluhrer, S., Lin, X.: Leakage of signal function with reused keys in RLWE key exchange. In: ICC 2017. pp. 1–6. IEEE (2017)
17. Ding, J., Branco, P., Schmitt, K.: Key Exchange and Authenticated Key Exchange with Reusable Keys Based on RLWE Assumption. Cryptology ePrint Archive, Report 2019/665 (2019)
18. Ding, J., Fluhrer, S., Rv, S.: Complete attack on RLWE key exchange with reused keys, without signal leakage. In: Australasian Conference on Information Security and Privacy. pp. 467–486. Springer (2018)
19. Ding, J., Xie, X., Lin, X.: A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem. Cryptology ePrint Archive, Report 2019/688 (2012)

20. Fluhrer, S.R.: Cryptanalysis of ring-LWE based key exchange with key share reuse. Cryptology ePrint Archive, Report 2016/085 (2016)
21. Greuet, A., Montoya, S., Renault, G.: Attack on LAC Key Exchange in Misuse Situation. Cryptology ePrint Archive, Report 2020/063 (2020)
22. Günther, F., Towa, P.: KEMTLS with delayed forward identity protection in (almost) a single round trip. Cryptology ePrint Archive, Report 2021/725 (2021)
23. Hashimoto, K., Katsumata, S., Kwiatkowski, K., Prest, T.: An Efficient and Generic Construction for Signal's Handshake (X3DH): Post-Quantum, State Leakage Secure, and Deniable. Cryptology ePrint Archive, Report 2021/616 (2021)
24. Huguenin-Dumittan, L., Vaudenay, S.: Classical Misuse Attacks on NIST Round 2 PQC - The Power of Rank-Based Schemes. In: ACNS 2020. pp. 208–227. Springer (2020)
25. Krawczyk, H.: HMQV: A High-Performance Secure Diffie-Hellman Protocol. In: Advances in Cryptology – CRYPTO 2005. pp. 546–566. Springer (2005)
26. Matsumoto, T., Takashima, Y., Imai, H.: On seeking smart public-key-distribution systems. IEICE TRANSACTIONS (1976-1990) **69**(2), 99–106 (1986)
27. Menezes, A., Qu, M., Vanstone, S.: Some new key agreement protocols providing implicit authentication. In: Workshop on Selected Areas in Cryptography (SAC 95). pp. 22–32. CRC Press (1995)
28. Menezes, A., Ustaoglu, B.: On reusing ephemeral keys in Diffie-Hellman key agreement protocols. International Journal of Applied Cryptography **2**(2), 154–158 (2010)
29. Okada, S., Wang, Y., Takagi, T.: Improving Key Mismatch Attack on NewHope with Fewer Queries. Cryptology ePrint Archive, Report 2020/585 (2020)
30. Peikert, C.: Lattice cryptography for the internet. In: PQCrypto 2014. pp. 197–219 (2014)
31. Qin, Y., Cheng, C., Ding, J.: A complete and optimized key mismatch attack on NIST candidate NewHope. In: ESORICS 2019. pp. 504–520. Springer (2019)
32. Qin, Y., Cheng, C., Ding, J.: An Efficient Key Mismatch Attack on the NIST Third Round Candidate Kyber. Cryptology ePrint Archive, Report 2019/1343 (2019)
33. Qin, Y., Cheng, C., Zhang, X., Pan, Y., Hu, L., Ding, J.: A systematic approach and analysis of key mismatch attacks on lattice-based nist candidate kems. In: ASIACRYPT 2021. pp. 92–121. Springer (2021)
34. Regev, O.: On Lattices, Learning with Errors, Random Linear Codes, and Cryptography. In: STOC 2005. pp. 84–93. ACM (2005)
35. Schwabe, P., Stebila, D., Wiggers, T.: Post-Quantum TLS Without Handshake Signatures. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 1461–1480 (2020)
36. Schwabe, P., Stebila, D., Wiggers, T.: More efficient post-quantum KEMTLS with pre-distributed public keys. In: ESORICS 2021. LNCS, Springer (October 2021)
37. Seyhan, K., Nguyen, T.N., Akleylek, S., Cengiz, K., Islam, S.H.: Bi-GISIS KE: modified key exchange protocol with reusable keys for IoT security. J. Inf. Secur. Appl. **58**, 102788 (2021)
38. Veitch, S.: Improved key reuse attack implementation, `https://git.uwaterloo.ca/ssveitch/improved-key-reuse` (visited on 05/2021)
39. Wang, Q., Wang, D., Cheng, C., He, D.: Quantum2FA: Efficient Quantum-Resistant Two-Factor Authentication Scheme for Mobile Devices. IEEE Transactions on Dependable and Secure Computing (Early Access, 2021), `https://ieeexplore.ieee.org/document/9623421`
40. Zhang, J., Zhang, Z., Ding, J., Snook, M., Dagdelen, Ö.: Authenticated key exchange from ideal lattices. In: EUROCRYPT 2015. pp. 719–751. Springer (2015)

41. Zhang, X., Cheng, C., Ding, R.: Small Leaks Sink a Great Ship: An Evaluation of Key Reuse Resilience of PQC Third Round Finalist NTRU-HRSS. In: ICICS 2021. pp. 283–300 (2021)

A Parameter Choices in the Improved Attack Against DXL-KE

In this section, we provide details of choosing k to match the corresponding targeted signals in Section 4.

A.1 The choices of k for absolute value recovery

Recall that $\mathbf{K}_B = \mathbf{P}_A\mathbf{s}_B + 2\mathbf{g}_B = k\mathbf{s}_B + 2\mathbf{g}_B$. Hence, $|k\mathbf{s}_B[i]| - |2\mathbf{g}_B[i]| \leq |\mathbf{K}_B[i]| \leq |k\mathbf{s}_B[i]| + |2\mathbf{g}_B[i]|$. Moreover, if $|\mathbf{K}_B[i]| < \lfloor \frac{q}{4} \rfloor$ the corresponding signal is 0, and the signal is 1 if $\lceil \frac{q}{4} \rceil < |\mathbf{K}_B[i]| < \lfloor \frac{3q}{4} \rfloor$. Thus, a signal is zero in a stable region if

$$|k\mathbf{s}_B[i]| + |2\mathbf{g}_B[i]| < \left\lfloor \frac{q}{4} \right\rfloor \Leftrightarrow k < \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{|\mathbf{s}_B[i]|}, \tag{16}$$

and 1 in a stable region if

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{|\mathbf{s}_B[i]|} < k < \frac{\lfloor \frac{3q}{4} \rfloor - |2\mathbf{g}_B[i]|}{|\mathbf{s}_B[i]|}. \tag{17}$$

We start with the frist targeted signal $(0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1)$. When $|\mathbf{s}_B[i]| \leqslant 7$, the corresponding signal $\omega_B[i]$ is in the stable region of 0, otherwise $\omega_B[i]$ is in the stable region of 1. Thus, according to Equation (16), we need to choose k_1 such that

$$k_1 < \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{7}.$$

When $7 < |\mathbf{s}_B[i]| \leqslant 15$, based on Equation (17), we need to choose k_1 such that

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{8} < k_1 < \frac{\lfloor \frac{3q}{4} \rfloor - |2\mathbf{g}_B[i]|}{15}.$$

Combing the above two results, we have

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{8} < k_1 < \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{7}. \tag{18}$$

For k_2, the corresponding targeted signal is $(0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0)$ as $|\mathbf{s}_B[i]|$ increases from 0 to 15. From our observation, we know that the signal is always 0 when $|\mathbf{s}_B[i]|$ increases from 0 to 3, and when $|\mathbf{s}_B[i]| \geqslant 12$. Based on Equation (16), we have

$$\frac{\lceil \frac{3q}{4} \rceil + |2\mathbf{g}_B[i]|}{12} < k_2 < \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{3}.$$

When $4 \leqslant |\mathbf{s}_B[i]| \leqslant 11$, the signal changes to 1. Thus, by Equation (17),

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{4} < k_2 < \frac{\lfloor \frac{3q}{4} \rfloor - |2\mathbf{g}_B[i]|}{11}.$$

Then we conclude that

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{4} < k_2 < \frac{\lfloor \frac{3q}{4} \rfloor - |2\mathbf{g}_B[i]|}{11}. \tag{19}$$

For k_3, when $|\mathbf{s}_B[i]|$ increases from 0 to 15, the corresponding targeted signal is $(0,0,1,1,0,0,1,1,0,0,1,1,0,0,1,1)$. Similarly to before, we conclude that We observe that the signal periodically changes in a way like $0 \to 1 \to 0$. We choose $4k_3$ to be close to q, that is, k_3 is close to $\frac{q}{4}$. Since the signal is 0 when $|\mathbf{s}_B[i]| = 1$, according to Equation (16), we know

$$k_3 < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]|. \tag{20}$$

From Equation (20) we know that when $|\mathbf{s}_B[i]| \pmod 4 = 1$, $|k_3\mathbf{s}_B[i]| < \lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|$, which ensures that the corresponding signals are 0. Therefore, when the signal is 1, $|\mathbf{s}_B[i]| \pmod 4 = 2$ or 3. According to Equation (17), we have

$$\left\lceil \frac{q}{4} \right\rceil + |2\mathbf{g}_B[i]| < k_2|\mathbf{s}_B[i]| \pmod q < \left\lfloor \frac{3q}{4} \right\rfloor - |2\mathbf{g}_B[i]|, \tag{21}$$

where $|\mathbf{s}_B[i]| \pmod 4 = 2$ or 3. When $|\mathbf{s}_B[i]| \pmod 4 = 0$ where the corresponding signal changes from 1 to 0, k_3 should satisfy

$$k_2|\mathbf{s}_B[i]| \pmod q > \left\lceil \frac{3q}{4} \right\rceil + |2\mathbf{g}_B[i]|. \tag{22}$$

Combining Equation (20), (21), and (22), we can deduce

$$\frac{3q + \lfloor \frac{q}{4} \rfloor + |2\mathbf{g}_B[i]|}{14} < k_3 < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]|,$$

or equivalently,

$$\left\lfloor \frac{q}{4} \right\rfloor - \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{14} < k_3 < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]|. \tag{23}$$

Similarly, for the targeted signal $(0,1,0,1,0,1,0,1,0,1,0,1,0,1,0,1)$, we observe that, every time the signal changes, $|\mathbf{s}_B[i]|$ is added by 1. Therefore, k_4 is set near $\frac{q}{2}$.To be specific, we have

$$\left|\mathbf{s}_B[i](\left\lfloor \frac{q}{2} \right\rfloor - k_4)\right| < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]| \quad \text{or} \quad \left|\mathbf{s}_B[i](k_4 - \left\lfloor \frac{q}{2} \right\rfloor)\right| < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]|.$$

Since $|\mathbf{s}_B[i]|$ is at most 15, k_4 satisfies

$$\left\lfloor \frac{q}{2} \right\rceil - \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{15} < k_4 < \left\lfloor \frac{q}{2} \right\rceil + \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{15}. \tag{24}$$

For parameters of DXL-KE, this means concretely $k_1 \in (515.88, 580.86)$, $k_2 \in (1031.75, 1114.36)$, $k_3 \in (3805.57, 4066)$, and $k_4 \in (7921.93, 8464.07)$. Consequently, we select $k_1 = 550$, $k_2 = 1,050$, $k_3 = 4,000$, and $k_4 = 8,192$.

A.2 The choices of k in sign recovery

In Section 4, we follow a similar way as that in Step 1 to decide the range of k_1 according to the corresponding targeted signal in Table 3. For k_j, where $j \in [2,5]$, we adopt the approaches mentioned above to obtain the remaining conditions that k_j satisfies.

We first consider the targeted signal

$$(0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0)$$

to choose k_2. Similarly, we can deduce

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{8} < k_2 < \frac{\lfloor \frac{3q}{4} \rfloor - |2\mathbf{g}_B[i]|}{23}. \tag{25}$$

For the third targeted signal

$$(0,0,0,0,1,1,1,1,1,1,1,1,0,0,0,0,0,0,0,0,1,1,1,1,1,1,1,1,0,0,0),$$

we observe that the signal changes from 0 to 1 and then to 0 twice as $|\mathbf{s}_B[i]|$ increases from 0 to 30. Thus, k_3 should be close to $\frac{q}{16}$. Since the signal first changes from 0 to 1 when $|\mathbf{s}_B[i]| = 4$,

$$k_3 > \frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{4}.$$

Similar to the selection of k in Step 1, when the signal corresponding to $|\mathbf{s}_B[i]|$ is 0 or 1, combining Equation (16) and (17) we obtain

$$\frac{\lceil \frac{q}{4} \rceil + |2\mathbf{g}_B[i]|}{4} < k_3 < \left\lfloor \frac{q}{16} \right\rfloor + \frac{\lfloor \frac{q}{16} \rfloor - |2\mathbf{g}_B[i]|}{27}. \tag{26}$$

For the last two targeted signals, they both change in a same way as that in Step 1, except that $|\mathbf{s}_B[i]|$ increases from 0 to 30. Similarly, we have

$$\left\lfloor \frac{q}{4} \right\rfloor - \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{30} < k_4 < \left\lfloor \frac{q}{4} \right\rfloor - |2\mathbf{g}_B[i]|, \tag{27}$$

and

$$\left\lfloor \frac{q}{2} \right\rceil - \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{30} < k_5 < \left\lfloor \frac{q}{2} \right\rceil + \frac{\lfloor \frac{q}{4} \rfloor - |2\mathbf{g}_B[i]|}{30}. \tag{28}$$

B Parameter choices in the weighted strategy attack on DXL-KE

Here we further analyze how to choose proper k and k'. First, following our idea, we should ensure that $|k\mathbf{s}_B[i+1]|$ is larger than the distance between $\lfloor \frac{q}{4} \rfloor$ and $|k'\mathbf{s}_B[i] \bmod q|$. To take into account the errors $2\mathbf{g}_B[i]$, we have

$$|k\mathbf{s}_B[i+1]| > \left| \left\lfloor \frac{q}{4} \right\rfloor - |k'\mathbf{s}_B[i] \bmod q| \right| + 1 + |2\mathbf{g}_B[i]|. \tag{29}$$

Table 3: Signals $\omega_{B_j}[i]$ for k_j and $s[i] = |\mathbf{s}_B[i] + \mathbf{s}_B[i+1]|$ in DXL-KE with $j+1, 2, 3, 4, 5$ and $i = 0, ..., n-1$

$s[i]$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
$k_1 = 260$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
$k_2 = 525$	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
$k_3 = 1,050$	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0
$k_4 = 4,000$	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1
$k_5 = 8,192$	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0

What we must emphasize is that all the coefficients are in $\mathbb{Z}_q$, when $|k\mathbf{s}_B[i+1]|$ is too large, the corresponding signals $Sig(|k'\mathbf{s}_B[i]|+|k\mathbf{s}_B[i+1]|)$ and $Sig(\big||k'\mathbf{s}_B[i]|-|k\mathbf{s}_B[i+1]|\big|)$ could be the same. To avoid this case, $|k\mathbf{s}_B[i+1]|$ should be less than $\frac{q}{2}$. More specifically,

$$|k\mathbf{s}_B[i+1]| < \left| \left\lfloor \frac{q}{2} \right\rfloor - \left| \left\lfloor \frac{q}{4} \right\rfloor - |k'\mathbf{s}_B[i] \bmod q| \right| \right| - |2\mathbf{g}_B[i]|. \tag{30}$$

Since we have already known the absolute values of $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$ in Step 1, we further discuss how to choose k according to whether $|\mathbf{s}_B[i]|$ is odd or even in the following.

First, we consider the case when $\mathbf{s}_B[i]$ is odd. Since q is 16385 in DXL-KE, we set $k' = 4096$ which is $\lfloor \frac{q}{4} \rfloor$. Then we choose $k = 500$ which satisfies Equations (29) and (30) when $|\mathbf{s}_B[i+1]| \in [0, 15]$. For odd $|\mathbf{s}_B[i]|$, we discuss two cases: $|\mathbf{s}_B[i]| \in \{1, 5, 9, 13\}$ and $|\mathbf{s}_B[i]| \in \{3, 7, 11, 15\}$.

- When $|\mathbf{s}_B[i]| \pmod 4 = 1$, (i.e. $|\mathbf{s}_B[i]| = 1, 5, 9, 13$), $k'|\mathbf{s}_B[i]| \pmod q \lessapprox \frac{q}{4}$. If the relative sign is $+$, $|\mathbf{K}_B[i+1]| > \lceil \frac{q}{4} \rceil$ and the corresponding signal is 1, otherwise, $|\mathbf{K}_B[i+1]| < \lfloor \frac{q}{4} \rfloor$ and $\omega_B[i+1] = 0$.
- When $|\mathbf{s}_B[i]| \pmod 4 = 3$, (i.e. $|\mathbf{s}_B[i]| = 3, 7, 11, 15$), $|k'\mathbf{s}_B[i]| \pmod q \lessapprox \frac{3q}{4}$. Therefore, when the relative sign is $+$, $|\mathbf{K}_B[i+1]| > \lceil \frac{3q}{4} \rceil$ corresponding to $\omega_B[i+1] = 0$; when the relative sign is $-$, it corresponds to $\omega_B[i+1] = 1$.

For the case that $\mathbf{s}_B[i]$ is even, we choose two values for setting k' in order to let $|k'\mathbf{s}_B[i]|$ approach $\frac{q}{4}$ as closely as possible. We first set $k' = 5461$ which is $\lfloor \frac{q}{3} \rfloor$. Based on Equation (29), we know that k is at least larger than $\frac{q}{12}$. Thus, if we let k also satisfy Equation (30), the range of $\mathbf{s}_B[i+1]$ needs to be restricted to be at most $[-4, 4]$. Then, we choose $k = 1500$ to satisfy Equation (29) and (30) when $\mathbf{s}_B[i+1] \in [-4, 4]$.

Similarly, for even $\mathbf{s}_B[i]$, there are still two cases: $|\mathbf{s}_B[i]| \pmod 3 = 2$ (i.e. $|\mathbf{s}_B[i]| \in \{2, 8, 14\}$) and $|\mathbf{s}_B[i]| \pmod 3 = 1$ (i.e. $|\mathbf{s}_B[i]| \in \{4, 10\}$). To cover a larger range of $\mathbf{s}_B[i+1]$, we use the second value to set k' as $k' = 3277$ which is $\frac{q}{5}$. Thus, when $|\mathbf{s}_B[i]| \pmod 5 = 1$ where $|\mathbf{s}_B[i]| = 6$ and $|\mathbf{s}_B[i]| \pmod 5 = 4$ where $|\mathbf{s}_B[i]| \in \{4, 14\}$, $|k'\mathbf{s}_B[i]| = \frac{q}{5}$ is close to $\frac{q}{4}$. Then we set $k = 900$ satisfying Equation (29) and (30) when $\mathbf{s}_B[i+1] \in [-8, 8]$. How to determine the relative sign of even $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$ using the collected signals is presented in Table **??**.

Note that to recover the relative sign of two non-zero coefficients separated by z consecutive zeros, the adversary also needs to set $\mathbf{P}_A = k + k'x^{z+1}$ to query the oracle. By doing so, the adversary can also recover relative signs in cases not included above. For instance, if $(\mathbf{s}_B[i-1], \mathbf{s}_B[i], \mathbf{s}_B[i+1]) = (5, 2, 5)$ and $z = 1$, the adversary can recover the relative sign of $\mathbf{s}_B[i-1]$ and $\mathbf{s}_B[i]$, but not the relative sign of $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$ by querying the oracle with $\mathbf{P}_A = k + k'x$ according to the above discussion. Then the adversary needs to query the oracle with $\mathbf{P}_A = k + k'x^2$ to recover the relative sign of $\mathbf{s}_B[i-1]$ and $\mathbf{s}_B[i+1]$. At the same time, the adversary can combine the information of two queries to recover the relative sign of $\mathbf{s}_B[i]$ and $\mathbf{s}_B[i+1]$.

C Parameter Choices of the Improved Attack on DBS-KE

In order to ensure that we can collect the targeted signals from the stable regions, k_1, k_2, k_3 and k_4 should satisfy conditions similar to those of the improved attack against DXL, except that the errors $|2\mathbf{g}_B[i]|$ are replaced with $|\Delta[i]|+|\varepsilon[i]|$. Then, based on Equations (18), (19), (23), and (24), it holds

$$\begin{aligned}
\frac{\left\lfloor \frac{q}{4} \right\rceil + |\Delta_1[i]| + |\varepsilon[i]|}{8} &< k_1 < \frac{\left\lfloor \frac{q}{4} \right\rceil - (|\Delta_1[i]| + |\varepsilon[i]|)}{7}, \\
\frac{\left\lfloor \frac{q}{4} \right\rceil + |\Delta_2[i]| + |\varepsilon[i]|}{4} &< k_2 < \frac{\left\lceil \frac{3q}{4} \right\rceil - (|\Delta_2[i]| + |\varepsilon[i]|)}{11}, \\
\left\lfloor \frac{q}{4} \right\rceil - \frac{\left\lfloor \frac{q}{4} \right\rfloor - (|\Delta_3[i]| + |\varepsilon[i]|)}{14} &< k_3 < \left\lfloor \frac{q}{4} \right\rceil - (|\Delta_3[i]| + |\varepsilon[i]|), \\
\left\lfloor \frac{q}{2} \right\rceil - \frac{\left\lfloor \frac{q}{4} \right\rfloor - (|\Delta_4[i]| + |\varepsilon[i]|)}{15} &< k_4 < \left\lfloor \frac{q}{2} \right\rceil + \frac{\left\lfloor \frac{q}{4} \right\rceil - (|\Delta_4[i]| + |\varepsilon[i]|)}{15}.
\end{aligned} \tag{31}$$

Therefore, we can determine the conditions that $\Delta_1[i], \Delta_2[i], \Delta_3[i]$, and $\Delta_4[i]$ should satisfy. For example, $|\Delta_1[i]| < \left\lfloor \frac{q}{4} \right\rceil / 15 - |\varepsilon[i]|$.

In the DBS-KE, $q = 26,038,273$, $n = 512$, and $\alpha = 4.19$. According to the Section 5.3 in [7], $\varepsilon[i]$ is normal distributed with standard deviation $\sqrt{12n\alpha^4 + 4\alpha^2}$. Since $4.5\sqrt{12n\alpha^4 + 4\alpha^2} \approx 6192.61$, we assume that $|\varepsilon[i]| \leq 6200$. We can first calculate the ranges of $\Delta_j[i]$ $(j = 1, 2, 3, 4)$, obtaining that $|\Delta_1[i]| < 427771.2$, $|\Delta_2[i]| < 427771.47$, $|\Delta_3[i]| < 427771.2$, and $|\Delta_4[i]| < 6503368$. Then we select the specific values for $\Delta_j[i]$ as shown in Section 5.1. Subsequently, we can get the ranges of k_1, k_2, k_3, and k_4 as $k_1 \in (867721, 868195.43)$, $k_2 \in (1735442, 1736045.91)$, $k_3 \in (6075470.29, 6077368)$, and $k_4 \in (13018912.47, 13019361.53)$.