Post-Quantum Backdoor for Kyber-KEM

Wenwen Xia[1,2], Geng Wang[3,2⋆], and Dawu Gu[3,2,1⋆]

[1] School of Cyber Engineering, Xidian University, Xi'an, 710071, China
xiawenwen@stu.xidian.edu.cn
[2] Lab of Cryptology and Computer Security, Shanghai Jiao Tong University, Shanghai, 200240, China
[3] School of Electronic, Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, 200240, China
{wanggxx, dwgu}@sjtu.edu.cn

Abstract. Kleptography, also known as cryptographic backdoor, poses a significant threat to cryptographic algorithms by clandestinely embedding a backdoor through the use of another cryptographic algorithm, often leveraging public-key encryption techniques. Achieving a carefully designed kleptographic attack demands reducing the detectability of the backdoor to the complexity of cryptographic hard problems. In this paper, we explore the application of kleptography to CRYSTALS-Kyber, a post-quantum algorithm standardized by NIST. Leveraging the Classic McEliece Key Encapsulation Mechanism (KEM), also a NIST round-4 candidate, we devise a backdoor for both Kyber-768 and Kyber-1024. Similar to the approach proposed by Yang et al. [AsiaCCS 2020], our backdoor manipulates only the key generation algorithm, ensuring compatibility with the key encapsulation mechanism (KEM) variant of Kyber, rather than solely the public key encryption. Moreover, we present a stronger definition of undetectability within a public-key framework, capturing the intuition that the backdoor could be hidden in both the Key Generation and Encryption processes, and prove the undetectability of our backdoor under this new definition. In addition, compared with Yang et al, our backdoor has two advantages: (1) We provide post-quantum undetectability; (2) Our backdoor remains independent of the public key seed, preserving public undetectability against certain potential countermeasures. The undetectability of our backdoor hinges on reducing it to the decisional version of the syndrome decoding problem (SDP) for Goppa codes.

Keywords: Kelptography · Backdoor attack · CRYSTALS-Kyber · Post-quantum cryptography

1 Introduction

Kleptography, introduced by Young and Yung in 1997 [YY97a], is a method for constructing the cryptographic backdoor within a cryptosystem. In simple

⋆ Corresponding author.

terms, kleptography relies on the security of another public key cryptographic algorithm to ensure the undetectability of the backdoor. Kleptography has already proven to be a threat for both symmetric and asymmetric cryptography. For asymmetric schemes, kleptography attacks can be applied to many standardized algorithms [YY97b, YY16], such as RSA or ECDSA. It succinctly emphasizes that kleptography has practical implications beyond theory. Given that many institutions including banks or governments may outsource programming to external parties, incorporating cryptosystem implementations into their products. It is possible that backdoors could be implanted in these outsourced binary codes using kleptography. Such backdoors may remain undetected unless uncovered through reverse engineering. Therefore, kleptography attacks indeed pose practical threats for users.

With the rapid advancement of quantum computing, the need for developing new methods for public-key algorithms resistant to quantum attacks has become urgent, a field commonly referred to as post-quantum cryptography. In 2022, NIST announced the standardization of four post-quantum algorithms, among which CRYSTALS-Kyber [ABD+20] stands out as the only key-encapsulation mechanism (KEM). It is foreseeable that Kyber will emerge as one of the most widely adopted public-key algorithms in the post-quantum era.

Kyber is a lattice-based scheme, relying on the module learning with errors (Module-LWE, or MLWE) problem for its security. To achieve indistinguishability against adaptive chosen ciphertext attacks (IND-CCA2), Kyber initially presented a public key encryption (PKE) scheme with only IND-CPA security. Subsequently, it utilized the quantum Fujisaki-Okamoto (QFO) transformation to convert the IND-CPA PKE into an IND-CCA2 key encapsulation mechanism (KEM).

Recent works, such as those in [XY18, YCL+20], have proposed common backdoor methods for LWE-type public key encryptions by modifying encryption algorithms Enc, with undetectability based on LWE and NTRU, respectively. However, these backdoors are not applicable to Kyber-KEM. In the decapsulation algorithm of Kyber-KEM, the ciphertext must be reproduced with the same encapsulation algorithm, which includes the encryption algorithm as its subroutine. Therefore, any modification to the encapsulation or encryption algorithm would be easily detected.

In [YXP20], Yang et al. introduced a novel construction for lattice backdoors, where the modification is limited to the key generation algorithm KeyGen. It's worth noting that their original construction had flaws, later rectified by [Hem22]. While they initially demonstrated their backdoor construction on the PKE version of NewHope [ADPS16], it can readily be extended to Kyber, including Kyber-KEM. However, their approach employs Diffie-Hellman key exchange from elliptic curves, which lacks post-quantum security. We highlight that, despite symmetric cryptosystems typically being considered post-quantum secure, kleptography necessitates the use of a public key scheme for hiding the backdoor, as outlined in the definition by Young and Yung [YY97a]. This requirement ensures that even if the backdoor is uncovered via reverse engineering,

the secret key for backdoor recovery remains concealed. Thus, determining the exact time of implantation and identifying the infected keys or ciphertexts remains elusive. For a more in-depth discussion, we refer readers to the original paper [YY97a].

The backdoor construction of Yang et al. In [YXP20] and [Hem22], the construction of the backdoor is described as follows: Let G be the base point of an elliptic curve, where the modulus q is very close to 2^{256}. Let p be the order of G, and $a \leftarrow \mathbb{Z}_p$ represent the backdoor private key, with aG being the backdoor public key. To embed a backdoor into the key generation algorithm, a value $b \leftarrow \mathbb{Z}_p$ is first chosen, followed by encoding bG and $b(aG)$ into pseudorandom 256-bit strings, which can be accomplished using techniques outlined in [BHKL13] as pointed out by [Hem22]. The encoding of bG and $b(aG)$ serves as the public seed and secret seed, respectively, in the key generation algorithm of the post-quantum scheme (see Section 2.2 for the description of Kyber as an example). Subsequently, upon receiving the post-quantum public key, the backdoor secret key holder can recover bG and calculate $a(bG)$ to determine the secret seed, thereby recovering the secret key. It's worth noting that for most lattice-based schemes, the LWE samples used as the public key must be generated from a public seed, with the seed included in the real public key to prevent a hidden trapdoor. Therefore, this backdoor construction can potentially affect most lattice-based schemes.

However, the lengths of the public seed and secret seed are only 256 bits, necessitating that the ciphertext of the public key encryption scheme used in the backdoor construction must have, or can be compressed into, a length of at most 256 bits. This requirement cannot be satisfied for any known post-quantum public-key schemes. As a result, it remains an open problem about: *Is there a kleptographic attack on Kyber-KEM feasible in the post-quantum era?*

In this paper, we address this issue by introducing a backdoor construction for Kyber-768 and Kyber-1024, derived from Classic McEliece, a code-based post-quantum Key Encapsulation Mechanism (KEM) that is a candidate in the NIST 4th round [ABC+22]. In our backdoor construction, we employ the 128-bit secure parameter selection mceliece-348864, which features a 768-bit ciphertext length, the shortest among all post-quantum schemes to the best of our knowledge. We reduce the undetectability of our trapdoor to the decisional version of the syndrome decoding problem (SDP) for Goppa codes, which essentially addresses the pseudorandomness of McEliece ciphertexts. Hence, it is appropriate to assert that our backdoor exhibits post-quantum undetectability. While our focus in this paper is primarily on Kyber, the same technique can be applied to backdoor other LWE-based post-quantum algorithms, such as NewHope and Frodo.

1.1 Technical Overview

In this paper, we employ a technique similar to [XY18, YCL+20], but with a variation: instead of embedding the secret seed into the Least Significant Bits (LSBs) of the ciphertext, we embed the ciphertext of the secret seed into the LSBs of the public key element $\mathbf{t} = \mathbf{As} + \mathbf{e} \bmod^{\pm} q$, where $\mathbf{e}$ represents a small

error term. In Kyber, only $\mathbf{s}$ is included in the secret key. Therefore, even if we require $\mathbf{t}$ to match $\mathbf{As}$ precisely, a small deviation (e.g., ± 1) is still permissible in $\mathbf{t}$, allowing the LSBs of $\mathbf{t}$ to leak the ciphertext of the Kyber secret seed. For this embedding to function, the number of elements on $\mathbb{Z}_q$ in the Kyber public key $\mathbf{t}$ must be equal to or greater than the length of the embedded message, which is a McEliece ciphertext (of the Kyber secret seed) with a bit-length of 768. This limitation explains why our backdoor can only be applied to Kyber-768 and Kyber-1024.

However, the primary challenge arises from the fact that while the ciphertext of the Kyber secret seed is pseudorandom, $\mathbf{e}$ follows a specific distribution, such as the central binomial distribution $\mathcal{B}_2$. If we simply modify the LSBs of $\mathbf{t}$ to align with the ciphertext of the Kyber secret seed, the secret key holder can use their secret key to recover $\mathbf{e}$ and verify whether $\mathbf{e}$ conforms to the expected distribution. This process enables the detection of the backdoor with a high probability. This issue is also encountered in previous works [XY18, RBC+24], leading to the failure of their schemes to achieve provable undetectability.

In this work, we propose a novel technique to address this challenge. Instead of modifying $\mathbf{t}$, we alter the error term $\mathbf{e}$ itself. To embed the ciphertext C of McEliece into the Least Significant Bits (LSBs) of $\mathbf{t}$, we first compare the LSB of the i-th coefficient in $\mathbf{As}$ with C_i (the i-th bit of C). If the two bits are identical, we sample an even error e_i (the i-th element of $\mathbf{e}$); if they are different, we sample an odd error e_i. Since the secret seed is never disclosed to the secret key holder, the correct $\mathbf{e}$ remains undisclosed. From the perspective of the secret key holder, e_i still adheres to the expected distribution, thereby rendering the backdoor undetectable.

Here we give a diagram (Fig. 1) to explain the concrete process of our backdooring Kyber scheme.

1.2 Our Contribution

We list the contributions of this paper below:

- We provide a formal definition for undetectability in a public-key scheme, which encompasses the concealment of the backdoor in both KeyGen and Enc. While [YXP20] offers a formal definition, we demonstrate that their definition lacks sufficient strength and is not applicable to KEMs.
- We present the first post-quantum backdoor for a post-quantum Key Encapsulation Mechanism (KEM) that is provably undetectable. Previous post-quantum backdoor constructions [XY18, YCL+20] are applicable only to post-quantum Public Key Encryption (PKE) schemes and can be readily detected if the PKE is transformed into an IND-CCA2 secure KEM using Fujisaki-Okamoto (FO) or quantum Fujisaki-Okamoto (QFO) transformations. Additionally, we provide a reduction demonstrating the undetectability of our trapdoor to the post-quantum hard problem of decisional Syndrome Decoding Problem (SDP).

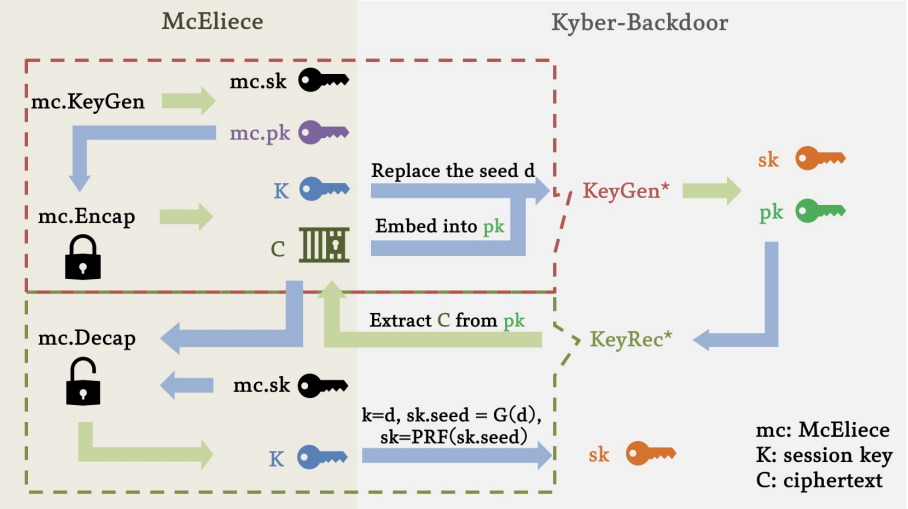

Fig. 1. Technical Overview of our Kyber-backdoor Scheme. The notation we used in this figure has been explained in Sec. 2.

– We propose a potential remedy for the backdoor introduced in [YXP20, Hem22] to nullify its public undetectability. Furthermore, we demonstrate that this remedy does not impact the effectiveness of our own backdoor. Additionally, we assert that rectifying our backdoor to achieve public undetectability is highly challenging, thus making our backdoor more robust compared to previous ones, even without considering its post-quantum characteristics.

1.3 Related Works

Kleptographic attacks targeting post-quantum algorithms have garnered significant attention since the initiation of the NIST Post-Quantum Cryptography (PQC) standardization process. The pioneering backdoor construction for post-quantum algorithms is documented in [KLT17], focusing on the NTRU encryption scheme. However, it was demonstrated in [YXP20] that their backdoor is detectable. Subsequently, [XY18, YCL+20] also presented their own backdoor constructions for LWE-type Public Key Encryption (PKE) schemes. However, their backdoor constructions cannot be applied to post-quantum Key Encapsulation Mechanisms (KEM) with IND-CCA2 security.

The seminal work by [YXP20] marks the inception of post-quantum Key Encapsulation Mechanisms (KEMs) with embedded backdoors. Despite its groundbreaking nature, the initial construction suffered from flaws, prompting corrective measures by [Hem22]. However, reliance on the Diffie-Hellman assumption over elliptic curves compromises their post-quantum undetectability. Recent research by [RBC+24] parallels our approach in constructing Kyber backdoors.

Nonetheless, their claim of public undetectability hints at potential vulnerability to detection by Kyber secret key holders. Intriguingly, our investigation uncovers that their backdoors are also susceptible to public detection, a topic we delve into further in Section 3.2.

We provide Table 1 to succinctly summarize existing research and juxtapose it with our own contributions.

Table 1. Comparison for previous backdoors on post-quantum schemes. The column "Post-Quantum" means that the backdoor construction is based on a Post-Quantum public key cryptosystem. The column of "Undetectability" shows the undetectability of each work. The column of "Provable" means that a formal proof of undetectability is provided.

Work	Post-Quantum	Valid for KEM	Undetectability	Provable
Kwant et al [KLT17]	×	×	×	N/A
Xiao and Yu [XY18]	✓	×	✓	×
Yang et al [YCL+20]	✓	×	✓	✓
Yang et al [YXP20]	×	✓	×	N/A
Hemmert [Hem22]	×	✓	✓	✓
Ravi et al [RBC+24]	✓	✓	×	N/A
This Work	✓	✓	✓	✓

2 Preliminaries

Notations. Let $\mathsf{LSB}(x)$ be the least significant bit of $x \in \mathbb{Z}$, that is $x \mod 2$. Let $R_q = \mathbb{Z}_q[X]/(X^n+1)$ be the polynomial ring, where each polynomial has degree at most $n-1$. Let $\mathbf{t} \in R_q^k$ be a vector of polynomials, each element $v = c_0 + c_1 X + ... + c_{n-1}X^{n-1}$ in $\mathbf{t}$ can be expressed by its coefficient vector $\mathrm{coeff}(v) = (c_0, ..., c_{n-1})$. For $\mathbf{t} = (v_1, ..., v_k)$, we express $\mathbf{t}$ as a vector of length nk: $\mathrm{coeff}(\mathbf{t}) = (\mathrm{coeff}(v_1)\|...\|\mathrm{coeff}(v_k))$, where $\|$ stands for vector concatenation. For $i = 1, ..., nk$, we use $[\mathbf{t}]_i$, or simpler, t_i to express the i-th element in $\mathrm{coeff}(\mathbf{t})$. Let $\mathsf{LSBs}(\mathbf{t})$ be the vector of least significant bits of $\mathrm{coeff}(\mathbf{t})$. Furthermore, for a binary string K, we let $K_i \in \{0,1\}$ be the i-th bit of K.

For any binary string $\mathbf{s} \in \{0,1\}^*$, let $\mathrm{wt}(\mathbf{s})$ be the hamming weight of $\mathbf{s}$, that is, the number of 1's in $\mathbf{s}$.

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. $-\frac{\alpha-1}{2} < r' \leq \frac{\alpha-1}{2}$) such that $r' = r \bmod \alpha$.

Denote by $\mathcal{B}$ the set $\{0, ..., 255\}$, i.e. the set of 8-bit unsigned integers (bytes), and denote by $\mathcal{B}^k$ the set of byte arrays of length k.

2.1 Kleptography

We introduce the idea of a kleptography attack by Young and Yung [YY97a], which is presented through a "SETUP" (Secretly Embedded Trapdoor with Universal Protection) mechanism.

Definition 1 (SETUP Mechanism). *Let C be a publicly known cryptosystem. A SETUP mechanism is an algorithmic modification made in C to get C', such that:*

- *The input of C' agrees with the public specifications of the input of C.*
- *C' computes using the attacker's public encryption function E (and possibly other functions as well), contained within C'. The attacker's private decryption function D is not contained within C' and is known only by the attacker.*
- *The output of C' agrees with the public specifications of the output of C. At the same time, the attacker can easily access a part or the whole of the output which is needed to perform the attack.*
- *Furthermore, the output of C and C' are polynomially indistinguishable to everyone (including those who have access to the code of C') except the attacker.*

Since the definition above is mostly informal, Yang et al [YXP20] gave a new definition for undetectability in their work. However, since in their definition, the detector does not have access to the secret key, their definition is not strong enough to capture the undetectable requirements if the backdoor is hidden in KeyGen other than Enc (although they used the term "strong undetectability"). In this paper, we rename their definition as "public undetectability", and also give a new definition to handle the case where the backdoor is hidden in KeyGen.

Definition 2 (Public Undetectability). *The public undetectability of a backdoored cryptosystem is defined by an interactive game between a challenger $\mathcal{C}$ and a detector $\mathcal{D}$:*

- *The challenger $\mathcal{C}$ samples $b \leftarrow \{0,1\}$. If $b = 0$, he will run the backdoored algorithm; otherwise, he will run the plain algorithm.*
- *After $\mathcal{C}$ runs the key generation algorithm, the public key is sent to $\mathcal{D}$.*
- *$\mathcal{D}$ can ask $\mathcal{C}$ to encrypt messages at his will and outputs a bit $b' = 0$ if he decides the encryption scheme is backdoored, otherwise outputs $b' = 1$.*
- *$\mathcal{D}$ succeeds if $b = b'$.*

The scheme is publicly undetectable, if for any polynomial time detector $\mathcal{D}$, $\Pr(b = b') - 1/2$ is negligible.

Definition 3 (Strict Undetectability). *The strict undetectability of a backdoored cryptosystem is defined by an interactive game between a challenger $\mathcal{C}$ and a detector $\mathcal{D}$:*

- *The challenger $\mathcal{C}$ samples $b \leftarrow \{0,1\}$. If $b = 0$, he will run the backdoored algorithm; otherwise, he will run the plain algorithm.*
- *After $\mathcal{C}$ runs the possibly backdoored key generation algorithm, both public key and secret key are sent to $\mathcal{D}$.*
- *$\mathcal{D}$ can ask $\mathcal{C}$ to run the possibly backdoored Enc algorithm to encrypt messages (for PKE) or encapsulate keys (for KEM) at his will. $\mathcal{D}$ finally outputs a bit $b' = 0$ if he decides the encryption scheme is backdoored, otherwise outputs $b' = 1$.*

– $\mathcal{D}$ *succeeds if* $b = b'$.

The scheme is strictly undetectable, if for any polynomial time detector $\mathcal{D}$, $\Pr(b = b') - 1/2$ *is negligible.*

2.2 Kyber Key Generation and Central Binomial Distribution

Overview of Kyber Key Generation In Kyber [ABD+20], the Key generation algorithm can be simplified as Alg. 1, ignoring the NTT acceleration and compression functions. It firstly generates a 256-bit seed d in a uniformly random distribution $\mathcal{B}^{32}$ (Defined in Sec. 2), and calls a Hash function G to generate a random seed $pk.seed$ for generating public key and a random seed $sk.seed$ for generating secret key by computing $(sk.seed, pk.seed) = G(d)$. Then it samples the public matrix $\mathbf{A} \in R_q^{k \times k}$ using $pk.seed$ through sample function Parse and the extendable function XOF, samples the secret vector $\mathbf{s} \in R_q^k$ and the noise vector $\mathbf{e} \in R_q^k$ using $sk.seed$ thrugh the pseudorandom function PRF, which follows the Central Binomial Distribution (Def. 4). Here, $n = 256$, $q = 3329$, $k = 2, 3, 4$ for Kybe r-512, Kyber-768 and Kyber-1024 respectively. Next, it computes the public vector $\mathbf{t} = \mathbf{As} + \mathbf{e} \bmod^{\pm} q$. Finally, it outputs $pk = (\mathbf{t}, pk.seed)$ and $sk = \mathbf{s}$.

output: $pk \leftarrow (\mathbf{t}, pk.seed)$, $sk \leftarrow \mathbf{s}$
Function `Kyber.KeyGen():`
  $d \leftarrow \mathcal{B}^{32}$.
  $(sk.seed, pk.seed) \leftarrow G(d)$ //Hash Function G is declared in Kyber
  $(\mathbf{s}, \mathbf{e}) \leftarrow \mathsf{PRF}(sk.seed)$ //Sample $\mathbf{s}$ and $\mathbf{e}$ from $sk.seed$ in distribution B_η
  $\mathbf{A} \leftarrow \mathsf{Parse}(\mathsf{XOF}(pk.seed))$ //Sample $\mathbf{A}$ from $pk.seed$.
  $\mathbf{t} \leftarrow \mathbf{As} + \mathbf{e} \bmod^{\pm} q$;
  return $pk \leftarrow (\mathbf{t}, pk.seed)$, $sk \leftarrow \mathbf{s}$

Algorithm 1: Kyber Key Generation Algorithm KeyGen

Central Binomial Distribution Each coefficient in secret vector $\mathbf{s}$ and noise vector $\mathbf{e}$ in Kyber is sampled from a centered binomial distribution B_η for $\eta = 2$ (Kyber-768 and Kyber-1024) or $\eta = 3$ (Kyber-512). B_η in Kyber is defined as follows:

Definition 4 (Central Binomial Distribution B_η).

$$\text{Sample } (a_1, \cdots, a_\eta, b_1, ..., b_\eta) \leftarrow \{0,1\}^{2\eta}$$
$$\text{and output } z = \sum_{i=1}^{\eta} (a_i - b_i),$$

then z *follows Central Binomial Distribution* B_η.

2.3 Classic McEliece KEM and the Syndrome Decoding Problem

Classic McEliece KEM The Classic McEliece Key Encapsulation Mechanism (KEM) [ABC+22] is a code-based scheme whose security relies on the hardness of syndrome decoding of Goppa codes. It comprises Key Generation (mc.KeyGen), Encapsulation (mc.Encap), and Decapsulation (mc.Decap) algorithms. The Key Generation algorithm, mc.KeyGen, produces a key pair (mc.pk, mc.sk), where the public key is represented as a matrix $\mathbf{T} \in \{0,1\}^{(m_1 \cdot t) \times k}$. Specific algorithmic details are not necessary for consideration in this paper.

In mc.Encap, it inputs mc.$pk = \mathbf{T}$ and firstly generates a random binary vector $\mathbf{v} \in \{0,1\}^n$ of weight $wt(\mathbf{v}) = t$. Then, we can compute the ciphertext $C = \text{ENCODE}(\mathbf{v}, \text{mc}.pk) = (\mathbf{I}|\mathbf{T}) \cdot \mathbf{v}$. After that, it computes the session key $K = H(1, \mathbf{v}, C)$, where H is a Hash function declared in [ABC+22]. Finally, it outputs the the ciphertext C and session key K.

In mc.Decap, we should input the ciphertext C and the private key mc.sk. It decodes the $\mathbf{v}$ by calling a DECODE function, i.e. $\mathbf{v} = \text{DECODE}(C, \text{mc}.sk)$. After obtaining $\mathbf{v}$, compute $K = H(1, \mathbf{v}, C)$ to get the session key.

In `mceliece348864`, $m_1 = 12$, $t = 64$, $k = 2720$, $n = m_1 \cdot t + k = 3488$, thus the ciphertext size $m_1 t = 768$, which is the shortest among all post-quantum KEM to the best of our knowledge. This is also the reason why we choose `mceliece348864` to design the backdoor in our scheme.

Syndrome Decoding Problem The undetectability of our backdoor is based on the following (decisional) syndrome decoding problem (SDP) for Goppa codes. We define the problem as follows (in a slightly modified way):

Definition 5 ((Decisional) Syndrome Decoding Problem). *Given* $\mathbf{H} \in \{0,1\}^{(m_1 \cdot t) \times n}$ *and* $\mathbf{s} \in \{0,1\}^{m_1 \cdot t}$*, decide whether* $\mathbf{s}$ *is chosen uniform randomly from* $\{0,1\}^{m_1 \cdot t}$ *or from* $\{\mathbf{Hv} | \mathbf{v} \in \{0,1\}^n \wedge \text{wt}(\mathbf{v}) = t\}$.

If we choose $\mathbf{H} = (\mathbf{I}|\mathbf{T})$, where $\mathbf{T}$ is the McEliece public key, thus $\mathbf{H}$ represents a (variant of) random Goppa code, then the decisional SDP problem claims that a McEliece ciphertext is independent from a random string. Note that we rely on a stronger assumption compared with the submission Classic McEliece in NIST PQC round 4, where the submission requires only the hardness of the search version of SDP, that is, given $\mathbf{Hv}$ where $\text{wt}(\mathbf{v}) = t$, find $\mathbf{v}$.

It has been proven in [BMvT78] that both the search and decision versions of SDP for random codes are NP-complete. In most McEliece-type cryptosystems including Classic McEliece, it is assumed that the public key matrix $\mathbf{T}$ is indistinguishable from random, thus we have high confidence that the problem is hard for Classic McEliece ciphertexts.

3 Our Backdoor Construction with Classic McEliece

In this section, we give our backdoor construction for Kyber-768 and Kyber-1024. Our backdoor construction consists of two algorithms: the backdoor key

generation algorithm and the backdoor key recovery algorithm. The backdoor key generation algorithm embeds the backdoor into the seed d and the public key of Kyber. The backdoor key recovery algorithm is used to recover the secret key of Kyber, hence can decrypt the ciphertext in Kyber-PKE or decapsulate the session key in Kyber-KEM. In this section, we firstly introduce our backdoor key generation algorithm (Sec. 3.1) and backdoor key recovery algorithm (Sec. 3.2) designed through Classic McEliece. Then, we give proof of the strict undetectability of our backdoor construction in Sec. 3.3.

3.1 Backdoor Key Generation Algorithm

For the convenience of description, we simplify the public and secret key of Kyber as $pk = (\mathbf{t}, pk.seed)$ and $sk = \mathbf{s}$ (pk and sk in Kyber are actually the compressed ones after NTT transform), where $\mathbf{t} = \mathbf{As} + \mathbf{e} \bmod^{\pm} q$ with modulo q, $\mathbf{e}$ is the noise vector and $\mathbf{A}$ is sampled from $pk.seed$. It is worth emphasizing that both $\mathbf{s}$ and $\mathbf{e}$ are generated through the seed $d \leftarrow \mathcal{B}^{32}$ (a randomly selected 256-bit unsigned integer) in Kyber.

Let the key pairs generated by the key generation algorithm (Sec. 2.3) from McEliece KEM be mc.pk and mc.sk. Suppose the key generation algorithm in McEliece KEM is mc.KeyGen, the encapsulation algorithm is mc.Encap and decapsulation algorithm is mc.Decap. Denote the session key in McEliece KEM by K and its ciphertext by C. It implies that $(K, C) = \text{mc.Encap}(\text{mc}.pk)$ and $K = \text{mc.Decap}(\text{mc}.sk, C)$.

In our Backdoor Generation Algorithm (Alg. 2), we should firstly replace the seed d with McEliece session key K (Since d is never transferred, no one except the key generator knows the replacement of d). Then, we could embed C into the LSBs of $\mathbf{t}$ by sampling a different $\mathbf{e}$. In Kyber-768 and Kyber-1024, the probabilistic distribution of each coefficient in $\mathbf{s}$ and $\mathbf{e}$ is $B_2 = \{-2 : \frac{1}{16}, -1 : \frac{1}{4}, 0 : \frac{3}{8}, 1 : \frac{1}{4}, 2 : \frac{1}{16}\}$. Since $\Pr(\text{LSB}(e_i) = 1) = \Pr(\text{LSB}(e_i) = 0) = 1/2$, we can depart the probabilistic distribution of B_2 into two distributions: $\mathcal{D}_0 = \{-2 : \frac{1}{8}, 0 : \frac{3}{4}, 2 : \frac{1}{8}\}$ and $\mathcal{D}_1 = \{-1 : \frac{1}{2}, 1 : \frac{1}{2}\}$.

Next, if the LSB of i-th coefficient in $\mathbf{As}$ agrees with the i-th bit of C, then sample the i-th coefficient of $\mathbf{e}$ from $\mathcal{D}_0$, otherwise from $\mathcal{D}_1$, without changing the distribution of $\mathbf{e}$. Now the LSBs of $\mathbf{t} = \mathbf{As} + \mathbf{e}$ are *almost* the same as the bit representation in C. However, there are some border cases as we compute in modular q. For example, if $[\mathbf{As}]_i = (q-1)/2$, and $C_i \neq \text{LSB}([\mathbf{As}]_i)$, then we need to sample e_i from $\{-1, 1\}$. If $e_i = 1$, then $t_i = (q-1)/2+1 \mod^{\pm} q = -(q-1)/2$, which leads to $\text{LSB}(t_i) \neq C_i$. However, since the border cases rarely occur and can easily be detected, we leave them to the key recovery algorithm for further discussion.

3.2 Backdoor Key Recovery Algorithm

Since we modify KeyGen instead of Enc, the backdoor decryption algorithm becomes a backdoor key recovery algorithm (Alg. 3) in our scheme. We embed the ciphertext of seed d, which is used to generate the secret key of Kyber, into

input : mc.pk
output: $pk \leftarrow (\mathbf{t}, pk.seed)$, $sk \leftarrow \mathbf{s}$

Function $\texttt{KeyGen}^*(\text{mc}.pk)$**:**
- $(K, C) \leftarrow \text{mc.Encap}(\text{mc}.pk)$
- $d \leftarrow K$ // Let the seed in Kyber be the session key of McEliece.
- $(sk.seed, pk.seed) \leftarrow G(d)$ //Function G is declared in Kyber
- $(\mathbf{s}, \_) \leftarrow \mathsf{PRF}(sk.seed)$ //Sample $\mathbf{s}$ from $sk.seed$ in distribution B_η
- $\mathbf{A} \leftarrow \mathsf{Parse}(\mathsf{XOF}(pk.seed))$ //Sample $\mathbf{A}$ from $pk.seed$.
- $\mathbf{t} \leftarrow \mathbf{As}$;
- **for** i **from** 1 **to** $\dim(\mathbf{t})$ **do**
  - **if** $i \leq \text{len}(C)$ **then**
    - **if** $(\mathbf{t}[i] - C[i]) \mod 2 = 1$ **then**
      - Sample e_i from the probabilistic distribution $\mathcal{D}_1$
    - **else**
      - Sample e_i from the probabilistic distribution $\mathcal{D}_0$
  - **else**
    - Sample e_i from the probabilistic distribution B_2
  - $\mathbf{t}[i] \leftarrow \mathbf{t}[i] + e_i \mod^{\pm} q$
- **return** $pk \leftarrow (\mathbf{t}, pk.seed)$, $sk \leftarrow \mathbf{s}$

Algorithm 2: Backdoor Key Generation Algorithm KeyGen^*

the LSBs of $\mathbf{t}$ in public key of Kyber. Suppose the backdoor user has mc.sk, then he can decrypt the seed d after receiving $pk = (\mathbf{t}, pk.seed)$ through implementing $d' = \text{mc.Decap}(\text{mc}.sk, \text{LSBs}(\mathbf{t}))$. He can use d' to regenerate the secret key seed $sk.seed'$ and public key $pk.seed'$ seed by calling G function. Since we know the correct $pk.seed$, we can verify $\mathbf{s}' \stackrel{?}{=} \mathbf{s}$ by determining whether $pk.seed' \stackrel{?}{=} pk.seed$. However, the backdoor user cannot always get a correct ciphertext C from computing the LSBs of $\mathbf{t}$ since there are border cases where $C_i \neq \text{LSB}(t_i)$. We just use exhaustive search to handle these border cases.

Discussion on the border case. We note that the existence of border cases where $\text{LSB}(t_i)$ disagrees with C_i is necessary for the undetectability of our backdoor. If there is no border case, which means that $\text{LSB}(t_i)$ follows exactly the same distribution as C_i, that is, uniform on $\{0, 1\}$. However, the real t_i is indistinguishable from uniform in $\mathbb{Z}_q$, and for $q = 3329$ as in Kyber, $\Pr(\text{LSB}(t_i) = 0) = 1665/3329 = 1/2 + 1/6658$. Since the two distributions are different, given sufficiently many public keys, it should be easy to determine whether the keys are backdoored by statistical approaches, such as Special Publication 800-22 Revision 1a of NIST [RSN+10], the Diehard tests [Bro], the TestU01 [LS07] or the popular test suite AIS20/31 [PMB+16]. In [RBC+24], the authors did not handle the border cases and simply let $\text{LSB}(t_i) = C_i$, so in contrast with their claim, their backdoors cannot satisfy even public undetectability.

We can see that $\text{LSB}(t_i)$ and C_i disagree only when $t_i \in \{-(q-1)/2, -(q-3)/2, (q-3)/2, (q-1)/2\}$, which has a probability of $p = 4/q$. For $q = 3329$ in

input : $pk \leftarrow (\mathbf{t}, pk.seed)$, mc.$sk$, $\eta \leftarrow 2$
output: $sk \leftarrow \mathbf{s}$ or $\perp$

Function $\texttt{KeyRec}^*(pk)$**:**
  Sample $\mathbf{A}$ from $pk.seed$;
  $C' \leftarrow \text{LSBs}(\mathbf{t})$, mark $C'[i] = \star$ if $\mathbf{t}[i] \geq (q-3)/2$ or $\mathbf{t}[i] \leq -(q-3)/2$;
  repeat
    Set $C'[i] = \star$ to 0 or 1 respectively;
    $d' \leftarrow \text{mc.Decap}(\text{mc}.sk, C')$;
    $(sk.seed', pk.seed') \leftarrow G(d')$;
    if $pk.seed' = pk.seed$ **then**
      $(\mathbf{s}', \_) \leftarrow \mathsf{PRF}(sk.seed')$; //Sample $\mathbf{s}'$ from $sk.seed'$ through pseudorandom function PRF
      return $sk \leftarrow \mathbf{s}'$;
  until *Exhaust all possibilities of* $C'[i] = \star$;
  return $\perp$*;*

Algorithm 3: Backdoor Key Recovery Algorithm KeyRec^*

Kyber, the probability that i border case elements occurrence is

$$\mathrm{P}_{\text{theo}} = \Pr(i \text{ border case elements in } (t_1, \cdots, t_m)) = C_m^i p^i (1-p)^{m-i}, \quad (1)$$

where $m = \text{len}(C) = 768$ is the length(bitsize) of C. It infers that the probability that there are more than 4 border case elements is only about 0.2%. We also note that not every border case leads to the disagreement between $\text{LSB}(t_i)$ and C_i. From our experiment in Section 4.1, we can see that the key recovery is efficient in most cases.

3.3 Proof of Undetectability

In this section, before proving the undetectability of the backdoor in our scheme, we first need to establish that the error term $\mathbf{e}$ in KeyGen and KeyGen* is identically distributed. From the description of Kyber.KeyGen (Algorithm 1), we know that each coefficient e_i of the noise vector $\mathbf{e}$ follows the B_2 distribution as shown in Table 2. Therefore, we only need to demonstrate that each coefficient in $\mathbf{e}$ generated by Kyber.KeyGen* (Algorithm 2) also follows the same B_2 distribution as Kyber.KeyGen. We provide Lemma 1 to support this conclusion.

Subsequently, we present Theorem 1. It proves that our backdoor scheme is strictly undetectable by the definition of strict undetectability, Lemma 1 and the hardness of decisional SDP.

Table 2. Error distribution B_2 in Kyber.KeyGen

value	-2	-1	0	1	2
distribution	1/16	1/4	3/8	1/4	1/16

Lemma 1. *If C is uniformly distributed and independent with $\mathbf{A}, \mathbf{s}$, then the distribution of $\mathbf{e}$ generated from Algorithm 2 is also independent with $\mathbf{A}, \mathbf{s}$, and identical with random $\mathbf{e}$ where each coefficient of $\mathbf{e}$ is randomly sampled from B_2.*

Proof. Since C is uniform and independent with $\mathbf{A}, \mathbf{s}$, the probability $\Pr(C_i = [\mathbf{As}]_i) = 1/2$ and independent with $\mathbf{As}$. By Algorithm 2, we can see that $\Pr(e_i | C_i = [\mathbf{As}]_i)$ and $\Pr(e_i | C_i \neq [\mathbf{As}]_i)$ satisfies the distribution in Table 3, also independent with $\mathbf{A}, \mathbf{s}$.

Table 3. Error distribution in Kyber.KeyGen* (assume that C is uniform)

value	-2	-1	0	1	2
distribution when $C_i = [\mathbf{As}]_i$	1/8	0	3/4	0	1/8
distribution when $C_i \neq [\mathbf{As}]_i$	0	1/2	0	1/2	0
distribution in average	1/16	1/4	3/8	1/4	1/16

Thus, we have: $\Pr(e_i) = \Pr(e_i | C_i = [\mathbf{As}]_i) \Pr(C_i = [\mathbf{As}]_i) + \Pr(e_i | C_i \neq [\mathbf{As}]_i) \Pr(C_i \neq [\mathbf{As}]_i) = (\Pr(e_i | C_i = [\mathbf{As}]_i) + \Pr(e_i | C_i \neq [\mathbf{As}]_i))/2$. As shown in Table 2 and Table 3, the distribution of e_i is identical to $e_i \leftarrow B_2$. Thus, we finish the proof. □

Theorem 1. *The backdoor scheme is strictly undetectable under the hardness of decisional SDP problem for Goppa codes.*

Proof. Let Game 0 be the strict undetectability game (Definition 3) where $\mathcal{C}$ runs Kyber. KeyGen*. We define the following sequence of games:

Game 1: $\mathbf{s}$ is generated as truly random from the distribution B_2 instead of $\mathsf{PRF}(sk.seed)$, and $pk.seed$ is generated as a truly random bit string. Since $sk.seed$ and $d = K$ are never revealed to $\mathcal{D}$, Game 0 and Game 1 are indistinguishable by the pseudorandomness of PRF and G (which is a cryptographic hash function, see the definition of Kyber [ABD+20]).

Game 2: C is changed into a truly random bit string which independent with $\mathbf{A}, \mathbf{s}$. We show that if there is a detector $\mathcal{D}$ which distinguishes between Game 1 and Game 2 with a non-negligible advantage, then there exists an algorithm that solves decisional SDP for Goppa codes with a non-negligible advantage.

Let $\mathcal{C}$ be the challenger which $\mathcal{D}$ interacts within the strict undetectability game. $\mathcal{C}$ is given a sample of the decisional SDP problem $\mathbf{H} = (\mathbf{I}|\mathbf{T})$ and C, where T is the McEliece public key (thus $\mathbf{H}$ represents a random Goppa code) and C is either a uniform random bit string or $C = \mathbf{Hv} \mod 2$ and $\mathrm{wt}(\mathbf{v}) = t$. Then $\mathcal{C}$ continues to run the backdoored key generation algorithm Kyber.KeyGen* with embedded string C as in Game 1, and interacts with $\mathcal{D}$ normally.

We can see that if $C = \mathbf{Hv}$, then C is a random McEliece ciphertext and $\mathcal{C}$ acts as the challenger in Game 1. If C is truly random, then $\mathcal{C}$ acts as the challenger in Game 2. So if $\mathcal{D}$ distinguishes between Game 1 and Game 2 with a

non-negligible advantage, then $\mathcal{C}$ can solve decisional SDP with a non-negligible advantage. So Game 1 and Game 2 are computationally indistinguishable.

Game 3: $\mathcal{C}$ runs Kyber.KeyGen, the real key generation algorithm of Kyber, except that $pk.seed$ is a truly random bit string and $\mathbf{s}, \mathbf{e}$ are generated as truly random from the distribution B_2. Since C is truly random and independent with $\mathbf{A}, \mathbf{s}$, by Lemma 1, the distribution of $\mathbf{s}, \mathbf{e}$ in Game 2 and Game 3 are identical, so Game 2 and Game 3 are the same from the view of $\mathcal{D}$.

Game 4: $\mathcal{C}$ uses a truly random seed d to generate $(sk.seed, pk.seed) \leftarrow G(d)$, and $\mathbf{s}, \mathbf{e}$ are generated as $\mathsf{PRF}(sk.seed)$. Game 3 and Game 4 are indistinguishable due to the pseudorandomness of PRF and G. Here Game 4 is the strict undetectability game where $\mathcal{C}$ runs Kyber.KeyGen.

Having established the indistinguishability of Game 0 and Game 4, we conclude the proof. □

4 Experimental Results

Our backdoor embedding method has been implemented in the C language[1]. All experiments were conducted on a single core (Intel(R) Core(TM) i5-9500 CPU @ 3.00GHz).

4.1 Efficiency Test of KeyGen* and KeyRec*

The median and average costs of KeyGen* (Sec. 3.1) and KeyRec* (Sec. 3.2) over 1000 trials are summarized in Table 4. The results indicate that the cost of KeyGen* is comparable to that of the original Kyber KeyGen algorithm (all running times are lower than 1 ms per trial), making it difficult for detectors to identify the backdoor solely based on the time cost of the Key Generation Algorithm. Furthermore, the time required to recover the secret key using KeyRec* with the backdoor is approximately 1 ms on a personal desktop computer, demonstrating the efficiency of the backdoor holder in recovering the secret key through the backdoor. Furthermore, we achieved a 100% success rate in recovering the Kyber secret key.

Table 4. The median and average costs of KeyGen (Alg. 1), KeyGen*(Sec. 3.1) and KeyRec* (Sec. 3.2). Each experimental result is median or averaged over 1000 instances.

Scheme	Cost Type	KeyGen	KeyGen*	KeyRec*
Kyber768	Median Cost(cycles/tick)	28397	115590	166088
	Average Cost(cycles/tick)	36207	118271	169267
Kyber1024	Median Cost(cycles/tick)	39636	133840	191503
	Average Cost(cycles/tick)	48604	135736	194552

[1] https://github.com/Summwer/kyber-backdoor

4.2 Border Case Test

In Sec. 3.2 we give a theoretical analysis of border case probability $\mathrm{P}_{\mathrm{theo}}$ (Eq. (1)), to show its accuracy we test 1000 instances of Kyber768 (the result of Kyber1024 is close to Kyber768 since the bit size of McEliece ciphertext is same) in Fig. 2. Assume there are i border case elements among $(t_1, \cdots, t_m)$, where $m = \mathrm{len}(C)$. The actual border case probability can be computed as $\mathrm{P}_{\mathrm{actual}}(i) = \frac{i \text{ border case elements occur in } (t_1, \cdots, t_m)}{1000}$. In Fig. 2, it shows that the accuracy of $\mathrm{P}_{\mathrm{theo}}$ fits well to $\mathrm{P}_{\mathrm{actual}}(i)$. Besides, we can also see in Fig. 2 that the border case probability decreases rapidly with the growth of i. As the maximal enumeration times for finding the correct elements in border case is 2^i, it takes an acceptable low cost in small i as shown in the cost of KeyRec* of Table 4.

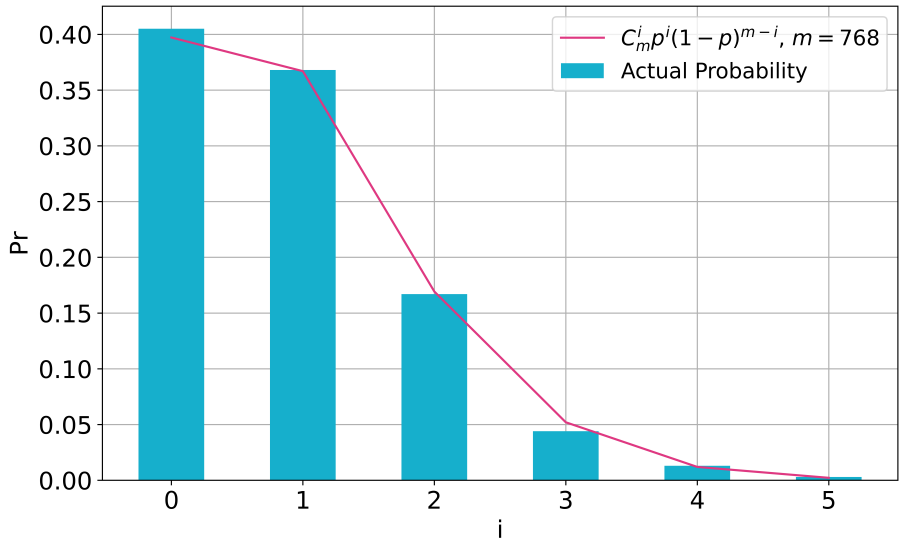

Fig. 2. Border case probability among $m = 768$ elements. Here the x-axis is the number of border case elements among m elements. The blue blocks are the experimental border case probability among 1000 Kyber768 instances, it can be computed as $\mathrm{P}_{\mathrm{actual}}(i) = \frac{i \text{ border case elements occur in } (t_1, \cdots, t_m)}{1000}$. The pink line is the theoretical border case probability declared in Eq. (1).

5 Discussion

5.1 Possible Fixes

In [Hem22], the author gave a possible fix for the [YXP20] type backdoor, which also works for our backdoor. We simply describe the method.

Note that in Kyber.KeyGen, both $pk.seed$ and the secret key are generated from a single seed d in PRF. We only need to include d in the secret key. The secret

key holder firstly generates $pk.seed$ and $sk.seed$ from d, then computes $\mathbf{A} = \mathsf{Parse}(\mathsf{XOF}(pk.seed))$, $(\mathbf{s}, \mathbf{e}) = \mathsf{PRF}(sk.seed)$. Then, the secrete key holder checks $\mathbf{As} + \mathbf{e} = \mathbf{t} \bmod^{\pm} q$. If the check fails, then the algorithm might be backdoored.

We note that such a fixing method can detect any backdoor hidden in the KeyGen algorithm, and we suggest that the fix should be applied to the Kyber standard to avoid potential risks.

5.2 Discussion on Public Undetectablity

We note that the fix above aims at strict undetectability, but not public undetectability. It is easy to see that both the [YXP20] type backdoor and our backdoor still have public undetectability even when the fix is applied to Kyber. Before further discussion, we give another fix for [YXP20] type backdoors which can invalidate their public undetectability but does not affect our backdoor. The idea came from the discussion of backdoor resistance in NewHope [ADPS16].

Let crs be a uniform common reference string, which is generated from trusted methods such as MPC protocols. Let H be a cryptographic hash function, and each user public seed is generated by $pk.seed = H(crs\|id)$, id is the identity of the user. Since $pk.seed$ is the output of a hash function, it cannot be the encoding of a point on the elliptic curve. However, our trapdoor does not modify $pk.seed$, hence is not affected.

If our backdoor becomes publicly detectable after a certain fix, it means that a detector without having the secret key can know whether the secret key is correctly generated. However, by the security of the original Kyber, no information about the secret key can be extracted from the public key, so if the fix maintains the security claim of Kyber, a zero-knowledge proof must be contained in the public key to show the correctness of private key generation. We note that the zero-knowledge proof must prove a relation on both hash functions and polynomial operations, and to the best of our knowledge, it takes several hundreds of MB to construct a post-quantum proof for such relations. This would result in a significant increase in public key size, conflicting with the desire for small-sized keys in signature schemes.

5.3 Backdoor on Other LWE-based Schemes

By the construction and undetectability proof of our backdoor, we can see that to plant our backdoor in an LWE-type scheme, the following three conditions are required for our backdoor to be valid:

(1) The LSBs of the LWE error term in the public key must be uniform;

(2) The number of coefficients in the public key vector must be equal or larger than the McEliece ciphertext size (at least 768);

(3) The LSBs in the public key are not compressed (which means that our backdoor cannot be applied to Dilithium).

While condition (1) may appear to be a stringent requirement, we demonstrate that it is easily achievable. For practical implementation convenience, the

LWE error is often modeled as a central binomial distribution rather than a discrete Gaussian. Under this assumption, we can establish that condition (1) can be met through Lemma 2.

Lemma 2. *Let $e \leftarrow B_\eta$ be sampled from a central binomial distribution with parameter η. Then* $\Pr(\mathrm{LSB}(e) = 0) = 1/2$.

Proof. By the definition of the central binomial distribution (Def. 4), $e \leftarrow B_\eta$ can be sampled by the algorithm below:

(1) Sample $a_i, b_i \leftarrow \{0, 1\}$, $i = 1, ..., \eta$;

(2) Calculate $e = \sum_{i=1}^{\eta}(a_i - b_i)$.

From the given $a_2, \ldots, a_\eta, b_1, \ldots, b_\eta$, we observe that $\mathrm{LSB}(e)$ differs for $a_1 = 0$ and $a_1 = 1$, implying that the conditional probability

$$\Pr(\mathrm{LSB}(e) = 0 | a_2, \ldots, a_\eta, b_1, \ldots, b_\eta) = 1/2$$

for any $a_2, \ldots, a_\eta, b_1, \ldots, b_\eta$. Thus, $\Pr(\mathrm{LSB}(e) = 0) = 1/2$.

Condition (2) is the main obstacle to implementing our backdoor, and this is the reason why our backdoor cannot be applied to Kyber-512. Similarly, our backdoor cannot be applied to NewHope-512.

Since all the listed conditions are satisfied for NewHope-1024, Frodo-640, and Frodo-976, our backdoor can be applied to these schemes. However, for Frodo-1344, $\Pr(\mathrm{LSB}(e_i) = 0) = 1/2 + 2^{-15}$, which is not uniform and violates condition (1). As a result, the strict undetectability proof of our backdoor fails for this case. Nonetheless, given the small deviation, detecting the backdoor would require knowledge of several hundred secret keys, making it still challenging to detect in practice.

6 Conclusion and Future Works

In this work, we give the first post-quantum provably undetectable backdoor for post-quantum KEM Kyber-768 and Kyber-1024. We use the post-quantum code-based KEM Classic McEliece to encapsulate the Kyber secret seed into the LSBs of the Kyber public key, so the attacker with Classic McEliece secret key can recover the Kyber secret seed, hence the Kyber secret key. We note that earlier works either cannot be applied to IND-CCA2 secure KEM or use pre-quantum backdoor schemes. Moreover, we show that our backdoor maintains public undetectability against some possible fixes.

Our backdoor can also be applied to NewHope-1024, Frodo-640, and Frodo-976. We leave as future work whether our backdoor can be modified for other post-quantum algorithms or we can use Classic McEliece with a higher security level as the backdoor scheme to achieve higher undetectability.

Acknowledgments This work was supported by the National Natural Science Foundation of China No. U2336210.

References

ABC+22. Roberto Avanzi, Daniel J Bernstein, Tung Chou, Tanja Lange, Ingo von Maurich, Rafael Misoczki, Ruben Niederhagen, Edoardo Persichetti, Christiane Peters, Peter Schwabe, Nicolas Sendrier, et al. Classic mceliece: conservative code-based cryptography. *NIST PQC Round 4 submissions*, 2022.

ABD+20. Roberto Avanzi, Joppe Bos, Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, John M Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-kyber algorithm specifications and supporting documentation (version 3.0). *NIST PQC Round 3 submissions*, 2020.

ADPS16. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In Thorsten Holz and Stefan Savage, editors, *25th USENIX Security Symposium, USENIX Security 16, Austin, TX, USA, August 10-12, 2016*, pages 327–343. USENIX Association, 2016.

BHKL13. Daniel J. Bernstein, Mike Hamburg, Anna Krasnova, and Tanja Lange. Elligator: elliptic-curve points indistinguishable from uniform random strings. In Ahmad-Reza Sadeghi, Virgil D. Gligor, and Moti Yung, editors, *2013 ACM SIGSAC Conference on Computer and Communications Security, CCS'13, Berlin, Germany, November 4-8, 2013*, pages 967–980. ACM, 2013.

BMvT78. Elwyn R. Berlekamp, Robert J. McEliece, and Henk C. A. van Tilborg. On the inherent intractability of certain coding problems (corresp.). *IEEE Trans. Inf. Theory*, 24(3):384–386, 1978.

Bro. Robert G. Brown. Dieharder: A random number test suite. https://webhome.phy.duke.edu/~rgb/General/dieharder.php.

Hem22. Tobias Hemmert. How to backdoor lwe-like cryptosystems. *IACR Cryptol. ePrint Arch.*, page 1381, 2022.

KLT17. Robin Kwant, Tanja Lange, and Kimberley Thissen. Lattice klepto - turning post-quantum crypto against itself. In Carlisle Adams and Jan Camenisch, editors, *Selected Areas in Cryptography - SAC 2017 - 24th International Conference, Ottawa, ON, Canada, August 16-18, 2017, Revised Selected Papers*, volume 10719 of *Lecture Notes in Computer Science*, pages 336–354. Springer, 2017.

LS07. Pierre L'Ecuyer and Richard Simard. TestU01: A C library for empirical testing of random number generators. *ACM Trans. Math. Softw.*, 33(4):22:1–22:40, August 2007.

PMB+16. Oto Petura, Ugo Mureddu, Nathalie Bochard, Viktor Fischer, and Lilian Bossuet. A survey of AIS-20/31 compliant TRNG cores suitable for FPGA devices. In *2016 26th International Conference on Field Programmable Logic and Applications (FPL)*, pages 1–10, August 2016. ISSN: 1946-1488.

RBC+24. Prasanna Ravi, Shivam Bhasin, Anupam Chattopadhyay, Aikata Aikata, and Sujoy Sinha Roy. Backdooring post-quantum cryptography: Kleptographic attacks on lattice-based kems. In *Proceedings of the Great Lakes Symposium on VLSI 2024*, GLSVLSI '24, page 216–221, New York, NY, USA, 2024. Association for Computing Machinery.

RSN+10. Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, N. Heckert, James Dray, San Vo, and Lawrence Bassham. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications. Technical Report NIST Special Publication (SP) 800-22 Rev. 1, National Institute of Standards and Technology, April 2010.

XY18. Dianyan Xiao and Yang Yu. Klepto for ring-lwe encryption. *Comput. J.*, 61(8):1228–1239, 2018.

YCL+20. Zhichao Yang, Rongmao Chen, Chao Li, Longjiang Qu, and Guomin Yang. On the security of LWE cryptosystem against subversion attacks. *Comput. J.*, 63(4):495–507, 2020.

YXP20. Zhaomin Yang, Tianyuan Xie, and Yanbin Pan. Lattice klepto revisited. In Hung-Min Sun, Shiuh-Pyng Shieh, Guofei Gu, and Giuseppe Ateniese, editors, *ASIA CCS '20: The 15th ACM Asia Conference on Computer and Communications Security, Taipei, Taiwan, October 5-9, 2020*, pages 867–873. ACM, 2020.

YY97a. Adam L. Young and Moti Yung. Kleptography: Using cryptography against cryptography. In Walter Fumy, editor, *Advances in Cryptology - EUROCRYPT '97, International Conference on the Theory and Application of Cryptographic Techniques, Konstanz, Germany, May 11-15, 1997, Proceeding*, volume 1233 of *Lecture Notes in Computer Science*, pages 62–74. Springer, 1997.

YY97b. Adam L. Young and Moti Yung. The prevalence of kleptographic attacks on discrete-log based cryptosystems. In Burton S. Kaliski Jr., editor, *Advances in Cryptology - CRYPTO '97, 17th Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 1997, Proceedings*, volume 1294 of *Lecture Notes in Computer Science*, pages 264–276. Springer, 1997.

YY16. Adam L. Young and Moti Yung. Cryptography as an attack technology: Proving the rsa/factoring kleptographic attack. In Peter Y. A. Ryan, David Naccache, and Jean-Jacques Quisquater, editors, *The New Codebreakers - Essays Dedicated to David Kahn on the Occasion of His 85th Birthday*, volume 9100 of *Lecture Notes in Computer Science*, pages 243–255. Springer, 2016.