SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

August 2020, Vol. 63 182101:1–182101:22
https://doi.org/10.1007/s11432-019-9861-3

# Improved lattice-based CCA2-secure PKE in the standard model

Jiang ZHANG[1,2*], Yu YU[3], Shuqin FAN[1] & Zhenfeng ZHANG[4]

[1]*State Key Laboratory of Cryptology, P.O. Box* 5159*, Beijing* 100878*, China;*
[2]*Guangdong Provincial Key Laboratory of Data Security and Privacy Protection, Jinan University, Guangzhou* 510632*, China;*
[3]*Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai* 200240*, China;*
[4]*Trusted Computing and Information Assurance Laboratory, Institute of Software, Chinese Academy of Sciences, Beijing* 100190*, China*

Received 12 February 2019/Accepted 4 April 2019/Published online 7 July 2020

**Abstract** Based on the identity-based encryption (IBE) from lattices by Agrawal et al. (Eurocrypt'10), Micciancio and Peikert (Eurocrypt'12) presented a CCA1-secure public-key encryption (PKE), which has the best known efficiency in the standard model and can be used to obtain a CCA2-secure PKE from lattices by using the generic BCHK transform (SIAM J Comput, 2006) with a cost of introducing extra overheads to both computation and storage for the use of other primitives such as signatures and commitments. In this paper, we propose a more efficient standard model CCA2-secure PKE from lattices by carefully combining a different message encoding (which encodes the message into the most significant bits of the LWE's "secret term") with several nice algebraic properties of the tag-based lattice trapdoor and the LWE problem (such as unique witness and additive homomorphism). Compared to the best known lattice-based CCA1-secure PKE in the standard model due to Micciancio and Peikert (Eurocrypt'12), we not only directly achieve the CCA2-security without using any generic transform (and thus do not use signatures or commitments), but also reduce the noise parameter roughly by a factor of 3. This improvement makes our CCA2-secure PKE more efficient in terms of both computation and storage. In particular, when encrypting a 256-bit (respectively, 512-bit) message at 128-bit (respectively, 256-bit) security, the ciphertext size of our CCA2-secure PKE is even 33%–44% (respectively, 36%–46%) smaller than that of their CCA1-secure PKE.

**Keywords** lattice, public-key encryption, chosen ciphertext security, standard model

**Citation** Zhang J, Yu Y, Fan S Q, et al. Improved lattice-based CCA2-secure PKE in the standard model. Sci China Inf Sci, 2020, 63(8): 182101, https://doi.org/10.1007/s11432-019-9861-3

## 1 Introduction

In the seminal work of [1], Diffie and Hellman introduced the concept of public-key cryptography. Soon afterwards, Rivest, Shamir and Adleman [2] proposed the first public-key encryption (PKE), which is known as RSA. Since then, PKE has aroused widespread public attention from the community, and has become one of the most fundamental and widely used cryptographic primitives. The basic security notion for PKE (i.e., CPA-security) which was formalized by Goldwasser and Micali [3] roughly requires that it should be computationally infeasible for a passive adversary to obtain any useful information from a honestly generated challenge ciphertext. Later, this notion was enhanced by Naor and Yung [4] to deal with the "lunchtime attack". Specifically, they [4] considered the security against non-adaptive chosen ciphertext attacks (i.e., CCA1-security) for PKE, where the adversary can access a decryption oracle to

* Corresponding author (email: jiangzhang09@gmail.com)

 

decrypt any ciphertext of his choice before seeing the challenge ciphertext. Now, the de facto standard security for PKE is CCA2-security [5], where the adversary can adaptively access the decryption oracle during the whole attack period (with a restriction that the decryption oracle cannot be directly used to decrypt the challenge ciphertext). For example, the National Institute of Standards and Technology (NIST) considered CCA2-security as a basic security requirement for the PKE submissions to the post-quantum cryptography (PQC) standardization [6].

By definition, CCA2-security is stronger than CCA1-security, which in turn is stronger than CPA-security. One of the main problems in this area is to construct CCA2-secure PKEs from primitives as weak as possible (e.g., a CPA-secure one). Using the Random Oracle (RO) heuristic, one can efficiently boost a CPA-secure PKE into a CCA2-secure one [7–9]. However, a scheme provably secure in the RO model may not be secure in the real world [10], and it is of great theoretical and practical interest to construct CCA2-secure PKE in the standard model. But this task becomes very challenging and highly non-trivial. In fact, Gertner et al. [11] showed that it is hard, if not impossible, to even construct a CCA1-secure PKE solely from a CPA-secure one in the standard model.

By relying on primitives with "stronger" functionality or security, there are roughly four approaches to CCA2-secure PKEs in the standard model. The first one is due to Naor and Yung [4], who showed a paradigm for transforming CPA-secure PKEs into CCA1-secure ones by using the non-interactive zero-knowledge (NIZK) proofs, which was further extended to achieve CCA2-security [12,13]. The second one is a framework under the name of hash proof systems (HPS) or extractable HPS [14,15], which essentially stems from high-level abstraction of some existing schemes. The third one is the BCHK transform [16] from identity-based encryption (IBE), which was later extended to the more general tag-based encryption (TBE) by Kiltz [17]. The last one follows the generic constructions from special types of injective trapdoor functions [18–20], such as lossy trapdoor functions [18] and adaptive trapdoor functions [20].

The above approaches have been shown very useful in constructing CCA2-secure PKEs from various hardness assumptions, but most of the instantiations were based on traditional number theoretic problems such as discrete logarithm and integer factorization, which are not quantum resistant [21]. Compared to the big success in the traditional setting, the progress on designing lattice-based CCA2-secure PKE in the standard model was relatively slow. For example, many practical CCA2-secure PKEs in the traditional setting were obtained by using the generic framework from HPS (e.g., [22]), but it is still hard to construct an HPS from lattices [23–26]. Moreover, it is also unclear how to obtain NIZKs from lattices in the standard model [25,27,28]. This means that we currently cannot use the first two approaches to construct standard model CCA2-secure PKEs from lattices.

In fact, almost all existing standard model CCA2-secure PKEs from lattices are, to the best of our knowledge, obtained either by using the techniques from special types of injective trapdoor functions [18,29–31] which are typically very inefficient (e.g., having large public key and ciphertext sizes due to the use of Dolev-Dwork-Naor like technique [13,32]), or by applying the BCHK transform from IBEs/TBEs [33–39]. Based on the standard model IBE from lattices due to Agrawal et al. [33], Micciancio and Peikert [40] presented the best known standard model (tag-based) CCA1-secure PKE from lattices by using a more efficient trapdoor technique and a new message encoding. They [40] also mentioned that the CCA2-security can be achieved by using the generic BCHK transform [16], which has two modes: BCHK-SIG [41] and BCHK-MAC [42]. BCHK-SIG requires (one-time) signatures, and typically incurs noticeable overheads to both computation and storage [16]. This becomes even worse on lattices, since the resulting ciphertext should include a verification key of the (one-time) signature, which, to the best of our knowledge, has at least one matrix [43]. In contrast, BCHK-MAC [42] makes use of message authentication codes (MAC) and commitments, and thus can reduce the extra overheads, e.g., it only adds a MAC tag and a commitment to the resulting ciphertext.

Given the difficulties in adapting traditional techniques to the lattice setting, and the insufficiencies of existing CCA2-secure PKEs from lattices, it is natural to ask: Can we directly construct a standard model CCA2-secure PKE from lattices (possibly by carefully exploiting the rich algebraic properties of lattices), such that it has better performance than those following generic approaches?

**Table 1** A concrete comparison of the ciphertext size of our CCA2-secure PKE and the CCA1-secure one in [40] for encrypting a 256-bit (respectively, 512-bit) message at 128-bit (respectively, 256-bit) security

| Scheme | LWE parameter $(n, m, q, \alpha q)$ | Ciphertext size | Decryption error | Security strength |
|---|---|---|---|---|
| MP12 [40] | $(450, 11905, 3^{10}, 2.45)$ | 24.74 KB (20.89 KB$^{\dagger}$) | $2^{-88}$ | $2^{128}$ |
| | $(660, 19041, 3^{11}, 5.2)$ | 44.19 KB (37.10 KB$^{\dagger}$) | $2^{-115}$ | $2^{257}$ |
| This paper | $(450, 10740, 3^{9}, 1.5)$ | **13.80 KB** (33%–44% $\downarrow$) | $\mathbf{2^{-100}}$ | $\mathbf{2^{131}}$ |
| | $(660, 17333, 3^{10}, 3.1)$ | **23.47 KB** (36%–46% $\downarrow$) | $\mathbf{2^{-138}}$ | $\mathbf{2^{260}}$ |

### 1.1 Our results

In this paper, we construct a standard model CCA2-secure PKE from lattices, which does not follow the generic approaches mentioned above. Technically, our PKE is obtained by carefully combining a different message encoding with several nice algebraic properties of the tag-based lattice trapdoor and the learning with errors (LWE) problem (e.g., unique witness and additive homomorphism). Unlike previous LWE-based PKEs which typically encode the message into the LWE's "error term", we encode the message into the most significant bits of the LWE's "secret term", which not only provides a better way to control the error size in the decryption, but also allows us to directly achieve CCA2-security. Compared to the best known standard model CCA1-secure PKE [40], our CCA2-secure PKE reduces the noise parameter for encryption roughly by a factor of 3, and thus improves the efficiency in both computation and storage. In the supplemental material, we also extend our techniques to a special type of rings and obtain an efficient CCA2-secure PKE from ring-LWE in the standard model.

In Table 1, we give a concrete comparison of the ciphertext size of our CCA2-secure PKE and the best known standard model CCA1-secure PKE from lattices [40] (note that the authors of [40] only mentioned to achieve CCA2-security by using generic approaches such as the BCHK transform [16]). For better efficiency, we set $q$ as a power of 3 for both schemes. The LWE parameter $(n, m, q, \alpha q)$ was chosen by taking account of the probability of decryption error and the security strength, where $n$ is the LWE dimension, $m$ is the number of LWE samples, $q$ is the LWE modulus and $\alpha q$ is the LWE Gaussian parameter. Since the noise parameter of our encryption algorithm is equal to the underlying LWE Gaussian parameter $\alpha q$ and that of their encryption algorithm is about 3 times larger than $\alpha q$ (as we will discuss later), we have to choose different parameter sets to roughly achieve the same degree of correctness and security. The concrete decryption error and security strength for the parameter sets given in Table 1 are estimated by using a Python script and the online LWE estimator [44], respectively. The ciphertext sizes at rows 1 and 3 (respectively, rows 2 and 4) correspond to encrypting a 256-bit message at 128-bit security (respectively, a 512-bit message at 256-bit security), where '$\downarrow$' means reduction of ciphertext sizes. We also apply known ciphertext compressing techniques to the PKE in [40], and estimate the final ciphertext sizes (marked by '$\dagger$')[1)]. Since the key sizes and the computational costs of both schemes are essentially dominated by the LWE parameter $(n, m, q, \alpha q)$, our PKE also has advantages in these two aspects due to the smaller $m$ and $\alpha q$. For example, the public key size of the PKE in [40] for 128-bit security is about 3.86 MB, while ours for 131-bit security is about 3.26 MB (i.e., a 15% reduction). In all, our CCA2-secure PKE is more efficient than the CCA1-secure PKE in [40].

### 1.2 Overview of techniques

As the lattice-based CCA2-secure PKEs in the standard model [29, 30, 33, 40], our scheme also relies on the lattice trapdoor technique, which dates back to the seminal work of Ajtai [45]. In 2008, Gentry et al. [46] first showed how to combine the lattice trapdoor technique and the LWE problem in designing encryption schemes. Technically, they [46] introduced the dual variant of the first LWE-based encryption due to Regev [47], which is nicely compatible with the trapdoor technique. Briefly, the public key of the dual encryption consists of two matrices $\mathsf{pk} = (\boldsymbol{A}, \boldsymbol{U}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times \ell}$. In order to encrypt a message $\mu \in \{0,1\}^{\ell}$, the encryption algorithm randomly chooses $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n, \boldsymbol{e}_1 \xleftarrow{\$} D_{\mathbb{Z}^m,\alpha q}, \boldsymbol{e}_2 \xleftarrow{\$} D_{\mathbb{Z}^{\ell},\alpha q}$,

1) Note that Ref. [40] actually did not give a concrete choice of parameters, and did not consider any ciphertext compressing technique.

and computes the ciphertext $C=(\boldsymbol{c}_1,\boldsymbol{c}_2)$, where $D_{\mathbb{Z}^m,\alpha q}$ is the Gaussian distribution with parameter $\alpha q$ (and we call $\alpha q$ the noise parameter for encryption), and

$$\boldsymbol{c}_1=\boldsymbol{A}^{\mathrm{T}}\boldsymbol{s}+\boldsymbol{e}_1\in\mathbb{Z}_q^m,\qquad \boldsymbol{c}_2=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{s}+\boldsymbol{e}_2+\mu\cdot\frac{q}{2}\in\mathbb{Z}_q^{\ell}.$$

As Regev's LWE-based PKE [47], the message $\mu\in\{0,1\}^{\ell}$ is encoded into the most significant bits of the LWE's "error term". In this case, the secret key can either be a small norm matrix $\boldsymbol{E}\in\mathbb{Z}^{m\times\ell}$ satisfying $\boldsymbol{AE}=\boldsymbol{U}$, or a trapdoor of the matrix $\boldsymbol{A}$ which can be used to extract a required matrix $\boldsymbol{E}$, so that one can recover the message from the most significant bits of the noise vector $\boldsymbol{c}_2-\boldsymbol{E}^{\mathrm{T}}\boldsymbol{c}_1=\mu\cdot\frac{q}{2}+\boldsymbol{e}_2-\boldsymbol{E}^{\mathrm{T}}\boldsymbol{e}_1$ as long as the $l_\infty$ norm $\|\boldsymbol{e}_2-\boldsymbol{E}^{\mathrm{T}}\boldsymbol{e}_1\|_\infty<q/4$.

The dual encryption [46] was latter employed by Peikert [29] to construct a standard model CCA2-secure PKE from lattices with similar techniques in spirit to the ones in [18, 19], and by Agrawal et al. [33] to construct an efficient IBE which can be transformed into a CCA2-secure by using the generic approaches such as the BCHK transform [16]. Specifically, Agrawal et al. [33] proved that the matrix $(\boldsymbol{A}\|\boldsymbol{AR}+\boldsymbol{CB})\in\mathbb{Z}_q^{n\times(m_1+m_2)}$ is a trapdoor matrix if (1) $\boldsymbol{A}\in\mathbb{Z}_q^{n\times m_1}$ is a trapdoor matrix, or (2) $\boldsymbol{B}\in\mathbb{Z}_q^{n\times m_2}$ is a trapdoor matrix, $\boldsymbol{R}\in\mathbb{Z}^{m_1\times m_2}$ is a small norm matrix and $\boldsymbol{C}\in\mathbb{Z}_q^{n\times n}$ is invertible. By combining this technique with a full-rank difference (FRD) encoding $\mathsf{FRD}:\{0,1\}^{\kappa}\to\mathbb{Z}_q^{n\times n}$, Agrawal et al. [33] obtained an efficient IBE secure against selective identity and chosen ciphertext attacks, where the public key has four matrices $\mathsf{pk}=(\boldsymbol{A}_1,\boldsymbol{A}_2,\boldsymbol{B},\boldsymbol{U})$, and the ciphertext $C=(\boldsymbol{c}_1,\boldsymbol{c}_2,\boldsymbol{c}_3)$ under a user identity $\mathrm{id}\in\{0,1\}^{\kappa}$ has three vectors:

$$\begin{aligned}&\boldsymbol{c}_1=\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{s}+\boldsymbol{e}_1\in\mathbb{Z}_q^{m_1},\qquad &&\boldsymbol{c}_2=(\boldsymbol{A}_2+\mathsf{FRD}(\mathrm{id})\boldsymbol{B})^{\mathrm{T}}\boldsymbol{s}+\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1\in\mathbb{Z}_q^{m_2},\\&\boldsymbol{c}_3=\boldsymbol{U}^{\mathrm{T}}\boldsymbol{s}+\boldsymbol{e}_2+\mu\cdot\tfrac{q}{2}\in\mathbb{Z}_q^{\ell},\end{aligned}$$

where $\boldsymbol{s}\xleftarrow{\$}\mathbb{Z}_q^n,\boldsymbol{e}_1\xleftarrow{\$}D_{\mathbb{Z}^{m_1},\alpha q},\boldsymbol{R}\xleftarrow{\$}D_{\mathbb{Z}^{m_1\times m_2},\omega(\sqrt{\log n})}$ and $\boldsymbol{e}_2\xleftarrow{\$}D_{\mathbb{Z}^{\ell},\alpha q}$. The secret key is a trapdoor of the matrix $\boldsymbol{A}_1$, which can be used to extract a small norm matrix $\boldsymbol{E}$ satisfying $\boldsymbol{A}_{\mathrm{id}}\boldsymbol{E}=(\boldsymbol{A}_1\|\boldsymbol{A}_2+\mathsf{FRD}(\mathrm{id})\boldsymbol{B})\boldsymbol{E}=\boldsymbol{U}$. In the security proof, the matrix $\boldsymbol{A}_1$ is uniformly chosen at random, while the matrix $\boldsymbol{A}_2$ is set to be $\boldsymbol{A}_2=\boldsymbol{A}_1\boldsymbol{R}-\mathsf{FRD}(\mathrm{id}^*)\boldsymbol{B}$ for a small norm matrix $\boldsymbol{R}\xleftarrow{\$}D_{\mathbb{Z}^{m_1\times m_2},\omega(\sqrt{\log n})}$ and a challenge identity $\mathrm{id}^*\xleftarrow{\$}\{0,1\}^{\kappa}$. By doing this, on the first hand, we have that $\boldsymbol{A}_{\mathrm{id}}=(\boldsymbol{A}_1\|\boldsymbol{A}_2+\mathsf{FRD}(\mathrm{id})\boldsymbol{B})$ for any $\mathrm{id}\neq\mathrm{id}^*$ is trapdoor matrix, and thus any ciphertext associated with identity $\mathrm{id}\neq\mathrm{id}^*$ can be decrypted by using $\boldsymbol{R}$ and the trapdoor of $\boldsymbol{B}$. On the other hand, we have that $\boldsymbol{A}_{\mathrm{id}^*}=(\boldsymbol{A}_1\|\boldsymbol{A}_1\boldsymbol{R})$ and the distribution of $\boldsymbol{c}_2^*$ of the challenge ciphertext $C^*=(\boldsymbol{c}_1^*,\boldsymbol{c}_2^*,\boldsymbol{c}_3^*)$ is essentially statistically close to the distribution of $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{c}_1^*$. Thus, in the security proof one can safely replace $\boldsymbol{c}_2^*$ with $\boldsymbol{R}^{\mathrm{T}}\boldsymbol{c}_1^*$, and base the security of the IBE scheme on the hardness of the LWE instances $(\boldsymbol{A}_1,\boldsymbol{c}_1^*=\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{s}^*+\boldsymbol{e}_1^*)$ and $(\boldsymbol{U},\boldsymbol{U}^{\mathrm{T}}\boldsymbol{s}^*+\boldsymbol{e}_2^*)$.

Let $\boldsymbol{G}\in\mathbb{Z}_q^{n\times nk}$ be the public known primitive matrix [40], and $\boldsymbol{S}\in\mathbb{Z}_q^{nk\times nk}$ be any basis of the lattice $\boldsymbol{\Lambda}_q(\boldsymbol{G})=\{\boldsymbol{y}\in\mathbb{Z}_q^{nk}:\boldsymbol{y}=\boldsymbol{G}^{\mathrm{T}}\boldsymbol{x}\mod q,\boldsymbol{x}\in\mathbb{Z}_q^n\}$, where $k=\lceil\log_2 q\rceil$. By improving the tag-based trapdoor technique in [33] with the public known matrix $\boldsymbol{G}\in\mathbb{Z}_q^{n\times nk}$ and using a new message encoding, Micciancio and Peikert [40] further improved the IBE scheme in [33] and obtained the best known CCA1-secure (tag-based) PKE which saves two matrices in the public key and a vector in the ciphertext (note that an IBE can be naturally treated as a tag-based PKE). Formally, the public key of the PKE in [40] only has two random matrices $\mathsf{pk}=(\boldsymbol{A}_1,\boldsymbol{A}_2)\in\mathbb{Z}_q^{n\times m_1}\times\mathbb{Z}_q^{n\times m_2}$, and for a message $\mu\in\{0,1\}^{nk}$, the ciphertext $C=(\mathrm{tag},\boldsymbol{c}_1,\boldsymbol{c}_2)$ consists of two vectors:

$$\begin{aligned}&\boldsymbol{c}_1=2(\boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{s}\mod q)+\boldsymbol{e}_1\mod 2q\in\mathbb{Z}_{2q}^{m_1},\\&\boldsymbol{c}_2=2((\boldsymbol{A}_2+\mathsf{FRD}(\mathrm{tag})\boldsymbol{G})^{\mathrm{T}}\boldsymbol{s}\mod q)+\boldsymbol{e}_2+\boldsymbol{S}\mu\mod 2q\in\mathbb{Z}_{2q}^{m_2},\end{aligned}$$

where $\mathrm{tag}\xleftarrow{\$}\{0,1\}^{\kappa},\boldsymbol{s}\xleftarrow{\$}\mathbb{Z}_q^n,\boldsymbol{e}_1\xleftarrow{\$}D_{\mathbb{Z}^{m_1},\alpha' q}$ and $\boldsymbol{e}_2\xleftarrow{\$}D_{\mathbb{Z}^{nk},\sqrt{\|\boldsymbol{e}_1\|^2+m_1(\alpha' q)^2}\cdot\omega(\sqrt{\log n})}$. The secret key is a small norm matrix $\boldsymbol{R}$ satisfying $\boldsymbol{A}_2=-\boldsymbol{A}_1\boldsymbol{R}$, which can be used to run a trapdoor inversion algorithm to recover the message $\mu\in\{0,1\}^{nk}$ as long as $\|\boldsymbol{e}_1\|$ and $\|\boldsymbol{e}_2\|$ are small. Although this encoding allows to save a matrix $\boldsymbol{U}$ in the public key and a vector in the ciphertext, it only works when the modulus in the encryption is changed from previous $q$ to $2q$, which in turn requires a larger noise parameter $\alpha' q=3\alpha q$

for lifting the LWE problem with Gaussian parameter $\alpha q$ and modulus $q$ to modulus $2q$ in the security proof (see [40, Theorem 6.3] for details).

We first note that the trapdoor inversion algorithm [40] crucially depends on the size of the LWE's "error term" (i.e., it only works when the error size is small), but does not care about the size of the LWE "secret term" $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$. Furthermore, by the fact that the HNF variant [48] of the LWE problem where the secret $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$ is as hard as the standard LWE problem where $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$, we can safely replace $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$ with $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$ in the encryption. Note that for fixed $\alpha q \in \mathbb{R}$ and randomly chosen $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$, we have that $\|\boldsymbol{s}\|_\infty \leqslant B$ holds for some constant $B > 0$ with overwhelming probability by the Gaussian tail inequality. Thus, for $B \ll q$ (as is usually the case), if we encode the secret $\boldsymbol{s} = (s_0, \ldots, s_{n-1})^{\mathrm{T}}$ as an element of $\mathbb{Z}_q^n$, the most significant bits of each $s_i$ in the binary representation are "not used" (i.e., those bits are always zeros with overwhelming probability). Our starting point is to encode the message into those "unused" bit-slots. Formally, we introduce a pair of message encode/decode algorithms $(\mathsf{encode}_d : \mathbb{Z}_d^n \to \mathbb{Z}_q^n, \mathsf{decode}_d : \mathbb{Z}_q^n \to \mathbb{Z}_d^n)$ for some integer $d \geqslant 2$ (see Subsection 3.1 for more details) such that for any $\boldsymbol{v} \in \mathbb{Z}_d^n$ and $0 < B \ll q$, we have $\mathsf{decode}_d(\boldsymbol{s} + \mathsf{encode}_d(\boldsymbol{v})) = \boldsymbol{v}$ as long as $\|\boldsymbol{s}\|_\infty \leqslant B$. By doing this, we can keep the advantage of the PKE in [40] (i.e., only having two matrices in the public key $\mathsf{pk} = (\boldsymbol{A}_1, \boldsymbol{A}_2)$ and two vectors in the ciphertext $C = (\mathrm{tag}, \boldsymbol{c}_1, \boldsymbol{c}_2)$) without lifting the modulus in the encryption from $q$ to $2q$, and thus reduce the noise parameter for the encryption from $\alpha' = 3\alpha$ in [40] to the LWE Gaussian parameter $\alpha$. Specifically, given a message $\mu \in \mathbb{Z}_d^n$, our new encryption algorithm first computes $\hat{\boldsymbol{s}} = \boldsymbol{s} + \mathsf{encode}_d(\mu)$, and then computes the ciphertext $C = (\mathrm{tag}, \boldsymbol{c}_1, \boldsymbol{c}_2)$ as follows:

$$\boldsymbol{c}_1 = \boldsymbol{A}_1^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_1 \in \mathbb{Z}_q^{m_1}, \qquad \boldsymbol{c}_2 = (\boldsymbol{A}_2 + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G})^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_2 \in \mathbb{Z}_q^{m_2},$$

where $\mathrm{tag} \xleftarrow{\$} \{0,1\}^\kappa, \boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}, \boldsymbol{e}_1 \xleftarrow{\$} D_{\mathbb{Z}^{m_1},\alpha q}$ and $\boldsymbol{e}_2 \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q\sqrt{2m_1}\cdot\omega(\sqrt{\log n})}$. The secret key is still a small norm matrix $\boldsymbol{R}$ satisfying $\boldsymbol{A}_2 = -\boldsymbol{A}_1\boldsymbol{R}$. In the security proof, given an LWE tuple $(\boldsymbol{A}_1, \boldsymbol{b}) \in \mathbb{Z}_q^{n\times m_1} \times \mathbb{Z}_q^{m_1}$, one can set the public key $\mathsf{pk} = (\boldsymbol{A}_1, \boldsymbol{A}_2)$ with $\boldsymbol{A}_2 = -\boldsymbol{A}_1\boldsymbol{R} - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G} \in \mathbb{Z}_q^{n\times nk}$ for $\boldsymbol{R} \xleftarrow{\$} D_{\mathbb{Z}^{m_1\times nk},\omega(\sqrt{\log n})}$ and $\mathrm{tag}^* \xleftarrow{\$} \{0,1\}^\kappa$, and simulate the challenge ciphertext $C^* = (\mathrm{tag}^*, \boldsymbol{c}_1^* = \boldsymbol{b} + \boldsymbol{A}_1^{\mathrm{T}}\mathsf{encode}_d(\mu_\delta), \boldsymbol{c}_2^* = -\boldsymbol{R}^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2')$ by using the additive homomorphism of the LWE problem (i.e., $\boldsymbol{A}_1^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_1 = \boldsymbol{A}_1^{\mathrm{T}}\boldsymbol{s} + \boldsymbol{e}_1 + \boldsymbol{A}_1^{\mathrm{T}}\mathsf{encode}_d(\mu)$) and an independently chosen $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q\sqrt{m_1}\cdot\omega(\sqrt{\log n})}$, where $(\mu_0, \mu_1)$ is the challenge message pair and $\delta \xleftarrow{\$} \{0,1\}$. Note that $\boldsymbol{c}_2^*$ in the challenge ciphertext $C^* = (\mathrm{tag}^*, \boldsymbol{c}_1^*, \boldsymbol{c}_2^*)$ is directly generated from $\boldsymbol{c}_1^*$, which relies on the fact that the distribution of $-\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1^* + \boldsymbol{e}_2'$ for $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{m_1},\alpha q}, \boldsymbol{R} \xleftarrow{\$} D_{\mathbb{Z}^{m_1\times nk},\omega(\sqrt{\log n})}$ and $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q\sqrt{m_1}\cdot\omega(\sqrt{\log n})}$ is statistically close to $\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q\sqrt{2m_1}\cdot\omega(\sqrt{\log n})}$ (see Lemma 3). In other words, the simulated $C^* = (\mathrm{tag}^*, \boldsymbol{c}_1^*, \boldsymbol{c}_2^*)$ is statistically close to the distribution of the real challenge ciphertext if $(\boldsymbol{A}_1, \boldsymbol{b})$ is a real LWE instance, and perfectly hides the message $\mu_\delta$ if $(\boldsymbol{A}_1, \boldsymbol{b})$ is uniformly random.

Our second observation is that under appropriate choice of parameters, the first ciphertext part $\boldsymbol{c}_1 = \boldsymbol{A}_1^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_1 \in \mathbb{Z}_q^{m_1}$ essentially uniquely fixes $\hat{\boldsymbol{s}}$ (see Lemma 6) and thus the message $\mu \in \mathbb{Z}_d^n$. By applying some necessary checks in the decryption algorithm, we can be assured that there is only a single valid message $\mu$ for all ciphertexts $C = (*, \boldsymbol{c}_1, *)$ which share the same first ciphertext part $\boldsymbol{c}_1$, i.e., conditioned on that the decryption algorithm does not return a failure symbol $\perp$, its output is essentially independent from the choice of the tag and the second ciphertext part $\boldsymbol{c}_2$. This feature basically says that the ciphertext is (partially) non-malleable, and our idea is to extend this non-malleability to the whole ciphertext (which is required for achieving CCA2-security). For this, we further modify the encryption algorithm. Formally, given a message $\mu \in \mathbb{F}_{2^\kappa}$, it first chooses $x, y, z \xleftarrow{\$} \mathbb{F}_{2^\kappa}$ from the finite field $\mathbb{F}_{2^\kappa}$, and interprets the bit-concatenation of $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v} = x\|y\|z \in \mathbb{Z}_d^n$ (which can always be done if $n\log_2 d \geqslant 3\kappa$). Then, it computes $\hat{\boldsymbol{s}} = \boldsymbol{s} + \mathsf{encode}_d(\boldsymbol{v})$ and the ciphertext $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ as follows:

$$\begin{aligned}
&\boldsymbol{c}_1 = \boldsymbol{A}_1^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_1 \in \mathbb{Z}_q^{m_1}, && \mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \in \mathbb{F}_{2^\kappa},\\
&\boldsymbol{c}_2 = (\boldsymbol{A}_2 + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G})^{\mathrm{T}}\hat{\boldsymbol{s}} + \boldsymbol{e}_2 \in \mathbb{Z}_q^{m_2}, && c_3 = x + \mu \in \mathbb{F}_{2^\kappa},\\
&\tau = \mathsf{H}(\boldsymbol{c}_2, c_3) \in \mathbb{F}_{2^\kappa}, && c_4 = \tau y + z \in \mathbb{F}_{2^\kappa},
\end{aligned}$$

where $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}, \boldsymbol{e}_1 \xleftarrow{\$} D_{\mathbb{Z}^{m_1},\alpha q}, \boldsymbol{e}_2 \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q\sqrt{2m_1}\cdot\omega(\sqrt{\log n})}$ and the function $\mathsf{H} : \{0,1\}^* \to \mathbb{F}_{2^\kappa}$ is a

collision resistant hash function. Technically, the third part $c_3$ is a one-time padding encryption which is used to ensure that we can generate $\boldsymbol{c}_1$ and thus the tag before seeing the challenge message pair in the security proof. The last part $c_4$ is a Carter-Wegman style one-time MAC, which is used to ensure the integrity of $(\boldsymbol{c}_2, c_3)$ when $\boldsymbol{c}_1$ (and thus $y, z$) is fixed.

At first glance, the above construction seems to raise a circularity issue: the MAC key $(y, z)$ is used to authenticate $\boldsymbol{c}_2$ which depends on the MAC key. However, the key point is that we will only invoke the security of the MAC to reject the decryption query with $C = (\boldsymbol{c}_1 = \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4 \neq c_4^*)$, where $\boldsymbol{c}_1^*$ uniquely fixes and computationally hides the real MAC key $(y^*, z^*)$ for the challenge ciphertext $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$. Namely, we will only rely on the security of the MAC when the key $(y^*, z^*)$ is not determined by the adversary's choice of $\boldsymbol{c}_2$. Thus, in order to ensure that the adversary cannot find a valid MAC tag $c_4 \neq c_4^*$ under the key $(y^*, z^*)$ and thus cannot make a valid decryption query with $C = (\boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4)$, we only have to show that $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*)$ leaks no information of $(y^*, z^*)$, which in turn can be proven by using the pseudorandomness of the LWE problem and the fact that $\boldsymbol{c}_2^*$ can be generated by directly using $\boldsymbol{c}_1^*$.

We now give a sketch of the security proof. Formally, given an LWE tuple $(\boldsymbol{A}_1, \boldsymbol{b})$, the reduction first randomly chooses $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, and interprets the bit-concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^* \| y^* \| z^* \in \mathbb{Z}_d^n$. Then, it computes $\boldsymbol{c}_1^* = \boldsymbol{b} + \boldsymbol{A}_1^{\mathrm{T}} \mathsf{encode}_d(\boldsymbol{v})$, $\boldsymbol{A}_2 = -\boldsymbol{A}_1 \boldsymbol{R} - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}$, $\boldsymbol{c}_2^* = -\boldsymbol{R}^{\mathrm{T}} \boldsymbol{c}_1^* + \boldsymbol{e}_2'$ and sets the public key $\mathsf{pk} = (\boldsymbol{A}_1, \boldsymbol{A}_2)$, where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$, $\boldsymbol{R} \xleftarrow{\$} D_{\mathbb{Z}^{m_1 \times nk}, \omega(\sqrt{\log n})}$, and $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk}, \alpha q \sqrt{m_1} \cdot \omega(\sqrt{\log n})}$. Given a challenge message pair $(\mu_0, \mu_1)$, the reduction randomly chooses $\delta \xleftarrow{\$} \{0, 1\}$, computes $c_3^* = x^* + \mu_\delta$, $c_4^* = \tau^* y^* + z^* \in \mathbb{F}_{2^\kappa}$, and returns the challenge ciphertext $C = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$. By the pseudorandomness of $(\boldsymbol{A}_1, \boldsymbol{b})$, we have that $(x^*, y^*, z^*)$ is computationally hidden in $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*)$, which means that $\mu_\delta$ is computationally hidden in the ciphertext. Thus, it suffices to show that the adversary cannot obtain non-negligible advantage from the decryption query. Note that for a decryption query $C = (\boldsymbol{c}_1 \neq \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4)$, we have that $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \neq \mathsf{H}(\boldsymbol{c}_1^*) = \mathrm{tag}^*$ holds by the collision-resistant of $\mathsf{H}$, which means that the ciphertext $C$ can be correctly decrypted by using $\boldsymbol{R}$. As for a decryption query $C = (\boldsymbol{c}_1 = \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4) \neq C^*$, the reduction will directly return $\perp$, since by the unique witness of the LWE problem, such a ciphertext $C \neq C^*$ is valid if and only if $(\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*)$ and $c_4 = \tau y^* + z^*$, where $\tau = \mathsf{H}(\boldsymbol{c}_2, c_3)$. By the collision resistant of $\mathsf{H}$, we have that $\tau \neq \tau^* = \mathsf{H}(\boldsymbol{c}_2^*, c_3)$ and $c_4 \neq c_4^*$ hold. In other words, if the adversary can output a valid decryption query $C = (\boldsymbol{c}_1 = \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4) \neq C^*$, it must be able to uniquely determine the one-time MAC key $(y^*, z^*)$ (since given $c_4^* = \tau^* y^* + z^*$ and $c_4 = \tau y^* + z^* \neq c_4^*$, one can efficiently recover the pair $(y^*, z^*)$), which contradicts the facts that $(y^*, z^*)$ are uniformly chosen at random and are computationally hidden in the challenge ciphertext $C^*$.

Finally, we emphasize that our message encoding is very crucial for our construction because: (1) the trapdoor inversion algorithm will not work if one encodes the message into the most significant bits of the "error term" of $\boldsymbol{c}_1$ or $\boldsymbol{c}_2$; (2) it would require a large noise parameter and thus increase the error size in decryption if one encodes the message into the "error term" of $\boldsymbol{c}_2$ as that in [40]; and (3) most importantly, the above proof of the CCA2-security will not work if one encodes the message into the "error term" of $\boldsymbol{c}_2$ as that in [40] since we cannot rely on the MAC security to reject a decryption query $C = (\boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4) \neq C^*$ when the MAC key is determined by $\boldsymbol{c}_2$ which itself is chosen by the adversary (i.e., there is a circularity issue). Besides, as an independent of interest, this message encoding may also be very useful in other applications such as lattice-based IBEs and attribute-based encryptions (ABEs).

### 1.3 Related work and discussion

Along with the introduction of the LWE problem, Regev [47] proposed the first LWE-based PKE, which can only encrypt a 1-bit message. Later, several studies were extended to support longer messages (e.g., [24]). At STOC 2008, Gentry et al. [46] gave a "dual" variant of Regev's scheme, which was used to construct the first identity-based encryption (IBE) from lattices. Lindner and Peikert [49] gave a more compact LWE-based PKE, which almost reduces all the parameters by a factor of $\log q$. At CCS 2016, Bos et al. [50] proposed a practical public key-encapsulation mechanism (KEM). The ring-LWE was considered in [51,52] to construct PKE with small key and ciphertext sizes. Stehlé and Steinfeld [53] gave a variant of the NTRU cryptosystem, which has a security proof based on the ring-LWE assumption.

Recently, several practical PKEs/KEMs from ring-LWE were proposed, e.g., NewHope [54, 55]. All the above PKEs only have CPA-security, but can be boosted into CCA2-secure ones by applying the Fujisaki-Okamoto transform in the random oracle (RO) model. However, a RO-based solution may not be always satisfiable in the real world [10]. In the post-quantum setting, this becomes more subtle since a classic RO-based scheme may even not be secure against adversaries who can query the RO with quantum state [56]. This is why some NIST PQC submissions also provide security arguments in the quantum RO model [56]. Unfortunately, the PKEs in the quantum RO model typically have large security reduction loss (e.g., at least a quadratic security loss [57]) with respective to (w.r.t.) the underlying hard assumptions [9,57,58], which has become one of the main concerns when estimating the actual security of a scheme with given concrete parameters. We note that this security reduction loss is solely introduced by the security arguments in the quantum RO model, and is irrelevant to the possibly generic speedup of the quantum algorithm in solving hard problems (namely, if the underlying hard problem also suffers from a generic quadratic speedup, e.g., by applying the Grover algorithm, the resulting scheme will suffer from a quartic security loss in the quantum setting). In contrast, a standard model PKE scheme usually has much tighter security reduction, and is thus relatively more interesting in the post-quantum era.

As an instantiation of the generic framework from lossy trapdoor functions (LTDF), Peikert and Waters [18] gave the first standard-model CCA2-secure PKE from LWE. The LTDF techniques were later extended to construct several standard model CCA2-secure PKE from lattices [29–31,59]. However, they are relatively inefficient (e.g., having large public-key and ciphertext sizes) due to the use of Dolev-Dwork-Naor like technique [13, 32] and signatures. By applying the BCHK transform [16], one can obtain CCA2-secure PKEs from IBEs or TBEs on lattices in the standard model [33–39]. In this setting, Micciancio and Peikert [40] improved the standard model IBE in [33], and presented the best known CCA1-secure PKE from lattices. Unlike many existing LWE-based PKEs which encode the messages into the "error term", we encode the messages into the most significant bits of the "secret term", which is very crucial to our CCA2-secure PKE construction, and might be of independent interest. We also note that the recent independent study [58] constructed a deterministic PKE by simply encoding the message as the "secret term" and the "error term" of the LWE problem, which is very different from ours from the perspective of both techniques and functionalities.

## 2 Preliminaries

### 2.1 Notation

Denote the natural logarithm (respectively, the logarithm with base $b$) as log (respectively, $\log_b$). The standard notations $O, \omega$ are used to classify the growth of functions. A function $f(n)$ is negligible in $n$ if for every positive $c$, we have $f(n) < n^{-c}$ for sufficiently large $n$. By $\mathrm{negl}(n)$ we denote an arbitrary negligible function. A probability is said to be overwhelming if it is $1 - \mathrm{negl}(n)$. The notation $\xleftarrow{\$}$ denotes randomly choosing elements from a distribution (or the uniform distribution over a finite set). By $x \sim D$ we mean the random variable $x$ follows a distribution $D$.

Denote $\mathbb{R}$ (respectively, $\mathbb{Z}$) as the set of real numbers (respectively, integers). Vectors are column vectors and denoted by bold lower-case letters (e.g., $\boldsymbol{v}$), and $\boldsymbol{v}^{\mathrm{T}}$ denotes the transpose of $\boldsymbol{v}$. Matrices are the sets of column vectors and denoted by bold capital letters (e.g., $\boldsymbol{X}$). The concatenation of a matrix $\boldsymbol{X} \in \mathbb{R}^{n \times m}$ followed by another matrix $\boldsymbol{Y} \in \mathbb{R}^{n \times m'}$ is denoted as $(\boldsymbol{X} \| \boldsymbol{Y}) \in \mathbb{R}^{n \times (m+m')}$. By $\|\cdot\|$ and $\|\cdot\|_\infty$ we denote the $\ell_2$ and $\ell_\infty$ norm, respectively. The largest singular value of $\boldsymbol{X}$ is $s_1(\boldsymbol{X}) = \max_{\boldsymbol{u}} \|\boldsymbol{X}\boldsymbol{u}\|$, where the maximum is taken over all unit vector $\boldsymbol{u}$.

### 2.2 Public-key encryption

A public-key encryption (PKE) $\Pi$ with message space $\mathcal{P}$ consists of three probabilistic polynomial time (PPT) algorithms (KeyGen, Enc, Dec):

• $\mathsf{KeyGen}(1^\kappa)$ is a PPT algorithm that takes a security parameter $\kappa$ as input, and outputs a pair of public and secret keys (pk, sk).

• $\mathsf{Enc}(\mathrm{pk}, \mu)$ is a PPT algorithm that encrypts a message $\mu \in \mathcal{P}$ under the public key pk and outputs the corresponding ciphertext $C$.

• $\mathsf{Dec}(\mathrm{sk}, C)$ is an efficient deterministic algorithm that decrypts a ciphertext $C$ using the secret key sk and outputs a message $\mu$ (or a symbol $\perp$).

We say that a PKE scheme $\Pi = (\mathsf{KeyGen},\mathsf{Enc},\mathsf{Dec})$ is correct, if for any $\mu \in \mathcal{P}$, $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$ and $C \leftarrow \mathsf{Enc}(\mathrm{pk}, \mu)$, the probability that $\mathsf{Dec}(\mathrm{sk}, C) \neq \mu$ is negligible over the random coins used in both $\mathsf{KeyGen}$ and $\mathsf{Enc}$. The de facto standard security notion for PKE is (adaptively) chosen-ciphertext security, which is modeled by a game between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$.

**KeyGen.** The challenger $\mathcal{C}$ first computes $(\mathrm{pk}, \mathrm{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$. Then, it gives the public key pk to the adversary $\mathcal{A}$, and keeps sk secret.

**Phase 1.** The adversary $\mathcal{A}$ is allowed to make any polynomial number of decryption queries by using any (different) ciphertext $C$ of his choice. The challenger $\mathcal{C}$ computes $\mu \leftarrow \mathsf{Dec}(\mathrm{sk}, C)$, and returns $\mu$ to $\mathcal{A}$.

**Challenge.** The adversary $\mathcal{A}$ outputs two equal-length messages $(\mu_0, \mu_1)$. The challenger $\mathcal{C}$ chooses a bit $\delta^* \xleftarrow{\$} \{0, 1\}$, and computes $C^* \leftarrow \mathsf{Enc}(\mathrm{pk}, \mu_{\delta^*})$. Finally, it returns the challenge ciphertext $C^*$ to $\mathcal{A}$.

**Phase 2.** The adversary is allowed to make more decryption queries with any ciphertext $C \neq C^*$. The challenger $\mathcal{C}$ responds as in Phase 1.

**Guess.** Finally, $\mathcal{A}$ outputs a guess $\delta \in \{0, 1\}$. If $\delta = \delta^*$, the challenger $\mathcal{C}$ outputs 1, else outputs 0.

**Definition 1** (CCA2-security)**.** We say that a PKE scheme $\Pi$ is CCA2-secure if for any PPT adversary $\mathcal{A}$, its advantage

$$\mathrm{Adv}^{\text{ind-cca2}}_{\Pi,\mathcal{A}}(\kappa) = \left|\Pr[\delta = \delta^*] - \frac{1}{2}\right|$$

in the above game is negligible in security parameter $\kappa$.

The CPA-security and CCA1-security can be defined via modified games. Concretely, the CPA-security game does not allow the adversary to make any decryption queries, while the CCA1-security game only allows the adversary to make decryption queries before the challenge phase.

### 2.3 Gaussian, learning with errors and trapdoors

**Gaussian.** The Gaussian function $\rho_{s,\boldsymbol{c}}(\boldsymbol{x})$ over $\mathbb{R}^m$ centered at $\boldsymbol{c} \in \mathbb{R}^m$ with parameter $s > 0$ is defined as $\rho_{s,\boldsymbol{c}}(\boldsymbol{x}) = \exp(-\pi\|\boldsymbol{x} - \boldsymbol{c}\|^2/s^2)$. For lattice $\boldsymbol{\Lambda} \subseteq \mathbb{R}^m$, let $\rho_{s,\boldsymbol{c}}(\boldsymbol{\Lambda}) = \sum_{\boldsymbol{x}\in\boldsymbol{\Lambda}} \rho_{s,\boldsymbol{c}}(\boldsymbol{x})$, and define the discrete Gaussian distribution over $\boldsymbol{\Lambda}$ as $D_{\boldsymbol{\Lambda},s,\boldsymbol{c}}(\boldsymbol{y}) = \frac{\rho_{s,\boldsymbol{c}}(\boldsymbol{y})}{\rho_{s,\boldsymbol{c}}(\boldsymbol{\Lambda})}$, where $\boldsymbol{y} \in \boldsymbol{\Lambda}$. We omit the subscript $\boldsymbol{c}$ in the above notations if $\boldsymbol{c} = \boldsymbol{0}$.

**Lemma 1** ( [49,60])**.** For any real $s, t > 0$, $c \geqslant 1, C = c \cdot \exp(\frac{1-c^2}{2}) < 1$, integer $m > 0$, and any $\boldsymbol{y} \in \mathbb{R}^m$ we have the followings hold:

- $\Pr_{\boldsymbol{x}\xleftarrow{\$}D_{\mathbb{Z}^m,s}}[\|\boldsymbol{x}\|_\infty > t \cdot s] \leqslant 2\mathrm{e}^{-\pi t^2}$,
- $\Pr_{\boldsymbol{x}\xleftarrow{\$}D_{\mathbb{Z}^m,s}}[\|\boldsymbol{x}\| > c \cdot \frac{1}{\sqrt{2\pi}} \cdot s\sqrt{m}] \leqslant C^m$,
- $\Pr_{\boldsymbol{x}\xleftarrow{\$}D_{\mathbb{Z}^m,s}}[|\langle \boldsymbol{x}, \boldsymbol{y}\rangle| > t \cdot s\|\boldsymbol{y}\|] \leqslant 2\mathrm{e}^{-\pi t^2}$.

**Lemma 2** ( [40])**.** Let integer $n > 0$, and $q$ a power of some prime $p \geqslant 2$. Let integer $m \geqslant n\log_2 q + \omega(\log n)$. Then, for any $\ell = \mathrm{poly}(n)$ and real $r \geqslant \omega(\sqrt{\log n})$, the distribution $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{R})$ is statistically close to uniform over $\mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^{n\times\ell}$, where $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^m,r})^\ell$.

The following lemma is implicit in the proof of [40, Theorem 6.3], which can be proven by combining [61, Theorem 3.1] and [47, Corollary 3.10].

**Lemma 3** ( [40])**.** Let $r \geqslant \omega(\sqrt{\log n})$. Then, for any vectors $\boldsymbol{v} \in \mathbb{Z}^m$ and $\boldsymbol{c} \in \mathbb{Z}^m$, any matrix $\boldsymbol{r} \xleftarrow{\$} D_{\mathbb{Z}^m,r,\boldsymbol{c}}$, and $e \xleftarrow{\$} D_{\mathbb{Z},\alpha q r\cdot\sqrt{m}}$, the distribution $\boldsymbol{r}^{\mathrm{T}}\boldsymbol{v} + e$ is statistically close to $D_{\mathbb{Z},s}$, where $s = r \cdot \sqrt{\|\boldsymbol{v}\|^2 + m(\alpha q)^2}$.

Following [40, 62], we say that a random variable $X$ over $\mathbb{R}$ is subgaussian with parameter $s$ if for all $t \in \mathbb{R}$, the (scaled) moment-generating function satisfies $\mathbb{E}(\exp(2\pi tX)) \leqslant \exp(\pi s^2 t^2)$. For any lattice $\boldsymbol{\Lambda} \subset \mathbb{R}^m$ and $s > 0$, $D_{\boldsymbol{\Lambda},s}$ is subgaussian with parameter $s$. Besides, any $B$-bounded symmetric random variable $X$ (i.e., $|X| \leqslant B$) is subgaussian with parameter $B\sqrt{2\pi}$ [40]. For random subgaussian matrix, we have the following result from the non-asymptotic theory of random matrices [63].

**Lemma 4.** Let $\boldsymbol{X} \in \mathbb{R}^{n\times m}$ be a random subgaussian matrix with parameter $s$. There exists a universal constant $C \approx 1/\sqrt{2\pi}$ such that for any $t \geqslant 0$, we have $s_1(\boldsymbol{X}) \leqslant C \cdot s \cdot (\sqrt{m} + \sqrt{n} + t)$ except with probability at most $2\exp(-\pi t^2)$.

**Learning with errors.** For any positive integers $n, q \in \mathbb{Z}$, real $\alpha > 0$ and vector $\boldsymbol{s} \in \mathbb{Z}_q^n$, define $A_{\boldsymbol{s},\alpha} = \{(\boldsymbol{a}, \boldsymbol{a}^{\mathrm{T}}\boldsymbol{s} + e \mod q) : \boldsymbol{a} \xleftarrow{\$} \mathbb{Z}_q^n, e \xleftarrow{\$} D_{\mathbb{Z},\alpha q}\}$. For $m$ independent samples $(\boldsymbol{a}_1, y_1), \ldots, (\boldsymbol{a}_m, y_m)$ from $A_{\boldsymbol{s},\alpha}$, we denote it in a matrix form $(\boldsymbol{A}, \boldsymbol{y}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m$, where $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m)$ and $\boldsymbol{y} = (y_1, \ldots, y_m)^{\mathrm{T}}$. We say that a PPT algorithm solves the $\mathrm{LWE}_{n,m,q,\alpha}$ problem if, for uniformly random $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$, given $m$ samples from $A_{\boldsymbol{s},\alpha}$ it outputs $\boldsymbol{s}$ with non-negligible probability. The decisional LWE is asked to distinguish $A_{\boldsymbol{s},\alpha}$ from the uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ (with only polynomial samples). For certain parameters, the decisional LWE problem is polynomially equivalent to its search version, which in turn is provably at least as hard as quantumly approximating SIVP on $n$-dimensional lattices to within polynomial factors in the worst case [47, 64]. A variant of the LWE problem (known as the Hermite normal form) where the secret $\boldsymbol{s} \in \mathbb{Z}_q^n$ is chosen from the error distribution (i.e., $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$) is also polynomially equivalent to the standard LWE problem [48].

**$q$-ary lattices and trapdoors.** Let $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, define two $q$-ary lattices:

$$\begin{aligned}\boldsymbol{\Lambda}_q^{\perp}(\boldsymbol{A}) &= \{\boldsymbol{e} \in \mathbb{Z}^m \text{ s.t. } \boldsymbol{A}\boldsymbol{e} = \boldsymbol{0} \mod q\},\\ \boldsymbol{\Lambda}_q(\boldsymbol{A}) &= \{\boldsymbol{y} \in \mathbb{Z}^m \text{ s.t. } \exists \boldsymbol{s} \in \mathbb{Z}^n,\ \boldsymbol{A}^t\boldsymbol{s} = \boldsymbol{y} \mod q\}.\end{aligned}$$

We have PPT algorithms [40,45,65] to generate an essentially uniform matrix $\boldsymbol{A}$ together with a trapdoor (or a short basis of $\boldsymbol{\Lambda}_q^{\perp}(\boldsymbol{A})$). We will use the trapdoor notion in [40]. Formally, let $\boldsymbol{G}_b \in \mathbb{Z}_q^{n\times nk}$ be the public primitive matrix with base integer $b \geqslant 2$ in [40, Theorem 4.1], where $k = \lceil \log_b q \rceil$. We usually omit the subscript $b$ if $b = 2$, and denote $\boldsymbol{G} = \boldsymbol{G}_2$ in brief. As shown in [40], there exists a PPT algorithm that inverts $\boldsymbol{y} = \boldsymbol{G}_b^{\mathrm{T}}\boldsymbol{s} + \boldsymbol{e}$ as long as $\|\boldsymbol{e}\| < \frac{q}{2\sqrt{b^2+1}}$. Moreover, if $q = b^k$, the algorithm can invert $\boldsymbol{y} = \boldsymbol{G}_b^{\mathrm{T}}\boldsymbol{s} + \boldsymbol{e}$ if $\|\boldsymbol{e}\|_\infty < \frac{q}{2b}$. The following lemma is implicit in [40, Theorem. 5.4].

**Lemma 5** ([40])**.** Let $\boldsymbol{I}_{nk}$ be the $nk \times nk$ identity matrix. For any matrices $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, $\boldsymbol{R} \in \mathbb{Z}_q^{(m-nk)\times nk}$ and invertible matrix $\boldsymbol{S} \in \mathbb{Z}_q^{n\times n}$ satisfying $\boldsymbol{A}\left(\begin{smallmatrix}\boldsymbol{R}\\ \boldsymbol{I}_{nk}\end{smallmatrix}\right) = \boldsymbol{S}\boldsymbol{G}_b$, there exists a PPT algorithm $\mathsf{Solve}(\boldsymbol{A}, \boldsymbol{R}, \boldsymbol{y})$ that given any $\boldsymbol{y} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s} + \left(\begin{smallmatrix}\boldsymbol{e}_1\\ \boldsymbol{e}_2\end{smallmatrix}\right) \in \mathbb{Z}_q^m$ satisfying $\|\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\| < \frac{q}{2\sqrt{b^2+1}}$, outputs $\boldsymbol{s} \in \mathbb{Z}_q^n$, where $\boldsymbol{e}_1 \in \mathbb{Z}^{m-nk}$ and $\boldsymbol{e}_2 \in \mathbb{Z}^{nk}$.

Moreover, if $q = b^k$, the algorithm $\mathsf{Solve}(\boldsymbol{A}, \boldsymbol{R}, \boldsymbol{y})$ can invert any $\boldsymbol{y} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s} + \left(\begin{smallmatrix}\boldsymbol{e}_1\\ \boldsymbol{e}_2\end{smallmatrix}\right) \in \mathbb{Z}_q^m$ satisfying $\|\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\|_\infty < \frac{q}{2b}$.

We also need the following useful lemma, which is important for our CCA2-secure PKE construction, and may be of independent interest.

**Lemma 6** (Unique witness)**.** Let $n, k > 0$ be integers. Let $q = p^k$ for some prime $p \geqslant 2$, and let $m \geqslant n\log_2 q + \omega(\log n)$. Then, for all but a negligible fraction of $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, and for any $\boldsymbol{u} \in \mathbb{Z}_q^m$, there exists at most one pair $(\boldsymbol{s}, \boldsymbol{e}) \in \mathbb{Z}_q^n \times \mathbb{Z}^m$ such that $\|\boldsymbol{e}\|_\infty < q/8$ and $\boldsymbol{u} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s} + \boldsymbol{e}$.

*Proof.* The proof is adapted from [46, Lemma 5.3]. For any $\boldsymbol{u} \in \mathbb{Z}_q^m$, we assume that there exist two tuples $(\boldsymbol{s}, \boldsymbol{e}) \neq (\boldsymbol{s}', \boldsymbol{e}') \in \mathbb{Z}_q^n \times \mathbb{Z}^m$, such that $\|\boldsymbol{e}\|_\infty, \|\boldsymbol{e}'\|_\infty < q/8$ and $\boldsymbol{u} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s} + \boldsymbol{e} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s}' + \boldsymbol{e}'$. Letting $\tilde{\boldsymbol{s}} = \boldsymbol{s} - \boldsymbol{s}'$ and $\tilde{\boldsymbol{e}} = \boldsymbol{e}' - \boldsymbol{e}$, we have that $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} = \tilde{\boldsymbol{e}}$ for some $\tilde{\boldsymbol{s}} \neq \boldsymbol{0}$ and $\|\tilde{\boldsymbol{e}}\|_\infty < q/4$. Now, it suffices to show that for all but an at most $2^{-\omega(\log n)} = \mathrm{negl}(n)$ fraction of $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, the vector $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ has norm $\|\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}\|_\infty \geqslant q/4$ for any $\tilde{\boldsymbol{s}} \in \mathbb{Z}_q^n \backslash \{\boldsymbol{0}\}$.

Formally, consider the open $\ell_\infty$ "cube" $\mathcal{V}$ of radius $q/4$ (i.e., each edge has length $q/2$). Denote $(\mathbb{Z}_q^n)^* \subseteq \mathbb{Z}_q^n$ as the set of vectors such that each vector has at least one coordinate which is invertible in $\mathbb{Z}_q$. For any fixed nonzero $\tilde{\boldsymbol{s}} \in \mathbb{Z}_q^n$, we can write $\tilde{\boldsymbol{s}} = p^{k'}\tilde{\boldsymbol{s}}'$ for some integer $k' \in \{0, \ldots, k-1\}$ and

$\tilde{\boldsymbol{s}}' \in (\mathbb{Z}_q^n)^*$. Then, for a uniformly random choice of $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, we have that $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}'$ is uniformly over $\mathbb{Z}_q^m$, and that $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} = p^{k'}\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}'$ is uniformly over $\mathbb{Z}_q^m \cap p^{k'}\mathbb{Z}^m$. Denote $S_{k'} = \mathcal{V} \cap p^{k'}\mathbb{Z}^m$, which contains at most $(p^{k-k'}/2)^m$ points. Thus, over the uniformly random choice of $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, the probability that $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} \in S_{k'}$ is at most $(p^{k-k'}/2)^m/p^{(k-k')m} \leqslant 2^{-m}$. Taking a union bound over all nonzero $\tilde{\boldsymbol{s}} \in \mathbb{Z}_q^n$, the probability that $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} \in S_0$ is at most $2^{-\omega(\log n)}$ (note that $S_{k-1} \subset \cdots \subset S_0 = \mathcal{V} \cap \mathbb{Z}^m$ by definition). Since $S_0$ contains all integer vectors with $\ell_\infty$ norm $< q/4$, we have that for all but an at most $2^{-\omega(\log n)}$ fraction of $\boldsymbol{A} \in \mathbb{Z}_q^{n\times m}$, and for any non-zero $\tilde{\boldsymbol{s}} \in \mathbb{Z}_q^n$, the vector $\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ has norm $\|\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}\|_\infty \geqslant q/4$.

## 3 CCA2-secure PKE from lattices

In this section, we first introduce some ingredients for our construction.

### 3.1 Some ingredients

**Collision-resistant hash function.** We say that $\mathsf{H} : X \to Y$ is a collision resistant hash (CRH) if given a security parameter $\kappa$ and a description of $\mathsf{H}$ as inputs, no PPT algorithm $\mathcal{F}$ can find two elements $x_1 \neq x_2 \in X$ such that $\mathsf{H}(x_1) = \mathsf{H}(x_2)$ holds except with negligible probability, where the probability is over the random coins used by $\mathcal{F}$. Namely, if $\mathsf{H} : X \to Y$ is a CRH, we have that

$$\Pr[(x_1, x_2) \leftarrow \mathcal{F}(1^\kappa, \mathsf{H}) : x_1 \neq x_2 \wedge \mathsf{H}(x_1) = \mathsf{H}(x_2)] \leqslant \mathrm{negl}(\kappa)$$

holds. Note that CRH exists under the LWE assumption [18], and it suffices to use the standard SHA3 in practice.

**Full-rank difference encoding.** Let $\kappa$ be the security parameter. We say that $\mathsf{FRD} : \{0,1\}^\kappa \to \mathbb{Z}_q^{n\times n}$ is an encoding with full-rank differences (FRD) if the following two conditions hold: (1) for any $\boldsymbol{u} \neq \boldsymbol{v}$, the matrix $\mathsf{FRD}(\boldsymbol{u} - \boldsymbol{v}) = \mathsf{FRD}(\boldsymbol{u}) - \mathsf{FRD}(\boldsymbol{v}) \in \mathbb{Z}_q^{n\times n}$ is invertible over $\mathbb{Z}_q^{n\times n}$; and (2) $\mathsf{FRD}(\cdot)$ is computable in polynomial time in $n \log q$. As shown in [33, 40, 66], FRD encodings over $\{0,1\}^\kappa$ can be efficiently constructed for any $\kappa \leqslant n$ and $q$ that is a power of some prime $p \geqslant 2$.

**Message encoding.** We define a pair of algorithms $(\mathsf{encode}_d, \mathsf{decode}_d)$, which are parameterized by positive integers $(n, q, d)$. Formally, given any $\boldsymbol{v} \in \mathbb{Z}_d^n$, the algorithm $\mathsf{encode}_d : \mathbb{Z}_d^n \to \mathbb{Z}_q^n$ is defined as $\mathsf{encode}_d(\boldsymbol{v}) = (v_1 \cdot \lfloor \frac{q}{d} \rceil, \ldots, v_n \cdot \lfloor \frac{q}{d} \rceil)$, where $\boldsymbol{v} = (v_1, \ldots, v_n) \in \mathbb{Z}_d^n$. For any $\boldsymbol{u} \in \mathbb{Z}_q^n$, the algorithm $\mathsf{decode}_d : \mathbb{Z}_q^n \to \mathbb{Z}_d^n$ is defined as $\mathsf{decode}_d(\boldsymbol{u}) = (\lfloor u_1 \cdot \frac{d}{q} \rceil, \ldots, \lfloor u_n \cdot \frac{d}{q} \rceil)$, where $\boldsymbol{u} = (u_1, \ldots, u_n) \in \mathbb{Z}_q^n$.

**Lemma 7.** Let $n, q$ be positive integers, and integer $2 \leqslant d < \sqrt{q}$. Then, for any $\boldsymbol{v} \in \mathbb{Z}_d^n$, any $\boldsymbol{e} \in \mathbb{Z}^n$ satisfying $\|\boldsymbol{e}\|_\infty < \frac{q-(d-1)d}{2d}$, and $\boldsymbol{w} = \mathsf{encode}_d(\boldsymbol{v}) + \boldsymbol{e}$, we have that $\boldsymbol{v} = \mathsf{decode}_d(\boldsymbol{w})$ always holds.

*Proof.* Since both algorithms simply apply the same operations on their inputs in a coordinate-wise way, it suffices to show that for any $v \in \mathbb{Z}_d$, any $e \in \mathbb{Z}$ satisfying $|e| < \frac{q-(d-1)d}{2d}$ and $w = v \cdot \lfloor \frac{q}{d} \rceil + e$, we always have $v = \lfloor w \cdot \frac{d}{q} \rceil$. By definition, we have $w = v \cdot (\frac{q}{d} + x) + e$ holds for some $x$ satisfying $|x| \leqslant 1/2$. Thus, $w \cdot \frac{d}{q} = v + (vx + e) \cdot \frac{d}{q}$. Since $|(vx + e) \cdot \frac{d}{q}| \leqslant (|vx| + |e|) \cdot \frac{d}{q} < (\frac{d-1}{2} + \frac{q-(d-1)d}{2d}) \cdot \frac{d}{q} = 1/2$ holds by assumption, we have $v = \lfloor w \cdot \frac{d}{q} \rceil$. This completes the proof.

### 3.2 The construction

Let $\kappa$ be the security parameter. Let $n, \bar{m} > 0$ be integers, and let $q$ be a prime or a power of prime $b \geqslant 2$. Let $k = \lceil \log_b q \rceil$ and $m = \bar{m} + nk$. Let $(\mathsf{encode}_d, \mathsf{decode}_d)$ be the pair of encode/decode algorithms parameterized by $(n, q, d)$ satisfying that $n \log_2 d \geqslant 3\kappa$. Let $\mathbb{F}_{2^\kappa}$ be a finite field of order $2^\kappa$. Let $\mathsf{H} : \{0,1\}^* \to \mathbb{F}_{2^\kappa} \backslash \{0\}$ be a collision-resistant hash function (Namely, we assume that the output of $\mathsf{H}$ does not contain the zero element in $\mathbb{F}_{2^\kappa}$ for simplicity). Let $\mathsf{FRD} : \mathbb{F}_{2^\kappa} \to \mathbb{Z}_q^{n\times n}$ be an FRD encoding. Our CCA2-secure PKE with parameters $(n, \bar{m}, q, b, d, \alpha)$ is given as follows.

- $\mathsf{KeyGen}(1^\kappa)$: randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, and compute $\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}$. Return the pair of public and secret keys $(\mathrm{pk}, \mathrm{sk}) = ((\boldsymbol{A}, \boldsymbol{B}), \boldsymbol{R})$.

• $\mathsf{Enc}(\mathrm{pk}, \mu \in \mathbb{F}_{2^\kappa})$: first randomly choose $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$, $\boldsymbol{e}_1 \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}, \boldsymbol{e}_2 \xleftarrow{\$} D_{\mathbb{Z}^{nk},\gamma}$, and $x, y, z \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $\gamma = \sqrt{\|\boldsymbol{e}_1\|^2 + \bar{m}(\alpha q)^2} \cdot \omega(\sqrt{\log n})$. Then, interpret the bit-concatenation of $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v} = x\|y\|z \in \mathbb{Z}_d^n$ (which can always be done since $n \log_2 d \geqslant 3\kappa$), and compute

$$\begin{aligned} &\tilde{\boldsymbol{s}} = \boldsymbol{s} + \mathsf{encode}_d(\boldsymbol{v}), && \boldsymbol{c}_1 = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_1, \\ &\boldsymbol{c}_2 = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_2, && c_3 = x + \mu \in \mathbb{F}_{2^\kappa}, \\ &c_4 = \tau y + z \in \mathbb{F}_{2^\kappa}, \end{aligned}$$

where $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \in \mathbb{F}_{2^\kappa}$ and $\tau = \mathsf{H}(\boldsymbol{c}_2, c_3) \in \mathbb{F}_{2^\kappa}$. Finally, return the ciphertext $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4) \in \mathbb{Z}_q^{m_1} \times \mathbb{Z}_q^{nk} \times \mathbb{F}_{2^\kappa} \times \mathbb{F}_{2^\kappa}$.

• $\mathsf{Dec}(\mathrm{sk}, C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4))$: first compute $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1)$,

$$\boldsymbol{A}_{\mathrm{tag}} = (\boldsymbol{A}\|\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b), \text{ and } \boldsymbol{u} = \begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{pmatrix}.$$

Then, compute $\tilde{\boldsymbol{s}} \leftarrow \mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}, \boldsymbol{u})$, $\boldsymbol{v} = \mathsf{decode}_d(\tilde{\boldsymbol{s}}) \in \mathbb{Z}_d^n$, and parse $\boldsymbol{v} = \|x\|y\|z$, where $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$. Let $\boldsymbol{e}_1 = \boldsymbol{c}_1 - \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ and $\boldsymbol{e}_2 = \boldsymbol{c}_2 - (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}$. Return $\perp$ if one of the following conditions holds:

- $\|\boldsymbol{e}_1\| > \alpha q\sqrt{\bar{m}}$, or
- $\|\boldsymbol{e}_2\| > \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty > \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$ a power of prime $b$), or
- $c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y + z \in \mathbb{F}_{2^\kappa}$.

Otherwise, return $\mu = c_3 - x \in \mathbb{F}_{2^\kappa}$.

**Correctness.** Note that $\boldsymbol{c}_1 = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_1$ and $\boldsymbol{c}_2 = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_2$, the algorithm $\mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}, \boldsymbol{u})$ can invert $\tilde{\boldsymbol{s}} \in \mathbb{Z}_q^n$ if $\|\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\| < \frac{q}{2\sqrt{b^2+1}}$ by Lemma 5. Because $\tilde{\boldsymbol{s}} = \boldsymbol{s} + \mathsf{encode}_d(\boldsymbol{v})$, one can correctly recover $\boldsymbol{v} \in \mathbb{Z}_d^n$ if $\|\boldsymbol{s}\|_\infty < \frac{q-(d-1)d}{2d}$ by Lemma 7. Note that $\boldsymbol{s} \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}, \boldsymbol{e}_1 \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, and $\boldsymbol{e}_2 \xleftarrow{\$} D_{\mathbb{Z}^{nk},\gamma}$, we have that $\|\boldsymbol{s}\|_\infty \leqslant \alpha q \cdot \omega(\sqrt{\log n}), \|\boldsymbol{e}_1\| \leqslant \alpha q\sqrt{\bar{m}}$, $\|\boldsymbol{e}_2\|_\infty \leqslant \gamma \cdot \omega(\sqrt{\log n})$ and $\|\boldsymbol{e}_2\| \leqslant \gamma\sqrt{nk}$ hold with overwhelming probability by Lemma 1. Since $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, the inequality $s_1(\boldsymbol{R}) \leqslant \sqrt{\bar{m}} \cdot \omega(\sqrt{\log n})$ holds with overwhelming probability by Lemma 4. Since $\gamma = \sqrt{\|\boldsymbol{e}_1\|^2 + \bar{m}(\alpha q)^2} \cdot \omega(\sqrt{\log n})$, we have $\|\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\| \leqslant \alpha q\bar{m} \cdot \omega(\sqrt{\log n})$. Besides, we need $\alpha q \geqslant 2\sqrt{n}$ for the hardness of the LWE problem [47]. We also need Lemmas 2 and 6 in the security proof, which require $\bar{m} \geqslant (n+1)\log_2 q + \omega(\log n)$ and $\|\boldsymbol{e}_1\|_\infty < q/8$.

In all, for the case where $b = 2$ and $q$ is a prime, the decryption algorithm is correct if we set the parameters $\bar{m}, \alpha, q$ such that

$$\bar{m} = (n+1)\log_2 q + \omega(\log n), \quad 1/\alpha = \bar{m} \cdot \omega(\sqrt{\log n}), \quad \alpha q = 2\sqrt{n}, \tag{1}$$

which means that $m = \bar{m} + nk = \tilde{O}(n), 1/\alpha = \tilde{O}(n)$ and $q = \tilde{O}(n^{1.5})$.

To obtain better efficiency, one can set $q$ as a power of a small prime $b$ (e.g., $b = 3$), which allows us to use the inequality $\|\boldsymbol{R}^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\|_\infty \leqslant \alpha q\sqrt{\bar{m}} \cdot \omega(\sqrt{\log n})^2 < \frac{q}{2b}$ in the correctness analysis. In this case, it suffices to set the parameters $\bar{m}, \alpha, q$ such that

$$\bar{m} = (n+1)\log_2 q + \omega(\log n), \quad 1/\alpha = \sqrt{\bar{m}} \cdot \omega(\sqrt{\log n})^2, \quad \alpha q = 2\sqrt{n}, \tag{2}$$

which means that $m = \bar{m} + nk = \tilde{O}(n), 1/\alpha = \tilde{O}(\sqrt{n})$ and $q = \tilde{O}(n)$. In both cases, we can set $2 \leqslant d \leqslant \tilde{O}(\sqrt{n})$.

As commented in [44, 49], the requirement $\alpha q \geqslant 2\sqrt{n}$ used for the theoretical worst-case reduction [47] is not tight, and it is better to mainly consider concrete hardness against known attacks when choosing actual parameters. For example, one can set $n = 450, \bar{m} = 6690, m = 10740$, $q = 3^9 \approx 2^{14.27}, \alpha q = 1.5$ to achieve a decryption error rate less than $2^{-100}$, and a security level about 131-bit by the online LWE estimator [44]. In this case, the sizes of the public key and the secret key are about $nm\lceil\log_2 q\rceil \approx 8.64$ MB, and $\bar{m}nk(\log_2(\alpha q \cdot \omega(\sqrt{\log n})) + 1) \approx 16.15$ MB, respectively. For 128-bit security, we set $\kappa = 256$, the ciphertext size for encrypting a 256-bit message is $m\lceil\log_2 q\rceil + 512$ bits $\approx 19.73$ KB. By using the compressing technique in Section 4, we can reduce the size of the public key, the secret key and the ciphertext to 3.26 MB, 32 bytes and 13.80 KB, respectively.

### 3.3 The security

In this subsection, we show that the above PKE is CCA2-secure. Formally, we have the following theorem.

**Theorem 1.** Let positive integers $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0,1)$ satisfy (1) or (2). If $\mathrm{LWE}_{n,\bar{m},q,\alpha}$ is hard and $\mathsf{H}$ is a collision-resistant hash function, then the above PKE scheme is CCA2-secure in the standard model.

Our proof uses a sequence of games $G_1, \ldots, G_{11}$, with $G_1$ being the real CCA2-security game (where the challenge ciphertext $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$ is honestly generated by first randomly choosing $\delta^* \xleftarrow{\$} \{0,1\}$ and then encrypting $\mu_{\delta^*}$) and $G_{11}$ a random game (where the challenge ciphertext $C^*$ is essentially uniformly random, and thus the adversary's advantage in game $G_{11}$ is negligible). The security is established by showing that $G_1$ and $G_{11}$ are computationally indistinguishable in the adversary's view. We outline the changes of game $G_i$ with respect to its previous game $G_{i-1}$ in Table 2.

*Proof.* We now give the formal proof of Theorem 1. Let $\mathcal{A}$ be an adversary which can break the CCA2-security of our PKE with advantage $\epsilon$. Let $F_i$ be the event that $\mathcal{A}$ correctly guesses $\delta = \delta^*$ in game $i \in \{1, \ldots, 11\}$. By definition, the adversary's advantage $\mathrm{Adv}^{\text{ind-cca2}}_{\mathcal{PKE},\mathcal{A}}(\kappa)$ in game $i$ is exactly $|\Pr[F_i] - 1/2|$.

**(1) Game $G_1$.** This game is the real security game as defined in Subsection 2.2. Formally, the challenger $\mathcal{C}$ works as follows:

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, and compute $\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}$. Then, return the pair of public key $\mathsf{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to the adversary $\mathcal{A}$, and keeps the secret key $\boldsymbol{R}$ private.

**Phase I.** Upon receiving a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$, first compute

$$\boldsymbol{A}_{\mathrm{tag}} = (\boldsymbol{A} \| \boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b), \text{ and } \boldsymbol{u} = \begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{pmatrix},$$

where $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1)$. Then, compute $\tilde{\boldsymbol{s}} \leftarrow \mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}, \boldsymbol{u})$, $\boldsymbol{v} = \mathsf{decode}_d(\tilde{\boldsymbol{s}})$, and parse $\boldsymbol{v} = x\|y\|z$, where $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$. Let $\boldsymbol{e}_1 = \boldsymbol{c}_1 - \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ and $\boldsymbol{e}_2 = \boldsymbol{c}_2 - (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}$. Return $\perp$ to the adversary $\mathcal{A}$ if one of the following conditions holds:

- $\|\boldsymbol{e}_1\| > \alpha q\sqrt{\bar{m}}$, or
- $\|\boldsymbol{e}_2\| > \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty > \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$ a power of prime $b$), or
- $c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y + z \in \mathbb{F}_{2^\kappa}$.

Otherwise, return $\mu = c_3 - x \in \mathbb{F}_{2^\kappa}$ to the adversary $\mathcal{A}$.

**Challenge.** Upon receiving two challenge messages $(\mu_0, \mu_1) \in \mathbb{F}_{2^\kappa} \times \mathbb{F}_{2^\kappa}$ from the adversary $\mathcal{A}$, first randomly choose $\delta^* \xleftarrow{\$} \{0,1\}$, $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$, $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, $\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk},\gamma}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $\gamma = \sqrt{\|\boldsymbol{e}_1^*\|^2 + \bar{m}(\alpha q)^2} \cdot \omega(\sqrt{\log n})$. Then, interpret the bit-concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$, and compute

$$\begin{aligned}
&\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*), && \boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*,\\
&\boldsymbol{c}_2^* = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^*, && c_3^* = x^* + \mu_{\delta^*} \in \mathbb{F}_{2^\kappa},\\
&c_4^* = \tau^* y^* + z^* \in \mathbb{F}_{2^\kappa},
\end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*) \in \mathbb{F}_{2^\kappa}$ and $\tau^* = \mathsf{H}(\boldsymbol{c}_2^*, c_3^*) \in \mathbb{F}_{2^\kappa}$. Finally, return the challenge ciphertext $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$ to $\mathcal{A}$.

**Phase II.** Upon receiving a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$, directly return $\perp$ to the adversary $\mathcal{A}$ if $C = C^*$, otherwise answer this query as in Phase I.

By definition, we have the following lemma.

**Lemma 8.** $|\Pr[F_1] - 1/2| = \epsilon$.

**(2) Game $G_2$.** This game is similar to game $G_1$ except that the challenger $\mathcal{C}$ changes the KeyGen and Challenge phases as follows:

**Table 2** Outline of the game sequences for proving Theorem 1[a)]

| Game | Changes w.r.t. previous game | Note |
|---|---|---|
| $G_1$ | Public key: $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B} = -\boldsymbol{AR})$,<br>Secret key: $\mathrm{sk} = \boldsymbol{R}$,<br>Challenge $C^*$: $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$,<br>where<br>$\boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*$,<br>$\boldsymbol{c}_2^* = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^*$,<br>$c_3^* = x^* + \mu_{\delta^*} \in \mathbb{F}_{2^\kappa}$,<br>$c_4^* = \tau^* y^* + z^* \in \mathbb{F}_{2^\kappa}$, for some<br>$\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$,<br>$\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$,<br>$\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk},\sqrt{\|\boldsymbol{e}_1^*\|^2+\bar{m}(\alpha q)^2}\cdot\omega(\sqrt{\log n})}$,<br>$x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$,<br>$\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$,<br>$\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*)$,<br>$\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$,<br>$\tau^* = \mathsf{H}(\boldsymbol{c}_2^*, c_3)$,<br>$\delta^* \xleftarrow{\$} \{0,1\}$,<br>Decryption query $C$: run $\mathsf{Dec}(\mathrm{sk}, C)$ for any $C \neq C^*$ | Real game |
| $G_2$ | Generate $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ before giving pk to the adversary (i.e., in the KeyGen phase) | The change is conceptual: $G_2 = G_1$ |
| $G_3$ | Immediately return $\perp$ to the decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ if $\boldsymbol{c}_1 \neq \boldsymbol{c}_1^* \wedge \mathsf{H}(\boldsymbol{c}_1) = \mathsf{H}(\boldsymbol{c}_1^*)$ | By the collision resistance of $\mathsf{H}$: $G_3 \overset{c}{\approx} G_2$ |
| $G_4$ | Immediately return $\perp$ to the decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ if $(\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge \mathsf{H}(\boldsymbol{c}_2, c_3) = \mathsf{H}(\boldsymbol{c}_2^*, c_3^*)$ | By the collision resistance of $\mathsf{H}$: $G_4 \overset{c}{\approx} G_3$ |
| $G_5$ | Immediately return $\perp$ to the decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase I if $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$ | By the high min-entropy of $\boldsymbol{c}_1^*$: $G_5 \overset{s}{\approx} G_4$ |
| $G_6$ | Immediately return $\perp$ to the decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase II if $(\boldsymbol{c}_1, \boldsymbol{c}_2, c_3) = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*)$ or $\boldsymbol{c}_1 = \boldsymbol{c}_1^* \wedge (\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$ | By the unique witness of LWE (i.e., Lemma 6) and the definition of $\mathsf{Dec}$: $G_6 \overset{s}{\approx} G_5$ |
| $G_7$ | Immediately return $\perp$ to the decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase II if $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$ | By the pseudorandomness of LWE and the definition of $\mathsf{Dec}$[b)]: $G_7 \overset{c}{\approx} G_6$ |
| $G_8$ | Set $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B} = -\boldsymbol{AR}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)$ and use $\boldsymbol{R}'$ to answer the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ if $C$ does not satisfy the "immediate rejection" rules in game $G_7$ (which means that $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \neq \mathsf{H}(\boldsymbol{c}_1^*) = \mathrm{tag}^*$) | By the properties of trapdoor generation and inversion algorithms, and the definition of $\mathsf{Dec}$: $G_8 \overset{s}{\approx} G_7$ |
| $G_9$ | Use $\boldsymbol{R}'$ and $\boldsymbol{c}_1^*$ to generate $\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2' = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{e}_1^* + \boldsymbol{e}_2'$, where $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{\bar{n}k},\alpha q\sqrt{\bar{m}}\cdot\omega(\sqrt{\log n})}$ | By Lemma 3 and the definition of $\mathsf{Enc}$: $G_9 \overset{s}{\approx} G_8$ |
| $G_{10}$ | Choose $\boldsymbol{c}_1^* \xleftarrow{\$} \mathbb{Z}_q^{\bar{m}}$ at random | By the pseudorandomness of LWE: $G_{10} \overset{c}{\approx} G_9$ |
| $G_{11}$ | Choose $\boldsymbol{c}_2^* \xleftarrow{\$} \mathbb{Z}_q^{nk}$ at random | By Lemma 2: $G_{11} \overset{s}{\approx} G_{10}$ |

a) $\mu_0$ and $\mu_1$ are the challenge messages. We use $\overset{c}{\approx}$ and $\overset{s}{\approx}$ to represent the computational indistinguishability and statistical indistinguishability between two games, respectively.

b) The proof of this claim is relatively involved, and we will use the proof technique of game transitions based on failure events in [67].

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$, $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, $\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk},\gamma}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $\gamma = \sqrt{\|\boldsymbol{e}_1^*\|^2 + \bar{m}(\alpha q)^2} \cdot \omega(\sqrt{\log n})$. Then, interpret the bit-concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$, and compute

$$\begin{aligned} &\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*), && \boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*, \\ &\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}, && \boldsymbol{c}_2^* = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^*, \end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Finally, give the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to the adversary $\mathcal{A}$, keep the secret $\mathrm{sk} = \boldsymbol{R}$ and $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ secret.

**Challenge.** Upon receiving two challenge messages $(\mu_0, \mu_1) \in \mathbb{F}_{2^\kappa} \times \mathbb{F}_{2^\kappa}$ from the adversary $\mathcal{A}$, first choose a bit $\delta^* \xleftarrow{\$} \{0,1\}$ and retrieve $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$. Then, compute $c_3^* = x^* + \mu_{\delta^*} \in \mathbb{F}_{2^\kappa}, c_4^* = \tau^* y^* + z^* \in \mathbb{F}_{2^\kappa}$, where $\tau^* = \mathsf{H}(\boldsymbol{c}_2^*, c_3^*)$. Finally, return the challenge ciphertext $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$ to $\mathcal{A}$.

**Lemma 9.** Games $G_2$ and $G_1$ are identical in the adversary's view. Moreover, $\Pr[F_2] = \Pr[F_1]$.

*Proof.* This lemma follows from the fact that $(\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ is independent from the adversary's choices of the challenge messages, and game $G_2$ is essentially a conceptual change of game $G_1$ in the adversary's view.

**(3) Game $G_3$.** This game is similar to game $G_2$ except that the challenger $\mathcal{C}$ immediately returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$ if $\boldsymbol{c}_1 \neq \boldsymbol{c}_1^* \wedge \mathsf{H}(\boldsymbol{c}_1) = \mathsf{H}(\boldsymbol{c}_1^*)$.

**Lemma 10.** If $\mathsf{H}$ is a collision-resistant hash function, then games $G_3$ and $G_2$ are computationally indistinguishable. Moreover, $|\Pr[F_3] - \Pr[F_2]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* Let $\mathcal{E}$ be the event that the adversary makes a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase I such that $\boldsymbol{c}_1 \neq \boldsymbol{c}_1^* \wedge \mathsf{H}(\boldsymbol{c}_1) = \mathsf{H}(\boldsymbol{c}_1^*)$. Note that if $\mathcal{E}$ can only happen with negligible probability, then games $G_3$ and $G_2$ are computationally indistinguishable in the adversary's view. Now, we show that if there is a PPT adversary $\mathcal{A}$ that makes $\mathcal{E}$ happen with non-negligible probability, there is a PPT adversary $\mathcal{F}$ that finds a collision of $\mathsf{H}$ with the same probability by honestly simulating the attack environment for $\mathcal{A}$ as in game $G_3$. Whenever $\mathcal{A}$ outputs a ciphertext $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ such that $\boldsymbol{c}_1 \neq \boldsymbol{c}_1^* \wedge \mathsf{H}(\boldsymbol{c}_1) = \mathsf{H}(\boldsymbol{c}_1^*)$ at some time in Phase I, $\mathcal{F}$ returns the pairs $(\boldsymbol{c}_1, \boldsymbol{c}_1^*)$ as its own output and aborts. Obviously, the probability that $\mathcal{F}$ succeeds is equal to the probability that $\mathcal{A}$ makes $\mathcal{E}$ happen. Thus, under the assumption that $H$ is collision-resistant, the probability that $\mathcal{E}$ happens is negligible, which completes the proof.

**(4) Game $G_4$.** This game is similar to game $G_3$ except that the challenger $\mathcal{C}$ immediately returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$ if $(\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge \mathsf{H}(\boldsymbol{c}_2, c_3) = \mathsf{H}(\boldsymbol{c}_2^*, c_3^*)$.

**Lemma 11.** If $\mathsf{H}$ is a collision-resistant hash function, then games $G_4$ and $G_3$ are computationally indistinguishable. Moreover, $|\Pr[F_4] - \Pr[F_3]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* The proof is the same to that of Lemma 10, we omit the details.

**(5) Game $G_5$.** This game is similar to game $G_4$ except that the challenger $\mathcal{C}$ immediately returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$ in Phase I if $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$.

**Lemma 12.** Let positive integers $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0,1)$ satisfy (1) or (2). Then, games $G_5$ and $G_4$ are statistically indistinguishable. Moreover, $|\Pr[F_5] - \Pr[F_4]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* Let $\mathcal{E}$ be the event that the adversary makes a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase I such that $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$. Note that if $\mathcal{E}$ does not happen, then games $G_5$ and $G_4$ are identical in the adversary's view. Thus, it is enough to show that $\Pr[\mathcal{E}]$ is negligible for any (unbounded) adversary $\mathcal{A}$ making at most a polynomial number of decryption queries in Phase I. Note that in both games $G_4$ and $G_5$, the ciphertext part $\boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*$ is always generated by using $\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*)$ and $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, where $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}, x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$ and $\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$. By the high min-entropy of the Gaussian distribution, we have that $\boldsymbol{c}_1^*$ has min-entropy at least $\kappa$, where $\kappa$ is the security parameter. In other words, the probability that for any (unbounded) adversary to output $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$ in Phase I (i.e., before seeing $\boldsymbol{c}_1^*$) is negligible. This means that if $\mathcal{A}$ can make $\mathcal{E}$ happen with non-negligible probability, which completes the proof.

**(6) Game $G_6$.** This game is similar to game $G_5$ except that the challenger $\mathcal{C}$ immediately returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$ in Phase II if $(\boldsymbol{c}_1, \boldsymbol{c}_2, c_3) = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*)$ or

$\boldsymbol{c}_1 = \boldsymbol{c}_1^* \wedge (\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$, where $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, \boldsymbol{c}_4^*)$ is the challenge ciphertext.

**Lemma 13.** Let positive integers $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0, 1)$ satisfy (1) or (2), then games $G_6$ and $G_5$ are statistically indistinguishable. Moreover, $|\Pr[F_6] - \Pr[F_5]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* It suffices to show that the challenger $\mathcal{C}$ in game $G_5$ will always return $\perp$ to a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4) \neq C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, \boldsymbol{c}_4^*)$ from the adversary $\mathcal{A}$ in Phase II if $(\boldsymbol{c}_1, \boldsymbol{c}_2, c_3) = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*)$ or $\boldsymbol{c}_1 = \boldsymbol{c}_1^* \wedge (\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$ in Phase II except with negligible probability. Note that given a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4) \neq C^*$, the challenger $\mathcal{C}$ in game $G_5$ will first compute $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1)$, and

$$\boldsymbol{A}_{\mathrm{tag}} = (\boldsymbol{A} \| \boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b), \text{ and } \boldsymbol{u} = \begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{pmatrix}.$$

Then, compute $\tilde{\boldsymbol{s}} \leftarrow \mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}, \boldsymbol{u})$, $\boldsymbol{v} = \mathsf{decode}_d(\tilde{\boldsymbol{s}})$, and parse $\boldsymbol{v} = x\|y\|z$, where $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$. Let $\boldsymbol{e}_1 = \boldsymbol{c}_1 - \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ and $\boldsymbol{e}_2 = \boldsymbol{c}_2 - (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}$. Finally, return $\perp$ to the adversary if one of the following conditions holds:

- $\|\boldsymbol{e}_1\| > \alpha q\sqrt{\bar{m}}$, or
- $\|\boldsymbol{e}_2\| > \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty > \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$ a power of prime $b$), or
- $c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y + z \in \mathbb{F}_{2^\kappa}$.

Otherwise, return $\mu = c_3 - x \in \mathbb{F}_{2^\kappa}$.

Clearly, the challenger $\mathcal{C}$ in game $G_5$ will not return $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4) \neq C^*$ only when $\|\boldsymbol{e}_1\|_\infty \leqslant \|\boldsymbol{e}_1\| \leqslant \alpha q\sqrt{\bar{m}}$ and $c_4 = \mathsf{H}(\boldsymbol{c}_2, c_3)y + z$. In addition, given $\boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*$ for $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}}, \alpha q}$, the challenger $\mathcal{C}$ in game $G_5$ will not return $\perp$ to a decryption query $C = (\boldsymbol{c}_1 = \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4)$ only if $c_4 = \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$ except with negligible probability, since in this case we always have $(\tilde{\boldsymbol{s}}, \boldsymbol{e}_1) = (\tilde{\boldsymbol{s}}^*, \boldsymbol{e}_1^*)$ with overwhelming probability by the unique witness property in Lemma 6, which in turn implies that $\boldsymbol{v} = \boldsymbol{v}^*$ and $(x, y, z) = (x^*, y^*, z^*)$ by the correctness of $\mathsf{decode}_d$. In other words, the challenger $\mathcal{C}$ in game $G_5$ will always return $\perp$ to a decryption query $C = (\boldsymbol{c}_1 = \boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4) \neq C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, \boldsymbol{c}_4^*)$ from the adversary $\mathcal{A}$ in Phase II if $(\boldsymbol{c}_1, \boldsymbol{c}_2, c_3) = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*)$ or $\boldsymbol{c}_1 = \boldsymbol{c}_1^* \wedge (\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$ holds, except with negligible probability. This completes the proof.

**(7) Game $G_7$.** This game is similar to game $G_6$ except that the challenger $\mathcal{C}$ immediately returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$ in Phase II if $\boldsymbol{c}_1 = \boldsymbol{c}_1^*$.

Note that our goal is to show games $G_7$ and $G_6$ are computationally indistinguishable under the LWE assumption, but for technical reason it is difficult to do this in game $G_7$. Fortunately, we can still continue the game sequences by using the proof strategy (i.e., game transitions based on failure events) in [67]. Formally, for $i \in \{6, 7, 8, \ldots, 11\}$, let $E_i$ be the failure event in game $G_i$ that the adversary makes a decryption query with $C = (\boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4)$ such that $\tau = \mathsf{H}(\boldsymbol{c}_2, c_3) \neq \tau^* \wedge c_4 = \tau y^* + z^*$.

**Lemma 14.** If $E_7$ and $E_6$ do not happen, then games $G_7$ and $G_6$ are identical in the adversary's view. Moreover, $\Pr[F_7|\neg E_7] = \Pr[F_6|\neg E_6]$ and $\Pr[E_7] = \Pr[E_6]$.

*Proof.* Let $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$ be the corresponding challenge ciphertext, where $\tau^* = \mathsf{H}(\boldsymbol{c}_2^*, c_3^*)$, and $c_4^* = \tau^* y^* + z^*$ for some $y^*, z^* \in \{0, 1\}^\kappa$. Note that upon receiving a decryption query with $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ in Phase II, the challenger $\mathcal{C}$ in both games will always return $\perp$ if $(\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge \tau = \mathsf{H}(\boldsymbol{c}_2, c_3) = \tau^*$. Moreover, the challenger $\mathcal{C}$ in game $G_6$ will return $\perp$ if $(\boldsymbol{c}_1, \boldsymbol{c}_2, c_3) = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*)$ or $\boldsymbol{c}_1 = \boldsymbol{c}_1^* \wedge (\boldsymbol{c}_2, c_3) \neq (\boldsymbol{c}_2^*, c_3^*) \wedge c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y^* + z^*$ holds. In other words, the only difference between games $G_7$ and $G_6$ is that the challenger $\mathcal{C}$ in game $G_7$ also returns $\perp$ to the decryption query $C = (\boldsymbol{c}_1^*, \boldsymbol{c}_2, c_3, c_4) \neq C^*$ even if $\tau = \mathsf{H}(\boldsymbol{c}_2, c_3) \neq \mathsf{H}(\boldsymbol{c}_2^*, c_3^*) = \tau^* \wedge c_4 = \tau y^* + z^*$. Clearly, if $E_7$ and $E_6$ do not happen, then both games are identical in the adversary's view. In particular, the adversary's view in game $G_7$ before $E_7$ happens is essentially identical to that in game $G_6$. Thus, we have $\Pr[F_7|\neg E_7] = \Pr[F_6|\neg E_6]$ and $\Pr[E_7] = \Pr[E_6]$.

**(8) Game $G_8$.** This game is similar to game $G_7$ except that the challenger $\mathcal{C}$ changes the KeyGen phase and handles the decryption queries as follows:

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times \bar{m}}$, $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}}, \omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n, \alpha q}$, $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}}, \alpha q}$, $\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk}, \gamma}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $\gamma = \sqrt{\|\boldsymbol{e}_1^*\|^2 + \bar{m}(\alpha q)^2} \cdot \omega(\sqrt{\log n})$. Then, interpret the bit-

concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^* \| y^* \| z^* \in \mathbb{Z}_d^n$, and compute

$$\begin{aligned}
&\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*), && \boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*,\\
&\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b,\\
&\boldsymbol{c}_2^* = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^* = -(\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^*,
\end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Finally, give the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to the adversary $\mathcal{A}$, and keep $(\boldsymbol{R}', \boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ secret.

**Decryption query.** Upon receiving a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ from the adversary $\mathcal{A}$, return $\perp$ to $\mathcal{A}$ if this query can be immediately responded with $\perp$ using the rules in previous games. Otherwise, first set

$$\boldsymbol{A}_{\mathrm{tag}} = (\boldsymbol{A} \| \boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b), \text{ and } \boldsymbol{u} = \begin{pmatrix} \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \end{pmatrix},$$

where $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1)$. Then, compute $\tilde{\boldsymbol{s}} \leftarrow \mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}', \boldsymbol{u})$, $\boldsymbol{v} = \mathsf{decode}_d(\tilde{\boldsymbol{s}})$, and parse $\boldsymbol{v} = x\|y\|z$, where $(x, y, z) \in (\mathbb{F}_{2^\kappa})^3$. Let $\boldsymbol{e}_1 = \boldsymbol{c}_1 - \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}$ and $\boldsymbol{e}_2 = \boldsymbol{c}_2 - (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}$. Return $\perp$ to the adversary $\mathcal{A}$ if one of the following conditions holds:

- $\|\boldsymbol{e}_1\| > \alpha q\sqrt{\bar{m}}$, or
- $\|\boldsymbol{e}_2\| > \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty > \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$ a power of prime $b$), or
- $c_4 \neq \mathsf{H}(\boldsymbol{c}_2, c_3)y + z$.

Otherwise, return $\mu = c_3 - x$ to the adversary $\mathcal{A}$.

**Lemma 15.** Let positive integers $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0, 1)$ satisfy (1) or (2). Then, games $G_8$ and $G_7$ are statistically indistinguishable. Moreover, $|\Pr[F_8|\neg E_8] - \Pr[F_7|\neg E_7]| \leqslant \mathrm{negl}(\kappa)$ and $|\Pr[E_8] - \Pr[E_7]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* Note that the only differences between games $G_8$ and $G_7$ are the generation of the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ and the responses to the decryption queries. Concretely, in game $G_7$ the matrix $\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}$ is generated by using $\boldsymbol{R} \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}}, \omega(\sqrt{\log n})})^{nk}$, while in game $G_7$ the matrix $\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b$, where $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}}, \omega(\sqrt{\log n})})^{nk}$. Since $\boldsymbol{A} \in \mathbb{Z}_q^{n \times \bar{m}}$ is always uniformly chosen at random in both games, we have that $-\boldsymbol{A}\boldsymbol{R}$ and $-\boldsymbol{A}\boldsymbol{R}'$ are statistically close to uniform distribution over $\mathbb{Z}_q^{n \times nk}$ by Lemma 2. Namely, the public keys in games $G_8$ and $G_7$ are statistically close (and $\mathrm{tag}^*$ is statistically hidden in game $G_8$).

It suffices to show that in the adversary's view, the responses to the decryption queries are indistinguishable in games $G_8$ and $G_7$. Since for a decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$, the challenger will use the same rules to check if the query can be immediately responded with $\perp$ in both games, we only have to consider the decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$ that needs the challenger $\mathcal{C}$ to perform the decryption operation. By the definition of game $G_7$, we must have that $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \neq \mathrm{tag}^*$ holds for such decryption query $C = (\boldsymbol{c}_1, \boldsymbol{c}_2, c_3, c_4)$. Note that in game $G_7$, the challenger has the real secret key $\mathrm{sk} = \boldsymbol{R}$, and can run the decryption algorithm to handle this query. We now show that the challenger $\mathcal{C}$ in game $G_8$ can almost perfectly simulate the decryption operation. Recall that $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)$, conditioned on $\mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1) \neq \mathrm{tag}^*$ we have that $\boldsymbol{R}'$ is a valid trapdoor for $\boldsymbol{A}_{\mathrm{tag}} = (\boldsymbol{A}\|\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)$, and thus can be used to compute $\tilde{\boldsymbol{s}} \leftarrow \mathsf{Solve}(\boldsymbol{A}_{\mathrm{tag}}, \boldsymbol{R}', \boldsymbol{u})$. Now, either there exists a tuple $(\tilde{\boldsymbol{s}}, \boldsymbol{e}_1, \boldsymbol{e}_2)$ such that $\|\boldsymbol{e}_1\| \leqslant \alpha q\sqrt{\bar{m}}$, $\|\boldsymbol{e}_2\| \leqslant \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty \leqslant \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$), $\boldsymbol{c}_1 = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_1$ and $\boldsymbol{c}_2 = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_2$, or there does not. For the latter case, the challenger will always return $\perp$ in both games. While for the former case, the challenger $\mathcal{C}$ in game $G_8$ can recover $\tilde{\boldsymbol{s}}$ as long as $\|(\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\| < \frac{q}{2\sqrt{b^2+1}}$ for prime $q$ (or $\|(\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{e}_1 + \boldsymbol{e}_2\|_\infty < \frac{q}{2b}$ for $q = b^k$), which is essentially the same constraint for a correct decryption using $\mathrm{sk} = \boldsymbol{R}$ in game $G_7$. Since both $\boldsymbol{R}$ and $\boldsymbol{R}'$ are chosen from the same Gaussian distribution, by Lemma 1 we have that the inequality holds with the same overwhelming probability conditioned on $\|\boldsymbol{e}_1\| \leqslant \alpha q\sqrt{m}$ and $\|\boldsymbol{e}_2\| \leqslant \gamma\sqrt{nk}$ for prime $q$ (or $\|\boldsymbol{e}_2\|_\infty \leqslant \gamma \cdot \omega(\sqrt{\log n})$ for $q = b^k$). By the fact that the challengers in both games will perform the same operations after obtaining $\tilde{\boldsymbol{s}}$, we have that the responses to such kind of decryption queries are identical in both games except with negligible probability. This finishes the proof.

**Remark 1.** Note that the challenger in game $G_8$ actually does not have the "real" secret key, which implies that the adversary cannot obtain extra information about the secret key from the decryption queries (except what is obtained from the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$). This fact will be used in our later proofs.

**(9) Game $G_9$.** This game is similar to game $G_8$ except that the challenger $\mathcal{C}$ changes the KeyGen phase as follows:

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$ and $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}, \boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},r}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $r = \alpha q\sqrt{\bar{m}}\cdot\omega(\sqrt{\log n})$. Then, interpret the bit-concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$, and compute

$$\begin{aligned} &\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*), && \boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*,\\ &\boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b, && \boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2', \end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Finally, give the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to $\mathcal{A}$, and keep $(\boldsymbol{R}', \boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ secret.

**Lemma 16.** Let positive integers $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0,1)$ satisfy (1) or (2). Then, games $G_9$ and $G_8$ are statistically indistinguishable. Moreover, $|\Pr[F_9|\neg E_9] - \Pr[F_8|\neg E_8]| \leqslant \mathrm{negl}(\kappa)$ and $|\Pr[E_9] - \Pr[E_8]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* Note that the only difference between games $G_9$ and $G_8$ is the generation of $\boldsymbol{c}_2^*$. In game $G_8$, $\boldsymbol{c}_2^* = (\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^* = -(\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_2^*$ is generated by using $\boldsymbol{e}_2^* \xleftarrow{\$} D_{\mathbb{Z}^{nk},\gamma}$ where $\gamma = \sqrt{\|\boldsymbol{e}_1^*\|^2 + \bar{m}(\alpha q)^2}\cdot\omega(\sqrt{\log n})$, while in game $G_9$, $\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2'$ is generated by using $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},r}$. Since $\boldsymbol{c}_1^* = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}_1^*$ for some $\boldsymbol{e}_1^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, we have that $\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2' = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \tilde{\boldsymbol{e}}_2$ for some $\tilde{\boldsymbol{e}}_2 = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{e}_1^* + \boldsymbol{e}_2'$ which is distributed statistically close to $D_{\mathbb{Z}^{nk},\gamma}$ by applying Lemma 3 $nk$ times using a standard hybrid argument. Thus, the distributions of $\boldsymbol{c}_2^*$ in games $G_9$ and $G_8$ are actually statistically close, which in turn shows that both games are statistically indistinguishable in the adversary's view.

**(10) Game $G_{10}$.** This game is similar to game $G_9$ except that the challenger $\mathcal{C}$ changes the KeyGen phase as follows:

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}, \boldsymbol{b} \xleftarrow{\$} \mathbb{Z}_q^{m_1}$, $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},r}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $r = \alpha q\sqrt{\bar{m}}\cdot\omega(\sqrt{\log n})$. Then, compute

$$\begin{aligned} &\boldsymbol{c}_1^* = \boldsymbol{b}, && \boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b,\\ &\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2', \end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Finally, give the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to $\mathcal{A}$, and keep $(\boldsymbol{R}', \boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ secret.

**Lemma 17.** If $\mathrm{LWE}_{n,\bar{m},q,\alpha}$ is hard, then games $G_{10}$ and $G_9$ are computationally indistinguishable. Moreover, $|\Pr[F_{10}|\neg E_{10}] - \Pr[F_9|\neg E_9]| \leqslant \mathrm{negl}(\kappa)$ and $|\Pr[E_{10}] - \Pr[E_9]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* We prove this lemma by showing that if there is a PPT adversary $\mathcal{A}$ that distinguishes game $G_{10}$ from $G_9$ with non-negligible advantage, then there is an efficient algorithm $\mathcal{B}$ that solves the $\mathrm{LWE}_{n,\bar{m},q,\alpha}$ problem with the same advantage by interacting with $\mathcal{A}$.

Formally, given an LWE challenge tuple $(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{Z}_q^{n\times\bar{m}} \times \mathbb{Z}_q^{\bar{m}}$, $\mathcal{B}$ randomly chooses $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{e}_2' \xleftarrow{\$} D_{\mathbb{Z}^{nk},r}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$, where $r = \alpha q\sqrt{\bar{m}}\cdot\omega(\sqrt{\log n})$. Then, it interprets the bit-concatenation of $(x^*, y^*, z^*) \in (\mathbb{F}_{2^\kappa})^3$ as a vector $\boldsymbol{v}^* = x^*\|y^*\|z^* \in \mathbb{Z}_d^n$, and computes

$$\begin{aligned} &\boldsymbol{c}_1^* = \boldsymbol{b} + \boldsymbol{A}^{\mathrm{T}}\mathsf{encode}_d(\boldsymbol{v}^*), && \boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b,\\ &\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2', \end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Then, $\mathcal{B}$ sets the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$, and keeps $(\boldsymbol{R}', \boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ private. Finally, $\mathcal{B}$ gives pk to the adversary $\mathcal{A}$, simulates the attack environment the same as in game $G_9$, and returns whatever $\mathcal{A}$ outputs as its own output.

Now, if $(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{Z}_q^{n\times\bar{m}} \times \mathbb{Z}_q^{\bar{m}}$ is a valid LWE tuple, i.e., $\boldsymbol{b} = \boldsymbol{A}^{\mathrm{T}}\boldsymbol{s}^* + \boldsymbol{e}^*$ for some $\boldsymbol{s}^* \xleftarrow{\$} D_{\mathbb{Z}^n,\alpha q}$ and $\boldsymbol{e}^* \xleftarrow{\$} D_{\mathbb{Z}^{\bar{m}},\alpha q}$, then we have that $\boldsymbol{c}_1^* = \boldsymbol{b} + \boldsymbol{A}^{\mathrm{T}}\mathsf{encode}_d(\boldsymbol{v}) = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}}^* + \boldsymbol{e}^*$, where $\tilde{\boldsymbol{s}}^* = \boldsymbol{s}^* + \mathsf{encode}_d(\boldsymbol{v}^*)$. In

this case, $\mathcal{B}$ perfectly simulates the attack environment in game $G_9$ for $\mathcal{A}$. Else if $(\boldsymbol{A}, \boldsymbol{b}) \in \mathbb{Z}_q^{n\times\bar{m}} \times \mathbb{Z}_q^{\bar{m}}$ is uniformly random, then $\boldsymbol{c}_1^* = \boldsymbol{b} + \boldsymbol{A}^{\mathrm{T}}\mathsf{encode}_d(\boldsymbol{v}^*)$ is also uniformly random over $\mathbb{Z}_q^{\bar{m}}$. This means that $\mathcal{B}$ perfectly simulates the attack environment in game $G_{10}$ for $\mathcal{A}$. Thus, if $\mathcal{A}$ can distinguish game $G_{10}$ from $G_9$ with non-negligible advantage, then $\mathcal{B}$ can solve the $\mathrm{LWE}_{n,\bar{m},q,\alpha}$ problem with the same advantage.

**(11) Game $G_{11}$.** This game is similar to game $G_{10}$ except that the challenger $\mathcal{C}$ changes the KeyGen phase as follows:

**KeyGen.** First randomly choose $\boldsymbol{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, $\boldsymbol{b} \xleftarrow{\$} \mathbb{Z}_q^{\bar{m}}, \boldsymbol{d} \xleftarrow{\$} \mathbb{Z}_q^{nk}$ and $x^*, y^*, z^* \xleftarrow{\$} \mathbb{F}_{2^\kappa}$. Then, compute

$$\begin{aligned} &\boldsymbol{c}_1^* = \boldsymbol{b}, \qquad \boldsymbol{B} = -\boldsymbol{A}\boldsymbol{R}' - \mathsf{FRD}(\mathrm{tag}^*)\boldsymbol{G}_b, \\ &\boldsymbol{c}_2^* = \boldsymbol{d}, \end{aligned}$$

where $\mathrm{tag}^* = \mathsf{H}(\boldsymbol{c}_1^*)$. Finally, give the public key $\mathrm{pk} = (\boldsymbol{A}, \boldsymbol{B})$ to $\mathcal{A}$, and keep $(\boldsymbol{R}', \boldsymbol{c}_1^*, \boldsymbol{c}_2^*, x^*, y^*, z^*)$ secret.

**Lemma 18.** Let positive $n, \bar{m}, b, d, q \in \mathbb{Z}$ and real $\alpha \in (0,1)$ satisfy (1) or (2), then games $G_{11}$ and $G_{10}$ are statistically indistinguishable. Moreover, $|\Pr[F_{11}|\neg E_{11}] - \Pr[F_{10}|\neg E_{10}]| \leqslant \mathrm{negl}(\kappa)$ and $|\Pr[E_{11}] - \Pr[E_{10}]| \leqslant \mathrm{negl}(\kappa)$.

*Proof.* Note that the only difference between games $G_{11}$ and $G_{10}$ is the generation of $\boldsymbol{c}_2^*$ in the challenge ciphertext. Thus, it is enough to show that $\boldsymbol{c}_2^*$ in game $G_{11}$ is actually statistically close to that in game $G_{10}$. Note that $\boldsymbol{c}_1^* = \boldsymbol{b}$ is uniformly chosen from $\mathbb{Z}_q^{\bar{m}}$ at random in both games, and $\boldsymbol{c}_2^* = (-\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{c}_1^* + \boldsymbol{e}_2'$ in game $G_{10}$. Using the facts that $\bar{m} \geqslant (n+1)\log_2 q + \omega(\log n)$ and $\boldsymbol{R}' \xleftarrow{\$} (D_{\mathbb{Z}^{\bar{m}},\omega(\sqrt{\log n})})^{nk}$, we have that $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{R}', \boldsymbol{b}, (\boldsymbol{R}')^{\mathrm{T}}\boldsymbol{b})$ is statistically close to uniform by Lemma 2. In other words, $\boldsymbol{c}_2^*$ in game $G_{10}$ is essentially statistically close to uniform over $\mathbb{Z}_q^{nk}$, which completes the proof.

**Lemma 19.** $\Pr[F_{11}] = 1/2$ and $\Pr[E_{11}] = \mathrm{negl}(\kappa)$.

*Proof.* Since $x^* \xleftarrow{\$} \{0,1\}^\ell$ is uniformly chosen at random, $\mu_{\delta^*} \in \{0,1\}^\ell$ is perfectly hidden in the challenge ciphertext $C^* = (\boldsymbol{c}_1^*, \boldsymbol{c}_2^*, c_3^*, c_4^*)$, where $c_3^* = x^* + \mu_{\delta^*} \in \mathbb{F}_{2^\kappa}$. Thus, $\Pr[F_{11}] = \Pr[\delta = \delta^*] = 1/2$, where $\delta \in \{0,1\}$ is output by the adversary $\mathcal{A}$ for the guess of $\delta^*$ in game $G_{11}$.

As for the second claim, since $y^*, z^* \in \{0,1\}^\kappa$ are uniformly chosen at random in game $G_{11}$, given $c_4^* = \tau^* y^* + z^* \in \mathbb{F}_{2^\kappa}$ there are still $2^\kappa$ possible choices of $(y^*, z^*)$. Thus, for any adversary $\mathcal{A}$ with the knowledge of $c_4^* = \tau^* y^* + z^*$, the probability that it outputs $c_4 = \tau y^* + z^*$ for any $\tau \neq \tau^*$ is at most $1/2^\kappa$ (because the adversary can uniquely determine $(y^*, z^*)$ if he can output a valid $c_4 = \tau y^* + z^*$), which means that $\Pr[E_{11}] \leqslant Q_{\mathrm{dec}}/2^\kappa$, where $Q_{\mathrm{dec}}$ is the maximum number of decryption queries made by $\mathcal{A}$.

In all, we have that $\Pr[F_1] \leqslant 1/2 + \mathrm{negl}(\kappa)$ by Lemmas 9–19. This completes the proof of Theorem 1.

## 4 Optimizations

### 4.1 Encrypting long message

In the description of our PKE, we only consider to encrypt a $\kappa$-bit message for simplicity. Although it suffices for many applications where a PKE is typically used to encrypt a session key for some symmetric encryption such as AES, our PKE can essentially encrypt messages of bit length up to $O(n\log n) - 2\kappa$. Note that in order to recover $\boldsymbol{v} \in \mathbb{Z}_d^n$ from $\tilde{\boldsymbol{s}} = \boldsymbol{s} + \mathsf{encode}_d(\boldsymbol{v}) \in \mathbb{Z}_q^n$, it is enough to set the parameters such that $\|\boldsymbol{s}\|_\infty < \frac{q-(d-1)d}{2d}$ by Lemma 7. For any $n, \bar{m}, q \in \mathbb{Z}$ and real $\alpha \in \mathbb{R}$ satisfying (1) or (2), we always have $\|\boldsymbol{s}\|_\infty \leqslant \alpha q \cdot \omega(\sqrt{\log n}) < q/\tilde{O}(\sqrt{n})$. This makes it possible to set $d = \tilde{O}(\sqrt{n})$ and encrypt messages of bit length up to $O(n\log n) - 2\kappa$ (since one can encode $n\log_2 d = O(n\log n)$-bit information into the vector $\boldsymbol{v} \in \mathbb{Z}_d^n$). Concretely, for the choice of $(n, m, q, \alpha q) = (450, 10740, 3^9, 1.5)$, we have that $\|\boldsymbol{s}\|_\infty \leqslant 9$ holds except with probability less than $2^{-162}$, which allows us to set $d = 128$ and encrypt messages of size up to 2638-bit. We also note that one can use a pseudorandom generator $\mathsf{PRG}$ to encrypt any polynomial long messages by replacing $c_3 = x + \mu$ with $c_3 = \mathsf{PRG}(x) \oplus \mu$.

### 4.2 Compressing the ciphertext

As many lattice-based PKEs in the literatures (e.g., [29,55,68]), it is possible to discard some lower bits of the ciphertext (and thus reduce the ciphertext size) without affecting the correctness of the PKEs (because those lower bits mainly carry noise). This can be seen as a modulus switch technique. Concretely, let $\mathsf{Switch}_{q,p}(\cdot) : \mathbb{Z}_q \to \mathbb{Z}_p$ be a function defined as $\mathsf{Switch}_{q,p}(x) = \lceil p/q \cdot x \rfloor \mod p$. It is easy to check that for any $x \in \mathbb{Z}_q$, $x - \mathsf{Switch}_{p,q}(\mathsf{Switch}_{q,p}(x)) \leqslant \lceil \frac{q}{2p} \rfloor$. Thus, for $p < q$, one can use $\mathsf{Switch}_{q,p}(\cdot)$ to compress the ciphertext in the encryption algorithm (i.e., applying to vectors in a coordinate-wise way), and use $\mathsf{Switch}_{p,q}(\cdot)$ to approximately recover the original ciphertext in the decryption algorithm. This can be simply seen as adding a noise of size at most $\lceil \frac{q}{2p} \rfloor$ to each coordinate of the lattice vectors in the ciphertext.

In our case, we cannot simply apply $\mathsf{Switch}_{q,p}(\cdot)$ to the ciphertext in a black-box way, since this will affect both the correctness and the security of our PKE. Instead, we have to plug it into the encryption algorithm to generate the ciphertext $C = (\boldsymbol{c}_1, \boldsymbol{c}'_2, c_3, c'_4)$ as follows (where $p$ is an integer, and other notations are the same as before):

$$\begin{aligned} &\boldsymbol{c}_1 = \boldsymbol{A}^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_1, && \mathrm{tag} = \mathsf{H}(\boldsymbol{c}_1),\\ &\boldsymbol{c}'_2 = \mathsf{Switch}_{q,p}((\boldsymbol{B} + \mathsf{FRD}(\mathrm{tag})\boldsymbol{G}_b)^{\mathrm{T}}\tilde{\boldsymbol{s}} + \boldsymbol{e}_2), && c_3 = x + \mu,\\ &\tau' = \mathsf{H}(\boldsymbol{c}'_2, c_3), && c'_4 = \tau' y + z. \end{aligned}$$

For the choice of $(n, m, q, \alpha q) = (450, 10740, 3^9, 1.5)$, we can set $p = 8$ to compress the ciphertext from previous 19.73 KB to 13.80 KB, while still keep the decryption error rate less than $2^{-100}$. Note that we do not use this technique to compress $\boldsymbol{c}_1$, because unlike the error in $\boldsymbol{c}_2$, any error in $\boldsymbol{c}_1$ will be sharply amplified by a factor of $s_1(\boldsymbol{R})$ in decryption.

### 4.3 Compressing the keys

The key sizes of LWE-based PKEs (e.g., [18, 29, 30, 33, 40]) are usually very large due to consist of big matrices in both the public keys and the secret keys. For example, under the choice of $(n, m, q, \alpha q) = (450, 10740, 3^9, 1.5)$, the public key and secret key sizes of our PKE are about 8.64 and 16.15 MB, respectively. However, the first element in the public key is essentially a uniformly random matrix which can be treated as a system parameter and shared among all users. By doing this, one can reduce the sizes of the public key from previous 8.64 to 3.26 MB. Besides, one can also use a PRG with a 256-bit random seed to deterministically generate the secret key matrix $\boldsymbol{R}$, and reduce the secret key size from 16.15 MB to 32 Bytes. As we will show in the supplemental material, one can also reduce the key sizes by adapting our construction to the ring setting.

**Acknowledgements** Jiang ZHANG is supported by National Key Research and Development Program of China (Grant Nos. 2017YFB0802005, 2018YFB0804105), National Natural Science Foundation of China (Grant No. 61602046), Young Elite Scientists Sponsorship Program by CAST (Grant No. 2016QNRC001), and Opening Project of Guangdong Provincial Key Laboratory of Data Security and Privacy Protection (Grant No. 2017B030301004). Yu YU is supported by National Natural Science Foundation of China (Grant Nos. 61872236, 61572192), National Cryptography Development Fund (Grant No. MMJJ20170209), and Anhui Initiative in Quantum Information Technologies (Grant No. AHY150100). Shuqin FAN is supported by National Key Research and Development Program of China (Grant No. 2017YFB0802005). Zhenfeng ZHANG is supported by National Key Research and Development Program of China (Grant No. 2017YFB0802005) and National Natural Science Foundation of China (Grant No. U1536205).

**References**

1 Diffie W, Hellman M. New directions in cryptography. IEEE Trans Inform Theor, 1976, 22: 644–654
2 Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM, 1978, 21: 120–126
3 Goldwasser S, Micali S. Probabilistic encryption. J Comput Syst Sci, 1984, 28: 270–299

4 Naor M, Yung M. Public-key cryptosystems provably secure against chosen ciphertext attacks. In: Proceedings of the 22nd Annual ACM Symposium on Theory of Computing, Baltimore, 1990. 427–437

5 Rackoff C, Simon D R. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In: Advances in Cryptology–CRYPTO'91. Berlin: Springer, 1992. 433–444

6 NIST. Post-quantum cryptography standardization. 2016. http://csrc.nist.gov/groups/ST/post-quantum-crypto/submission-requirements/index.html

7 Fujisaki E, Okamoto T. Secure integration of asymmetric and symmetric encryption schemes. J Cryptol, 2013, 26: 80–101

8 Pointcheval D. Chosen-ciphertext security for any one-way cryptosystem. In: Public Key Cryptography. Berlin: Springer, 2000. 129–146

9 Targhi E E, Unruh D. Post-quantum security of the Fujisaki-Okamoto and OAEP transforms. In: Theory of Cryptography. Berlin: Springer, 2016. 192–216

10 Canetti R, Goldreich O, Halevi S. The random oracle methodology, revisited. J ACM, 2004, 51: 557–594

11 Gertner Y, Malkin T, Myers S. Towards a separation of semantic and CCA security for public key encryption. In: Theory of Cryptography. Berlin: Springer, 2007. 434–455

12 Sahai A. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In: Proceedings of the 40th Annual Symposium on Foundations of Computer Science, New York City, 1999. 543–553

13 Dolev D, Dwork C, Naor M. Non-malleable cryptography. SIAM J Comput, 2000, 30: 391–437

14 Cramer R, Shoup V. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. SIAM J Comput, 2001, 33: 167–226

15 Wee H. Efficient chosen-ciphertext security via extractable hash proofs. In: Proceedings of the 30th Annual Conference on Advances in Cryptology. Berlin: Springer, 2010. 314–332

16 Boneh D, Canetti R, Halevi S, et al. Chosen-ciphertext security from identity-based encryption. SIAM J Comput, 2006, 36: 1301–1328

17 Kiltz E. Chosen-ciphertext security from tag-based encryption. In: Theory of Cryptography. Berlin: Springer, 2006. 581–600

18 Peikert C, Waters B. Lossy trapdoor functions and their applications. In: Proceedings of STOC 2008. New York: ACM, 2008. 187–196

19 Rosen A, Segev G. Chosen-ciphertext security via correlated products. In: Theory of Cryptography. Berlin: Springer, 2009. 419–436

20 Kiltz E, Mohassel P, O'Neill A. Adaptive trapdoor functions and chosen-ciphertext security. In: Advances in Cryptology–EUROCRYPT 2010. Berlin: Springer, 2010. 673–692

21 Shor P W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J Comput, 1997, 26: 1484–1509

22 Cramer R, Shoup V. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In: Advances in Cryptology–CRYPTO'98. Berlin: Springer, 1998. 13–25

23 Katz J, Vaikuntanathan V. Smooth projective hashing and password-based authenticated key exchange from lattices. In: Advances in Cryptology–ASIACRYPT 2009. Berlin: Springer, 2009. 636–652

24 Peikert C, Vaikuntanathan V, Waters B. A framework for efficient and composable oblivious transfer. In: Advances in Cryptology–CRYPTO 2008. Berlin: Springer, 2008. 554–571

25 Benhamouda F, Blazy O, Ducas L, et al. Hash proof systems over lattices revisited. In: Public-Key Cryptography–PKC 2018. Berlin: Springer, 644–674

26 Han G, Li H, Qin B D, et al. Chameleon all-but-one extractable hash proof and its applications. Sci China Inf Sci, 2018, 61: 099103

27 Zhang J, Yu Y. Two-round pake from approximate SPH and instantiations from lattices. In: Advances in Cryptology–ASIACRYPT 2017. Berlin: Springer, 2017. 37–67

28 Kim S, Wu D J. Multi-theorem preprocessing NIZKs from lattices. In: Advances in Cryptology–CRYPTO 2018. Berlin: Springer, 2018. 733–765

29 Peikert C. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In: Proceedings of the Annual ACM Symposium on Theory of Computing, Bethesda, 2009. 333–342

30 Wee H. Public key encryption against related key attacks. In: Public Key Cryptography–PKC 2012. Berlin: Springer, 2012. 262–279

31 Steinfeld R, Ling S, Pieprzyk J, et al. NTRUCCA: how to strengthen ntruencrypt to chosen-ciphertext security in the standard model. In: Public Key Cryptography–PKC 2012. Berlin: Springer, 2012. 353–371

32 Dowsley R, Hanaoka G, Imai H, et al. Reducing the ciphertext size of Dolev-Dwork-Naor like public key cryptosystems. Cryptology ePrint Archive, Report 2009/271, 2009

33 Agrawal S, Boneh D, Boyen X. Efficient lattice (H)IBE in the standard model. In: Advances in Cryptology–EUROCRYPT 2010. Berlin: Springer, 2010. 553–572

34 Agrawal S, Boneh D, Boyen X. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In: Advances in Cryptology–CRYPTO 2010. Berlin: Springer, 2010. 98–115

35 Cash D, Hofheinz D, Kiltz E, et al. Bonsai trees, or how to delegate a lattice basis. In: Advances in Cryptology–EUROCRYPT 2010. Berlin: Springer, 2010. 523–552

36 Yamada S. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In: Advances in Cryptology–EUROCRYPT 2016. Berlin: Springer, 2016. 32–62

37 Zhang J, Chen Y, Zhang Z. Programmable hash functions from lattices: short signatures and IBEs with small key sizes. In: Advances in Cryptology–CRYPTO 2016. Berlin: Springer, 2016. 303–332

38 Yamada S. Asymptotically compact adaptively secure lattice IBEs and verifiable random functions via generalized partitioning techniques. In: Advances in Cryptology–CRYPTO 2017. Berlin: Springer, 2017. 161–193

39 Döttling N, Garg S, Hajiabadi M, et al. New constructions of identity-based and key-dependent message secure encryption schemes. In: Public-Key Cryptography–PKC 2018. Berlin: Springer, 2018. 3–31

40 Micciancio D, Peikert C. Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Advances in Cryptology–EUROCRYPT 2012. Berlin: Springer, 2012. 700–718

41 Canetti R, Halevi S, Katz J. Chosen-ciphertext security from identity-based encryption. In: Proceedings of EUROCRYPT 2004. Berlin: Springer, 2004. 207–222

42 Boneh D, Katz J. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In: Topics in Cryptology–CT-RSA 2005. Berlin: Springer, 2005. 87–103

43 Lyubashevsky V, Micciancio D. Asymptotically efficient lattice-based digital signatures. J Cryptol, 2018, 31: 774–797

44 Albrecht M R, Player R, Scott S. On the concrete hardness of learning with errors. J Math Cryptol, 2015, 9: 169–203

45 Ajtai M. Generating hard instances of the short basis problem. In: Automata, Languages and Programming. Berlin: Springer, 1999. 706

46 Gentry C, Peikert C, Vaikuntanathan V. Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the 40th Annual ACM Symposium on Theory of Computing, Victoria, 2008. 197–206

47 Regev O. On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing. New York: ACM, 2005. 84–93

48 Applebaum B, Cash D, Peikert C, et al. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Advances in Cryptology–CRYPTO 2009. Berlin: Springer, 2009. 595–618

49 Lindner R, Peikert C. Better key sizes (and attacks) for LWE-based encryption. In: Topics in Cryptology–CT-RSA 2011. Berlin: Springer, 2011. 6558: 319–339

50 Bos J, Costello C, Ducas L, et al. Frodo: take off the ring! practical, quantum-secure key exchange from LWE. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security. New York: ACM, 2016. 1006–1018

51 Stehlé D, Steinfeld R, Tanaka K, et al. Efficient public key encryption based on ideal lattices. In: Advances in Cryptology – ASIACRYPT 2009. Berlin: Springer, 2009. 617–635

52 Lyubashevsky V, Peikert C, Regev O. On ideal lattices and learning with errors over rings. In: Advances in Cryptology–EUROCRYPT 2010. Berlin: Springer, 2010. 1–23

53 Stehlé D, Steinfeld R. Making NTRU as secure as worst-case problems over ideal lattices. In: Advances in Cryptology–EUROCRYPT 2011. Berlin: Springer, 2011. 27–47

54 Alkim E, Ducas L, Pöppelmann T, et al. Post-quantum key exchange-a new hope. In: Proceedings of the 25th USENIX Security Symposium, Austin, 2016

55 Alkim E, Ducas L, Pöppelmann T, et al. Newhope Without Reconciliation. Cryptology ePrint Archive, Report 2016/1157, 2016

56 Boneh D, Dagdelen Ö, Fischlin M, et al. Random oracles in a quantum world. In: Advances in Cryptology–ASIACRYPT 2011. Berlin: Springer, 2011. 41–69

57 Jiang H, Zhang Z, Chen L, et al. IND-CCA-secure key encapsulation mechanism in the quantum random oracle model, revisted. In: Advances in Cryptology – CRYPTO 2018. Berlin: Springer, 2018. 96–125

58 Saito T, Xagawa K, Yamakawa T. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In: Advances in Cryptology – EUROCRYPT 2018. Berlin: Springer, 2018. 520–551

59 Libert B, Sakzad A, Stehlé D, et al. All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from LWE. In: Advances in Cryptology – CRYPTO 2017. Berlin: Springer, 2017. 332–364

60 Banaszczyk W. New bounds in some transference theorems in the geometry of numbers. Math Ann, 1993, 296: 625–635

61 Peikert C. An efficient and parallel Gaussian sampler for lattices. In: Advances in Cryptology–CRYPTO 2010. Berlin: Springer, 2010. 80–97

62 Ducas L, Micciancio D. Improved short lattice signatures in the standard model. In: Advances in Cryptology–CRYPTO 2014. Berlin: Springer, 2014. 335–352
63 Vershynin R. Introduction to the non-asymptotic analysis of random matrices. 2010. ArXiv: 10113027
64 Peikert C, Regev O, Stephens-Davidowitz N. Pseudorandomness of ring-LWE for any ring and modulus. In: STOC 2017. ACM, 2017. 461–473
65 Alwen J, Peikert C. Generating shorter bases for hard random lattices. In: Proceedings of STACS, 2009. 75–86
66 Cramer R, Damgård I. On the amortized complexity of zero-knowledge protocols. In: Proceedings of CRYPTO 2009. Berlin: Springer, 2009. 177–191
67 Shoup V. Sequences of Games: a Taming Complexity in Security Proofs. Cryptology ePrint Archive, Report 2004/332, 2004
68 Brakerski Z, Vaikuntanathan V. Efficient fully homomorphic encryption from (standard) LWE. In: Proceedings of the 2011 IEEE 52nd Annual Symposium on Foundations of Computer Science. Washington: IEEE Computer Society, 2011. 97–106