All-But-Many Lossy Trapdoor Functions from Lattices and Applications

Xavier Boyen *, Qinyi Li

Queensland University of Technology, Brisbane, Australia

Abstract. "All-but-many lossy trapdoor functions" (ABM-LTF) are a powerful cryptographic primitive studied by Hofheinz (Eurocrypt 2012). ABM-LTFs are parametrised with tags: a lossy tag makes the function lossy; an injective tag makes the function injective, and invertible with a trapdoor. Existing ABM-LTFs rely on non-standard assumptions.

Our first result is an ABM-LTF construction from lattices, based on the learning-with-errors (LWE) problem. Unlike the previous schemes which behaved as "encrypted signatures", the core of our construction is an "encrypted, homomorphic-evaluation-friendly, weak pseudorandom function". The weak pseudorandom function outputs matrices, where the lossy tags are preimages of the zero matrix, and the injective tags are preimages of random full-rank matrices.

Our second result is a public-key system tightly secure against "selective opening" attacks, where an attacker gets many challenges and can ask to see the random bits of any of them. Following the steps of Hemenway et al. (Asiacrypt 2011) and Hofheinz (Eurocrypt 2012), our ABM-LTF gives the first lattice-based, compact public-key encryption (PKE) scheme that has indistinguishability against adaptive chosen-ciphertext and selective opening attacks (IND-SO-CCA2), with tight security, and whose public-key size and security reduction are independent of the number of decryption queries and ciphertext challenges.

Meanwhile, this result provides an alternative solution to the problem of building pairing-free IND-CCA2 PKE schemes with tight security in the multi-challenge setting, which was firstly answered by Gay et al. (Eurocrypt 2016). Additionally, our ABM-LTF answers the open question of constructing (non-necessarily lossy) all-but-many trapdoor functions from lattices, first asked by Alperin-Sheriff and Peikert (PKC 2012).

1 Introduction

All-but-many lossy trapdoor functions (ABM-LTF) are a useful cryptographic primitive formalised by Hofheinz [29]. ABM-LTFs generalise lossy trapdoor functions (LTFs) [39], all-but-one lossy trapdoor functions (ABO-LTFs) [39], and all-but-N lossy trapdoor functions (ABN-LTFs) [27]. ABM-LTF have shown their usefulness in constructiing public-key encryption schemes with strong security

* Research supported in part by ARC Discovery Project grant number DP140103885 and ARC Future Fellowship FT140101145 from the Australian Research Council.

properties including selective opening security, e.g, [29], key-dependent message security, e.g., [30] and key leakage resilience, e.g., [40].

An ABM-LTF is a function described by public evaluation parameters and parametrised by a tag from some set. The tag set consists of two disjoint super-polynomially large subsets: the set of injective tags and the set of lossy tags. An injective tag makes the function injective and, hence, invertible with trapdoors. A lossy tag makes the function lossy meaning that the function looses information of its inputs and, therefore, can not be inverted in the information-theoretical sense (except negligible probability). Note that there *could* exist a spurious set of invalid tags, that make the function injective yet disable its trapdoor invertibility: in our construction we need to avoid this possibility. An ABM-LTF is equipped with two trapdoors: one is the inversion trapdoor which allows one to correctly invert the function in case of the tag is injective; the other is a lossy tags generation trapdoor which allows security reduction to generate lossy tags.

ABM-LTFs have two main security properties. The first one, "lossy-tag indistinguishability", guarantees that a lossy tag is computationally indistinguishable from a random tag, even given access to the lossy tag generation oracle. The second one, "evasiveness", prevents efficient adversaries from generating lossy tags (notice that this implies that a random tag is an injective tag w.h.p.). Theses two security properties make ABM-LTFs particularly useful for handling adaptive attacks in the multi-challenge setting, in which adversaries are able to obtain multiple challenge targets (e.g., challenge ciphertext). For instance, evasiveness forces that all adaptive queries be made with injective tags, enabling inversion trapdoors in security reductions. Indistinguishability allows security reductions to use multiple lossy tags for creating multiple challenges embedding the same computational problem, without tipping off adversaries.

Constructions of ABM-LTFs. Not very surprisingly, with such powerful properties, ABM-LTFs have more complicated constructions than its simpler counterparts, say plain LTFs. So far, essentially two types of constructions of ABM-LTFs exist. The first type is based on Paillier/Damgard-Jurik encryption [37,19] together with some non-standard assumptions, and first instantiated by Hofheinz [29] and latter improved by Fujisaki [23]. The second type, based on subgroup indistinguishable problems over composite-order bilinear groups, was design by Hofheinz [29]. Though relying on different assumptions and algebraic structures, the two types of constructions share the same flavour at a conceptual level. Both of them can be seen as "encrypted signature" schemes in which a lossy tag corresponds to a valid (but disguised) signature. Existential unforgeability of signatures guarantees the evasiveness. Tag indistinguishability is provided by the semantic security of Paillier/Damgard-Jurik encryption or hardness of subgroup decisional problems. Roughly, the two types of construction utilise either additive homomorphism of Paillier/Damgard-Jurik ciphertexts, or group exponentiation operations, to conduct the lossy trapdoor function evaluations. Apart from the elegance of existing constructions, one of their disadvantages is their need for non-standard assumptions. Thus, a first motivation for our present work is to

solve the open problem of finding different constructions of ABM-LTFs under reasonable assumptions, first posed by Hofheinz [29].

All-but-Many Trapdoor Function. Without regard to lossines, a notion similar to ABM-LTF is that of all-but-many trapdoor function (ABM-TF). An ABM-TF's inversion trapdoor can be concealed among super-polynomially many tags. Candidate constructions from assumptions related to factoring or discrete logarithm have already been proposed [29,23]. On the other hand, while there exist many constructions and applications of lattice-based all-but-one trapdoor functions [1,35,4] and all-but-N trapdoor functions for N bounded a priori [5], lattice-based ABM-TFs appears to be harder to construct. Therefore, a second motivation for this work is to solve the open problem stated in [5], namely to construct lattice-based ABM-TFs (and, a fortiori, ABM-LTFs).

IND-SO-CCA2 Public-Key Encryption. A direct application of ABM-LTFs, shown in [29], is to construct compact public-key encryption schemes that have ciphertext indistinguishability against adaptively chosen-ciphertext attacks and selective opening attacks (IND-SO-CCA2) [1].

In selective opening attacks (SOA), an adversary gets a collection of some arbitrary N challenge ciphertexts $(\mathsf{ct}_i = \mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}_i; r_i))_{i\in[N]}$ that encrypt m_i with randomness r_i under public key pk, where $\{\mathsf{m}_i\}_{i\in[N]}$ satisfy some joint distribution dist chosen by the adversary. The adversary may choose some subset $\mathcal{I} \subset [N]$ and ask that the corresponding ciphertexts ct_i be "opened" to get (m_i, r_i). The adversary must try to extract information on the messages in the unopened ciphertexts $(\mathsf{ct}_i)_{i\in[N]\setminus\mathcal{I}}$. IND-SO-CCA2 security ensures that no adversary can distinguish the unopened messages from new messages which are freshly and *efficiently* sampled according to dist conditioned on the opened messages. One drawback of this definition of IND-SO-CCA2 is that it requires that the joint message distributions be efficiently re-sampleable conditionally on opened messages. Unfortunately, it is not difficult to come up with examples of efficiently sampleable joint distributions whose conditionals as above would *not* be efficiently sampleable.

A stronger version of indistinguishability-based security definition (sometimes called Full IND-SO-CCA2, see Definition 2 of [31]) does not have the requirement of efficient conditional resampling. This *appears* preferable, but problems remain. First, such stronger definition neither has any known instantiation nor is implied by any known realisable definition, suggesting that it could be too strong to achieve. Second, the existence of efficiently sampleable joint distributions with inefficient conditionals could be exploited by an adversary to use the challenger as a hard-problem oracle, rather than the other way around. Nevertheless, it has been shown by Hofheinz and Rupp[31] that even the first version of IND-SO-CCA2 is stronger than traditional IND-CCA2 security. Therefore it is well motivated to find efficient constructions that are IND-SO-CCA2 secure.

[1] "Compact" here means that the size of public keys is independent of the number of challenge ciphertexts adversary asks for. ABN-LTFs results in IND-SO-CCA2 PKE schemes but the size of public keys is at least linear in N.

For completeness, we mention that stronger and/or more natural definitions than IND-SO-CCA2 are possible, especially in a simulation-based real/ideal framework. We mention the SIM-SO-CCA2 definition (see [8,11] for details) and several PKE schemes that meet it (see, e.g., [22,27,29,32,23]). Nevertheless, SIM-SO-CCA2 secure PKE schemes from lattice assumptions remain unknown.

1.1 Our Contribution

In this paper, we address Hofheinz's [29] open problem of building tightly secure ABM-LTFs under reasonable assumptions. We propose a new ABM-LTF from widely accepted lattice assumptions: specifically, all the security properties of our ABM-LTF can be tightly and ultimately reduced to the computational hardness of Learning with Errors (LWE). Our ABM-LTF also provides a solution to Alperin-Sheriff and Peikert's [5] open problem of constructing ABM-TFs from lattices.

Moreover, by following the pathway given in [27,29,42], our ABM-LTF further leads to the first IND-SO-CCA2 public-key cryptosystem from lattices with a tight security reduction. In turn, such a scheme provides an alternative solution to the question of building tightly secure PKE (without bilinear maps) in the multi-challenge setting, recently and very differently answered by Gay et al. [24]. Being high-dimensional-lattice-based, all of our constructions are conjectured to be quantum-safe.

Our Approach. At a high level, instead of building ABM-LTFs as "encrypted signatures" which is the approach of [29], our ABM-LTF builds an "encrypted homomorphic-evaluation-friendly pseudorandom function" whose outputs are (encrypted) matrices whose rank controls the function's lossiness.

Our starting point is the lattice-based (and lossy) trapdoor function from [10], given by $g(\mathbf{s}, \mathbf{e}) = \mathbf{s}^t \cdot [\mathbf{A}|\mathbf{AR} + \mathbf{HG}] + \mathbf{e}^t \bmod q$, where the matrix $\mathbf{R}$ has low-norm, and $\mathbf{G}$ is the now famous "gadget" matrix (a public matrix with a public trapdoor $\mathbf{T_G}$ such that $\mathbf{G} \cdot \mathbf{T_G} = \mathbf{0}$ with very low norm).

The trapdoor function $g()$ traces back to the two-sided lattice trapdoor framework from [1,13] and the efficient strong lattice trapdoor generators from [35]. It was showed by Bellare et al. [10] that if $\mathbf{A}$ is built from LWE samples (to consist of a truly random matrix on its top and a pseudorandom matrix on its bottom), then for certain parameters, the function is injective and invertible if $\mathbf{H}$ has full column-rank, and is lossy if $\mathbf{H} = \mathbf{0}$. The indistinguishability property of all-but-many trapdoors requires that there must be unbounded many tags that can be mapped to $\mathbf{H} = \mathbf{0}$ and this mapping should be oblivious to "outside" evaluators. Boyen and Li [14] recently showed such a way in another context by embedding a pseudorandom function (PRF) into the above trapdoor function to compute $\mathbf{H}$, i.e., $\mathbf{H} = \mathsf{PRF}(K, \mathsf{tag}) \cdot \mathbf{I}$, where $\mathsf{PRF}(K, \mathsf{tag}) \in \{0, 1\}$ and $\mathbf{I}$ is the identity matrix (in this case $\mathbf{H}$ is square). However, their method only allows two values for $\mathbf{H}$. [2] This makes a random tag lossy with probability half, for

[2] The binary restriction on $\mathbf{H}$ in [14] comes from the fact that the fully homomorphic evaluation techniques from [26,12,16] usually supports operations on two bits or two

hitting $\mathbf{H} = \mathbf{0}$, thereby violating the evasiveness property (i.e., lossy tags should be hard to find without trapdoor).

Our first idea is to parallelly apply multiple PRFs and expand their pseudorandom outputs from bit strings to matrices, through universal hash functions. Particularly, we set tags with form $\mathsf{tag} = (\mathbf{D}, \mu)$. $\mathbf{D}$ is a matrix, which allows us to add additional control on generating $\mathbf{H}$. μ is the input for PRFs. Then we set $\mathbf{H} = \mathbf{ZD} + \sum \mathsf{PRF}(K_i, \mu) \cdot \mathbf{H}_i \bmod q$ for randomly sampled, encrypted matrices $\mathbf{H}_i$, and full-rank matrix $\mathbf{Z}$. Firstly, the subset-sum operation $\sum \mathsf{PRF}(K_i, \mu) \cdot \mathbf{H}_i$ and $\mathbf{ZD}$ can be easily performed by existing evaluation techniques with small adjustments on the dimensions of gadget matrices as we will show. Secondly, for any "outside" evaluator, as the outputs of PRFs are unpredictable, the output of the subset-sum formula, and, hence, $\mathbf{H}$ will be pseudorandom. For one who knows the keys of PRFs and the matrix $\mathbf{Z}$, a lossy tag can be generated by randomly selecting μ and solving $\mathbf{D}$ for the equation $\mathbf{0} = \mathbf{ZD} + \sum \mathsf{PRF}(K_i, \mu) \cdot \mathbf{H}_i \pmod q$. Now the problem we have is that the adversary can reuse μ from a prior lossy tag $(\mathbf{D}, \mu)$ it was given, to create a new tag $(\bar{\mathbf{D}}, \mu)$ where $\bar{\mathbf{D}} = \mathbf{D} + \mathbf{D'}$ for non-full-rank $\mathbf{D'}$. This special tag — we call it an "invalid tag" — could disable the gadget trapdoors while still making the function injective. To solve this problem, we use a chameleon hash function to tie $\mathbf{D}$ and μ together (say μ is the output of the chameleon hash on input $\mathbf{D}$ and some fresh randomness) to enforce the one-time use of μ. For generating lossy tags in the simulation, we can pick random μ, solve for $\mathbf{D}$ and use the trapdoor of the chameleon hash function to find randomness under which μ chameleon-hashes to $\mathbf{D}$.

As a consequence of using a chameleon hash function, the inputs to the PRFs (i.e., μ) will be random for all randomly generated tags in the real schemes and all responses from the lossy tag generation oracle to queries in the security reductions. Moreover, the collision-resistant property of the chameleon hash function essentially forces all adversarially generated PRF inputs (i.e., μ) to be different. This fact drives us towards relaxing the PRFs into so-called "weak PRFs" [3], which only guarantee pseudorandomness for *random* inputs. The advantages of using weak PRFs is that weak PRFs admit potentially much simpler, more efficient constructions from weaker assumptions, with shallower circuit implementations than normal PRFs. The remaining problem of using a weak PRF (WPRF for short) instead of a usual PRF is that, in the evasiveness security game, the adversary is allowed adaptively to come up with lossy tag guesses in which μ may not be random, and receive binary answers of "lossy/invalid" or "injective". Such answers may leak damaging information to the adversary, since the WPRF indistinguishability from random may not apply on non-random inputs μ.

We resolve this last problem by pre-processing μ with a (very basic) universal hash, essentially XOR-ing μ with a secret constant. This keeps the WPRF input

small scalars. It would be very useful to find a way to do such evaluation over two vectors or matrices. We note that Hiromasa et al. [28] showed how to do homomorphic evaluation on matrices for GSW-FHE scheme [26]. But it is not clear how to apply such technique to the gadget-based trapdoors.

random for all the challenger-generated μ_i, and further randomises one of the adversarially generated μ to make it jointly random with the random μ_i. This restores WPRF indistinguishability for one adversarial queries, which in turns all but guarantees (with probability overwhelmingly close to 1) that the response to the adversary's guess will be "injective". Because the response was a foregone conclusion, it is devoid of information, and could have been answered without looking at μ. This allows us to consider the *second* adversarial query without regard for the first one (which we answered without even looking at it). Repeating the previous argument, this *second* adversarial query μ together with the random μ_i induce a set of jointly random WPRF inputs after universal hashing, and thus the adversary will also expect an "injective" answer with all but negligible probability on this second query, as for the first query. The conclusion carries inductively for any polynomially bounded number of queries.

For our purpose of constructing ABM-LTF and PKE schemes without relying on any of the "pre-quantum" assumptions of existing schemes, WPRFs can be instantiated directly from the Learning-With-Rounding assumption [7]. Such WPRFs can be implemented as Boolean NAND circuits in the $\mathbf{NC}^1$ circuit class, which allows us to use smaller modulus in our construction (or nearly equivalently, larger relative LWE noise). The addition of a universal hash (or a simple XOR) at the input of the WPRF barely makes the circuit more complex.

Finally, we also mention that we need that random tags (and even adversarially chosen tags) make the column-rank of $\mathbf{H}$ full, with overwhelming probability, as required for evasiveness of ABM-LTFs. Since we are able to use a polynomial rather than sub-exponential modulus (in the security parameter), a randomly sampled square matrix $\mathbf{H}$ will not overwhelmingly likely be full-rank. We resolve this by adding extra columns to $\mathbf{H}$, making it "wider", and to such end we also adjust the dimension of the gadget matrices. (We note that if the WPRF, which we can view as a black-box, is instantiated from LWR problem, it would use another modulus which unfortunately is slightly super-polynomial [7].)

A Parallel and Independent Work. In concurrent and independent work, Libert et al. [34] propose an ABM-LTF and a SIM-SO-CCA2 secure PKE scheme using rather similar techniques. Both papers give ABM-LTF constructions based on embedding key-homomorphic PRF evaluation into the lattice-based LTF of Bellare et al. [9], and give applications to PKE with selective-opening security.

The first notabale difference is that our ABM-LTF uses the weaker notion of weak PRF in the homomorphic evaluation. Unlike the stronger usual PRFs, weak PRFs need not to be pseudo-random on all inputs; only on random ones. They have more efficient constructions from weaker assumptions, along with tighter reductions. Using weak PRFs gives us shallower circuit implementations, which cause milder noise growth in the key-homomorphic evaluations. In turn, this lessens our LWE assumptions for the construction of ABM-LTF.

The second important difference is that the PKE scheme in [34] does achieve SIM-SO-CCA2 security, compared to ours which has IND-SO-CCA2 security. It is the first lattice-based PKE scheme that enjoys such strong notion of selective-opening security. At a high level, they first build an IND-SO-CCA2-secure PKE

scheme from their ABM-LTF, then give an efficient mechanism to "explain" any lossy ciphertext as an encryption of an arbitrary message to get SIM-SO-CCA2.

A natural question, given the complementary strengths of our respective papers, would be to combine them and achieve the best of both worlds.

1.2 Other Related Works

Lossy trapdoor functions (LTFs) were proposed by Peikert and Waters [39]. They admit instantiations from standard assumptions, e.g., DDH, DCR and LWE. They also have enormous applications, e.g., in the construction of IND-CCA2 public-key schemes, the first lattice trapdoor function, lossy encryption [8]. All-but-N LTFs (ABN-LTFs) were firstly proposed by Hemenway et al. [27] as a means to construct PKE secure against chosen-ciphertext and selective opening attacks. In contrast to ABM-LTFs in which unbounded many lossy tags are provided, an ABN-LTF contains exact N lossy tags. ABN-LTFs suffer from a drawback that N has to be fixed when generating the public parameters, making the size of public parameters grow at least linearly in N. Last, we mention that lossiness arguments have been used in a LWE context for establishing the hardness of the LWE problem with uniform rather than Gaussian noise [21,36].

2 Preliminaries

Notation. 'PPT' abbreviates "probabilistic polynomial-time". If S is a set, we denote by $a \xleftarrow{\$} S$ the uniform sampling of a random element of S. For a positive integer n, we denote by $[n]$ the set of positive integers no greater than n. We use bold lowercase letters (e.g. $\mathbf{a}$) to denote vectors and bold capital letters (e.g. $\mathbf{A}$) to denote matrices. For a positive integer $q \geq 2$, let $\mathbb{Z}_q$ be the ring of integers modulo q. We denote the group of $n \times m$ matrices in $\mathbb{Z}_q$ by $\mathbb{Z}_q^{n\times m}$. Vectors are treated as column vectors. The transpose of a vector $\mathbf{a}$ (resp. a matrix $\mathbf{A}$) is denoted by $\mathbf{a}^t$ (resp. $\mathbf{A}^t$). For $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{B} \in \mathbb{Z}_q^{n\times m'}$, let $[\mathbf{A}|\mathbf{B}] \in \mathbb{Z}_q^{n\times(m+m')}$ be the concatenation of $\mathbf{A}$ and $\mathbf{B}$. We write $\|\mathbf{x}\|$ for the Euclidean norm of a vector $\mathbf{x}$. The Euclidean norm of a matrix $\mathbf{R} = \{\mathbf{r}_1, \ldots, \mathbf{r}_m\}$ is denoted by $\|\mathbf{R}\| = \max_i \|\mathbf{r}_i\|$. The spectral norm of $\mathbf{R}$ is denoted by $s_1(\mathbf{R}) = \sup_{\mathbf{x}\in\mathbb{R}^{m+1}} \|\mathbf{R}\cdot\mathbf{x}\|$. The inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ is written $\langle \mathbf{x}, \mathbf{y}\rangle$. For a security parameter λ, a function $\mathsf{negl}(\lambda)$ is negligible in λ if it is smaller than all polynomial fractions for a sufficiently large λ. The logarithm function $\log_2(\cdot)$ is abbreviated as $\log(\cdot)$.

We will be using the following lemma which is directly implied by the Theorem 1.1 of [17]

Lemma 1. *Let an integer $n \geq 2$, and a prime $q \geq 2$. A randomly sampled $\mathbb{Z}_q^{n\times 2n}$-matrx $\mathbf{H}$ will have n linearly independent columns, i.e., rank n, with all but negligible probability in n.*

Proof. By the Theorem 1.1 of [17] the probability that $\mathbf{H}$ has rank n is

$$\prod_{i=1}^{n}(1-\frac{1}{q^{n+i}}) \geq (1-\frac{1}{q^{n+1}})^n \geq 1 - n\cdot q^{-(n+1)} \geq 1 - \mathsf{negl}(n)$$

as required. □

2.1 Randomness Extractor

Let X and Y be two random variables over some finite set S. The statistical distance between X and Y, denoted as $\Delta(X,Y)$, is defined as

$$\Delta(X,Y) = \frac{1}{2}\sum_{s\in S} |\Pr[X=s] - \Pr[Y=s]|\,.$$

Let X_λ and Y_λ be ensembles of random variables indexed by the security parameter λ. X and Y are statistically close if $\Delta(X_\lambda, Y_\lambda) = \mathsf{negl}(\lambda)$.

The min-entropy of a random variable X over a set S is defined as

$$H_\infty(X) = -\log(\max_{s\in S} \Pr[X=s]).$$

The average min-entropy of a random variable X given Y is defined as

$$\tilde{H}_\infty(X|Y) = -\log\left(\mathbb{E}_{y\leftarrow Y}\left[2^{-H_\infty(X|Y=y)}\right]\right)$$

Lemma 2 ([38], Lemma 2.1). *If Y takes at most 2^r possible values and X is any random variable, then*

$$\tilde{H}_\infty(X|Y) \geq H_\infty(X) - r.$$

Definition 1 (Universal Hash Functions). *A family of functions $\mathcal{UH} = \{\mathsf{UH}_\mathbf{k} : \mathcal{X} \to \mathcal{Y}\}$ is called a family of universal hash functions with index (key) $\mathbf{k}$, if for all $x, x' \in \mathcal{X}$, with $x \neq x'$, we have $\Pr[\mathsf{UH}_\mathbf{k}(x) = \mathsf{UH}_\mathbf{k}(x')] \leq \frac{1}{|\mathcal{X}|}$ over the random choice of $\mathsf{UH}_\mathbf{k}$.*

Lemma 3 ([38], Lemma 2.2). *Let X, Y be random variables such that $X \in \{0,1\}^n$ and $\tilde{H}_\infty(X|Y) \geq k$. Let $\mathcal{UH}$ be a family of universal hash functions from $\{0,1\}^n$ to $\{0,1\}^\ell$ where $\ell \leq k - 2\log(1/\epsilon)$. It holds that for $\mathsf{UH}_\mathbf{k} \xleftarrow{\$} \mathcal{UH}$ and $r \xleftarrow{\$} \{0,1\}^\ell$, $\Delta\left((\mathsf{UH}_\mathbf{k}, \mathsf{UH}_\mathbf{k}(X), Y), (\mathsf{UH}_\mathbf{k}, r, Y)\right) \leq \epsilon$.*

Corollary 1. *Let $q > 2$, $\epsilon > 0$. Let $\mathcal{UH} = \{\mathsf{UH}_\mathbf{h} : \{0,1\}^\ell \to \mathbb{Z}_q\}$ be a family of hash functions where $\ell \geq \log(q/(\epsilon^2))$, $y = \mathsf{UH}_\mathbf{h}(x) = \sum_{i=1}^{\ell} h_i x_i \bmod q$ for $x = x_1 \ldots x_\ell \in \{0,1\}^\ell$, $\mathbf{h} = h_1 \ldots h_\ell \xleftarrow{\$} \mathbb{Z}_q^\ell$. Let $r \xleftarrow{\$} \mathbb{Z}_q$, we have $\Delta((\mathsf{UH}_\mathbf{h}, \mathsf{UH}_\mathbf{h}(x)), (\mathsf{UH}_\mathbf{h}, r)) \leq \epsilon$.*

Proof. It is easy to see that for different inputs x and x', and $\mathbf{h} \xleftarrow{\$} \mathbb{Z}_q^\ell$, $\mathsf{UH}_\mathbf{h}(x) = \mathsf{UH}_\mathbf{h}(x')$ happens with probability $1/q$. So $\mathcal{UH}$ is a family of universal hash function. Applying Lemma 3 concludes the proof. □

2.2 Discrete Gaussians

Let $m \in \mathbb{Z}_{>0}$ be a positive integer. Let an integer lattice $\Lambda \subset \mathbb{Z}^m$. For any real vector $\mathbf{c} \in \mathbb{R}^m$ and positive parameter $\sigma \in \mathbb{R}_{>0}$, let the Gaussian function $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp\left(-\pi\|\mathbf{x}-\mathbf{c}\|^2/\sigma^2\right)$ on $\mathbb{R}^m$ with centre $\mathbf{c}$ and parameter σ. Define the discrete Gaussian distribution over Λ with centre $\mathbf{c}$ and parameter σ as $D_{\Lambda,\sigma} = \rho_{\sigma,\mathbf{c}}(\mathbf{y})/\rho_\sigma(\Lambda)$ for $\forall \mathbf{y} \in \Lambda$, where $\rho_\sigma(\Lambda) = \sum_{\mathbf{x}\in\Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$. For notational convenience, $\rho_{\sigma,\mathbf{0}}$ and $D_{\Lambda,\sigma,\mathbf{0}}$ are abbreviated as ρ_σ and $D_{\Lambda,\sigma}$.

Lemma 4 ([9], Lemma 5.1). *Let $h > 0$, $w > 0$ be integers and $\sigma > 0$ be Gaussian parameter. For $\mathbf{R} \leftarrow D_{\mathbb{Z},\sigma}^{h\times w}$, we have $s_1(\mathbf{R}) \le \sigma \cdot O(\sqrt{h}+\sqrt{w})$ with all but probability $2^{-\Omega(h+w)}$.*

Lemma 5 ([9], Lemma 5.2). *For prime q and integer $b \ge 2$, let $\bar{m} \ge n\log_b q + \omega(\log n)$. With overwhelming probability over the uniformly random choice of $\mathbf{A} \in \mathbb{Z}_q^{n\times\bar{m}}$, the following holds: for $\mathbf{r} \leftarrow D_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}^{\bar{m}}$, the distribution of $\mathbf{Ar}$ is statistically close to the uniform distribution over $\mathbb{Z}_q^n$.*

2.3 Gadget Matrices

We define two gadget matrices with different dimensions than the canonical gadget matrix given by Micciancio and Peikert [35]. Let an integer $n \ge 2$, a primt $q \ge 2$, a radix $b \ge 2$, and let $w = \log_b q$. Let $\mathbf{G}^*$ be the primitive matrix defined as $\mathbf{G}^* = \mathbf{I}_n \otimes [1, b, b^2, \ldots, b^{w-1}] \in \mathbb{Z}_q^{n\times nw}$. We define the gadget matrices

$$\mathbf{G} = \left[\mathbf{G}^* | \ \mathbf{0}\right] \in \mathbb{Z}_q^{n\times 2nw}$$

and

$$\hat{\mathbf{G}} = \mathbf{I}_{2n} \otimes [1, b, b^2, \ldots, b^{w-1}] = \begin{bmatrix} \mathbf{G} & \mathbf{0} \\ \mathbf{0} & \mathbf{G} \end{bmatrix} \in \mathbb{Z}_q^{2n\times 2nw}.$$

Those gadget matrices have useful properties as stated below.

Lemma 6 ([12], Lemma 2.1). *There is a deterministic algorithm, denoted $\mathbf{G}^{-1}(\cdot) : \mathbb{Z}_q^{n\times m} \to \mathbb{Z}^{m\times m}$, that takes any matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ as input, and outputs the preimage $\mathbf{G}^{-1}(\mathbf{A})$ of $\mathbf{A}$ such that $\mathbf{G}\cdot\mathbf{G}^{-1}(\mathbf{A}) = \mathbf{A} \pmod q$ and $s_1\left(\mathbf{G}^{-1}(\mathbf{A})\right) \le (b-1)m$.*

There is a deterministic algorithm, denoted $\hat{\mathbf{G}}^{-1}(\cdot) : \mathbb{Z}_q^{2n\times m} \to \mathbb{Z}^{m\times m}$, that takes any matrix $\mathbf{A} \in \mathbb{Z}_q^{2n\times m}$ as input, and outputs the preimage $\hat{\mathbf{G}}^{-1}(\mathbf{A})$ of $\mathbf{A}$ such that $\hat{\mathbf{G}}\cdot\hat{\mathbf{G}}^{-1}(\mathbf{A}) = \mathbf{A} \pmod q$ and $s_1(\hat{\mathbf{G}}^{-1}(\mathbf{A})) \le (b-1)m$.

Lemma 7 ([35], Theorem 3). *Let $\mathbf{A} \in \mathbb{Z}_q^{n\times\bar{m}}$, $\mathbf{R} \in \mathbb{Z}^{\bar{m}\times 2nw}$. Let $\mathbf{H} \in \mathbb{Z}_q^{n\times 2nw}$ with rank n. Let $\hat{\mathbf{G}} \in \mathbb{Z}_q^{2n\times 2nw}$ be the gadget matrix. For $\mathbf{y}^t = g_{\mathbf{F}}(\mathbf{x}) = \mathbf{x}^t \begin{bmatrix} \mathbf{I}_m \\ \mathbf{F} \end{bmatrix} = \mathbf{x}_1^t + \mathbf{x}_2^t \cdot \mathbf{F} \bmod q$ where $\mathbf{F} = [\mathbf{A}|\mathbf{AR} + \mathbf{H}\hat{\mathbf{G}}]$, there is a PPT algorithm $\mathsf{Invert}(\mathbf{F}, \mathbf{R}, \mathbf{H}, \mathbf{y})$ that outputs $\mathbf{x}$ with overwhelming probability if $\|\mathbf{x}_1\| \le q/\Theta(b\cdot s_1(\mathbf{R}))$.*

2.4 Homomorphic Evaluation Algorithms

In our construction we use the homomorphic evaluation algorithms developed in [26,16,12]. The next lemma follows directly from Claim 3.4.2, the Lemma 3.6, and Theorem 3.5 of [16]. It has been used in [14].

Lemma 8. *Let $C : \{0,1\}^\ell \to \{0,1\}$ be a NAND Boolean circuit in class $\boldsymbol{NC}^1$, i.e. C has depth $d = \eta \log \ell$ for some constant η. Let $\{\mathbf{A}_i = \mathbf{A}\mathbf{R}_i + x_i\mathbf{G} \in \mathbb{Z}_q^{n\times 2nw}\}_{i\in[\ell]}$ be ℓ matrices correspond to the ℓ input wires of C where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times\bar{m}}$, $\mathbf{R}_i \leftarrow D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$, $x_i \in \{0,1\}$ and $\mathbf{G} \in \mathbb{Z}_q^{n\times 2nw}$ is the gadget matrix. There is an efficient deterministic algorithm $\mathsf{Eval}_{\mathsf{BV}}$ that takes as input C and $\{\mathbf{A}_i\}_{i\in[\ell]}$ and outputs a matrix $\mathbf{A}_C = \mathbf{A}\mathbf{R}_C + C(x_1,\ldots,x_\ell)\mathbf{G} = \mathsf{Eval}_{\mathsf{BV}}(C,\mathbf{A}_1,\ldots,\mathbf{A}_\ell)$ where $\mathbf{R}_C \in \mathbb{Z}^{\bar{m}\times 2nw}$ can be computed deterministically from $\{\mathbf{R}_i\}_{i\in[\ell]}$ and $\{x_1\}_{i\in[\ell]}$, and $C(x_1,\ldots,x_\ell)$ is the output of C on the arguments $x_1,\ldots,x_\ell$. $\mathsf{Eval}_{\mathsf{BV}}$ runs in time poly$(4^d,\ell,n,\log q)$.*

Let $2nw \le \bar{m}$. So $s_1(\mathbf{R}_{max}) = max\{s_1(\mathbf{R}_i)\}_{i\in[\ell]} \le b\cdot O(\sqrt{\bar{m}})$ by Lemma 4. the spectral norm of $\mathbf{R}_C$ can be bounded, with overwhelming probability, by $s_1(\mathbf{R}_C) \le O(4^d\cdot\bar{m}^{3/2}) = O(\ell^{2\eta}\cdot\bar{m}^{3/2})$.

We also explicitly use the following two evaluation formulas. Let $\mathbf{C} = \mathbf{A}\mathbf{R} + x\mathbf{G}$ and $\hat{\mathbf{C}} = \mathbf{A}\hat{\mathbf{R}} + \mathbf{H}\hat{\mathbf{G}}$ where $\mathbf{A} \in \mathbb{Z}_q^{n\times\bar{m}}$, $\mathbf{R},\hat{\mathbf{R}} \in \mathbb{Z}^{\bar{m}\times 2nw}$ has low norm, $x \in \{0,1\}$, $\mathbf{H} \in \mathbb{Z}^{n\times 2n}$, and $\mathbf{G} \in \mathbb{Z}_q^{n\times 2nw}$, $\hat{\mathbf{G}} \in \mathbb{Z}_q^{2n\times 2nw}$ be gadget matrices. We can multiplicatively evaluate $\mathbf{C}$ and $\hat{\mathbf{C}}$ with respect to the "message" product $x\mathbf{H}$ by computing

$$\begin{aligned}\hat{\mathbf{C}}' &= \mathbf{C}\cdot\mathbf{G}^{-1}(\hat{\mathbf{C}}) \pmod q\\ &= \mathbf{A}\mathbf{R}\cdot\mathbf{G}^{-1}(\hat{\mathbf{C}}) + x(\mathbf{A}\hat{\mathbf{R}}) + x\mathbf{H}\hat{\mathbf{G}} \pmod q\\ &= \mathbf{A}\hat{\mathbf{R}}' + x\mathbf{H}\hat{\mathbf{G}} \pmod q\end{aligned}$$

Let $\hat{\mathbf{C}}_1 = \mathbf{A}\hat{\mathbf{R}}_1 + \mathbf{Z}\hat{\mathbf{G}}$ and $\hat{\mathbf{C}}_2 = \mathbf{M}\hat{\mathbf{G}}$ [3] where $\mathbf{A} \in \mathbb{Z}_q^{n\times\bar{m}}$, $\hat{\mathbf{R}}_1,\hat{\mathbf{R}}_2 \in \mathbb{Z}^{\bar{m}\times 2nw}$ have low norm, $\mathbf{Z} \in \mathbb{Z}_q^{n\times 2n}$, $\mathbf{M} \in \mathbb{Z}_q^{2n\times 2n}$, and $\hat{\mathbf{G}}$ is the gadget matrix. We compute the "encryption" of $\mathbf{Z}\mathbf{M} \in \mathbb{Z}_q^{n\times 2n}$ by computing:

$$\begin{aligned}\hat{\mathbf{C}} &= \hat{\mathbf{C}}_1\cdot\hat{\mathbf{G}}^{-1}(\hat{\mathbf{C}}_2) \pmod q\\ &= \mathbf{A}\left(\hat{\mathbf{R}}_1\cdot\hat{\mathbf{G}}^{-1}(\hat{\mathbf{C}}_2)\right) + (\mathbf{Z}\mathbf{M})\hat{\mathbf{G}} \pmod q\\ &= \mathbf{A}\hat{\mathbf{R}} + (\mathbf{Z}\mathbf{M})\hat{\mathbf{G}} \pmod q\end{aligned}$$

2.5 Computational Assumptions

We use the classic variant of learning-with-errors (LWE) problem where the secret components have the same distribution as the noise components. Such variant is known as the normal-form LWE problem and is no easier than the

[3] In our construction, $\mathbf{Z}$ will be hidden and $\mathbf{M}$ will be publicly samplable.

LWE problem with uniform secret, up to a small difference in the number of available samples (see e.g., [6]). Additionally, we consider the LWE problem in which the secret is a matrix in $\mathbb{Z}^{n\times h}$ rather than single vector in $\mathbb{Z}^n$. By a standard hybrid argument, such problem, as shown in Lemma 6.2 of [39], can be reduced to the LWE problem with a single vector secret, while loosing a factor h of security. We point out that in our constructions h is independent of the number of adversarial queries.

Definition 2. *Let n, q, h be positive integers. Let χ be be a distribution over $\mathbb{Z}_q$. Let $\mathbf{S} \leftarrow \chi^{n\times h}$ be a secret matrix. Define two oracles:*

- $\mathcal{O}_{\mathbf{S}}$*: samples* $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^h$*; returns* $(\mathbf{a}, \mathbf{S}^t\mathbf{a} + \mathbf{e}^t \bmod q)$.
- $\mathcal{O}_{\$}$*: samples* $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{b} \xleftarrow{\$} \mathbb{Z}_q^h$*; returns* $(\mathbf{a}, \mathbf{b})$.

The normal form of the $\mathsf{LWE}_{n,h,q,\chi}$ problem with matrix secret asks for distinguishing between $\mathcal{O}_{\mathbf{S}}$ and $\mathcal{O}_{\$}$. The advantage of a distinguishing algorithm $\mathcal{A}$ in the security parameter λ is defined as

$$\mathsf{Adv}^{\mathsf{LWE}_{n,h,q,\chi}}_{\mathsf{NF},\mathcal{A}}(\lambda) = \left|\Pr[\mathcal{A}^{\mathcal{O}_{\mathbf{S}}}(1^\lambda) = 1] - \Pr[\mathcal{A}^{\mathcal{O}_{\$}}(1^\lambda) = 1]\right|$$

We also implicity make the short integer solution (SIS) assumption [2,25] for invoking the lattice-based chameleon hash function by Cash et al. [18], which is viewed as a black box in our constructions. Since the SIS assumption is quantitatively much weaker than the LWE assumption we use, and is implied by it, our constructions are ultimately based on LWE assumption.

3 Definitions

3.1 Weak Pseudorandom Functions

Weak pseudorandom functions (weak PRFs) [3] are keyed functions that have pseudorandom outputs on *random* inputs. They hav many applications in protocol design, e.g., [20,33], improving efficiency when a full PRF is not needed.

Let λ be a security parameter, $t = t(\lambda)$, and $\ell = \ell(\lambda)$. An efficiently computable, deterministic (one-bit-output) function family $F : \{0,1\}^t \times \{0,1\}^\ell \to \{0,1\}$ is called weak PRF if it satisfies the following: For every $Q = \mathsf{poly}(\lambda)$, the ensemble $\mathcal{X} = \{(x_i, F_K(x_i)\}_{i\in[Q]}$ is computationally indistinguishable from the ensemble $\mathcal{Y} = \{(x_i, R(x_i))\}_{i\in[Q]}$, where K is random in $\{0,1\}^t$, x_i is random in $\{0,1\}^\ell$, and $R : \{0,1\}^\ell \to \{0,1\}$ is a random function.

Weak PRFs, which turn out to be much weaker that normal PRFs, admit simple and efficient constructions from various assumptions. To base our ABM-LTF purely on lattice assumptions, we can use a weak PRF from [7]

$$F_K(\cdot) = \lfloor \frac{p}{q} \langle \mathbf{s}, \cdot \rangle \rceil \bmod p \qquad \text{where } 2 \le p \ll q$$

For binary output, $p = 2$. The key $K = \mathbf{s}$ is a randomly chosen vector in $\mathbb{Z}_q^n$. $F_K(\cdot)$ has input space $\mathbb{Z}_q^n$. The security of $F_K(\cdot)$ is based on the hardness of learning with rounding (LWR), a deterministic variation on LWE, defined in [7].

Let $q > p \geq 2$. For a vector $\mathbf{s} \in \mathbb{Z}_q^n$, the LWR distribution $L_{\mathbf{s}}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_p$ is obtained by randomly choosing $\mathbf{a}$ form $\mathbb{Z}_q^n$, and outputting $(\mathbf{a}, \lfloor \frac{p}{q} \langle \mathbf{s}, \mathbf{a} \rangle \rceil \bmod p)$. The $\mathsf{LWR}_{n,p,q}$ problem asks for distinguishing between any desired number of independent samples from $L_{\mathbf{s}}$, and the same number of samples from uniform distribution over $\mathbb{Z}_q^n \times \mathbb{Z}_p$. It has been shown that the hardness of the decision LWR problem can be based on the decision LWE problem for certain parameters.

Notice that $F_K(\cdot)$ with $p = 2$ is exactly an instance of the decision-$\mathsf{LWR}_{n,2,q}$ problem, and it is a weak pseudorandom function if the $\mathsf{LWR}_{n,2,q}$ problem is hard. It has been shown that for $q/2 \geq (\alpha q) \cdot n^{\omega(1)}$, the $\mathsf{LWR}_{n,2,q}$ problem is no easier than the $\mathsf{LWE}_{n,q,D_{\mathbb{Z}},\alpha q}$ problem where $\alpha \leq n^{-\omega(1)}$. [4] We note that F_K here is essentially the same decryption circuit as in many lattice-based encryption schemes (e.g., [15,26,16]) and belongs to a very shallow $\mathbf{NC}^1$ circuit class.

3.2 Chameleon Hash Functions

A chameleon hash function $\mathsf{CH} = (\mathsf{CH.Gen}, \mathsf{CH.Eval}, \mathsf{CH.Equiv})$ has three PPT algorithms. The key generation algorithm $\mathsf{CH.Gen}$ takes as input a security parameter λ, outputs a hash key and trapdoor pair $(\mathsf{Hk}, \mathsf{Td})$. The randomised hashing algorithm takes as input a message X, random coins $r \in \mathcal{R}_{\mathsf{CH}}$, and outputs $Y = \mathsf{CH.Eval}(\mathsf{Hk}, X; R)$. The equivocation algorithm takes as input a trapdoor Td, an arbitrary valid hash value y and an arbitrary message x, and outputs a valid randomness $R \in \mathcal{R}_{\mathsf{CH}}$ such that $Y = \mathsf{CH.Eval}(\mathsf{Hk}, X; R)$.

A chameleon hash function has output uniformity which guarantees the distribution of hashes is independent of the messages. Particularly, for all Hk, two messages X, X', the distributions $\{R \xleftarrow{\$} \mathcal{R}_{\mathsf{CH}} \ : \ \mathsf{CH.Eval}(\mathsf{Hk}, X; R)\}$ and $\{R \xleftarrow{\$} \mathcal{R}_{\mathsf{CH}} \ : \ \mathsf{CH.Eval}(\mathsf{Hk}, X'; R)\}$ are identical. A chameleon hash function is collision-resistant. That is, for all PPT adversary $\mathcal{A}$, for random $(\mathsf{Hk}, \mathsf{Td}) \leftarrow \mathsf{CH.Gen}(1^\lambda)$, the advantage

$$\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{A}}(\lambda) = \begin{bmatrix} ((X, R), (X', R')) \leftarrow \mathcal{A}(1^\lambda, \mathsf{Hk}) \\ (X, R) \neq (X', R'), \\ \mathsf{CH.Eval}(\mathsf{Hk}, X; R) = \mathsf{CH.Eval}(\mathsf{Hk}, X'; R') \end{bmatrix}$$

must be negligible in λ.

As in the definition of chameleon hash function from [29], the message space is assumed to be $\{0,1\}^*$. This is not a big issue since we can always apply a collision-resistant hash function on the input to get a chameleon-hash input with fixed size. We additionally require the chameleon hash function used in

[4] Unfortunately, this proof indicates that if such LWR-based weak PRFs are used in our construction, we need to make a slightly stronger LWE assumption with super-polynomial modulus q. However, such LWE assumption remains weaker than widely used LWE assumptions with sub-exponential moduli q, e.g., [12].

our ABM-LTF construction to have the following property in order to achieve selective opening security:

Definition 3. *Let* $\mathsf{CH} = (\mathsf{CH.Gen}, \mathsf{CH.Eval}, \mathsf{CH.Equiv})$ *be a secure chameleon hash function. We say* CH *has* ***equivocation indistinguishability*** *if, for random* $(\mathsf{Hk}, \mathsf{Td}) \leftarrow \mathsf{CH.Gen}(1^\lambda)$*, given a fixed message* $X \in \mathcal{X}_{\mathsf{CH}}$*, the following two distributions of tuple* (X, R, Y) *are statistically indistinguishable:*

$$\{X \in \mathcal{X}_{\mathsf{CH}}, R \xleftarrow{\$} \mathcal{R}_{\mathsf{CH}}, Y \leftarrow \mathsf{CH.Eval}(\mathsf{Hk}, X; R)) \in \mathcal{Y}_{\mathsf{CH}}\}$$

and

$$\{X \in \mathcal{X}_{\mathsf{CH}}, Y \xleftarrow{\$} \mathcal{Y}_{\mathsf{CH}}, R \leftarrow \mathsf{CH.Equiv}(\mathsf{Td}, Y, X) \in \mathcal{R}_{\mathsf{CH}}\}$$

Cash et al. [18] constructed a chameleon hash function from the short integer solutions (SIS) assumption [2]. Such construction has equivocation indistinguishability and output uniformity which follow directly from the properties of preimage-sampleable functions given by Gentry et al. [25].

3.3 Lossy Trapdoor Functions

A lossy trapdoor function with domain D consists of three PPT algorithms:

- $\mathsf{LTF.Gen}(1^\lambda, \mathsf{mode})$: a key generation algorithm that takes as input a security parameter and a mode parameter $\mathsf{mode} = \{\mathsf{inj}, \mathsf{loss}\}$, then behaves as follows:
  - $\mathsf{LTF.Gen}(1^\lambda, \mathsf{inj})$ outputs $(\mathsf{LTF.ek}, \mathsf{LTF.ik})$ where $\mathsf{LTF.ek}$ is a injective evaluation key and $\mathsf{LTF.ik}$ is an inversion trapdoor.
  - $\mathsf{LTF.Gen}(1^\lambda, \mathsf{loss})$ outputs $(\mathsf{LTF.ek}, \perp)$ where $\mathsf{LTF.ek}$ is a lossy evaluation key.
- $\mathsf{LTF.Eval}(\mathsf{LTF.ek}, X)$: an evaluation function that evaluates the function on input $X \in \mathsf{D}$ using evaluation key $\mathsf{LTF.ek}$.
- $\mathsf{LTF.Inv}(\mathsf{LTF.ik}, Y)$: an inversion function that takes as input a value Y, and uses the inversion key $\mathsf{LTF.ik}$ to find a value X.

A lossy trapdoor function has the following properties.

Invertibility. For all $(\mathsf{LTF.ek}, \mathsf{LTF.ik}) \leftarrow \mathsf{LTF.Gen}(1^\lambda, \mathsf{inj})$, $X \in \mathsf{D}$, and $Y = \mathsf{LTF.Eval}(\mathsf{LTF.ek}, X)$, we have

$$\Pr\left[X = \mathsf{LTF.Inv}(\mathsf{LTF.ik}, Y)\right] = 1 - \mathsf{negl}(\lambda)$$

Lossiness. We say that the lossy trapdoor function is ℓ-lossy if for all $\mathsf{LTF.ek} = \mathsf{LTF.Gen}(1^\lambda, \mathsf{loss})$, the image set of $\mathsf{LTF.Eval}(\mathsf{LTF.ek}, \mathsf{D})$ has size at most $|\mathsf{D}|/2^\ell$.

Indistinguishability. The first outputs of $\mathsf{LTF.Gen}(1^\lambda, \mathsf{inj})$ and $\mathsf{LTF.Gen}(1^\lambda, \mathsf{loss})$ are computationally indistinguishable. That is, for all PPT adversary $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{A}}(\lambda)$, given by

$$\left|\Pr\left[\mathcal{A}(1^\lambda, \mathsf{LTF.ek}) = 1\right] - \Pr\left[\mathcal{A}(1^\lambda, \mathsf{LTF.ek}') = 1\right]\right|$$

is negligible in λ, where $(\mathsf{LTF.ek}, \mathsf{LTF.ik}) \leftarrow \mathsf{LTF.Gen}(1^\lambda, \mathsf{inj})$ and $(\mathsf{LTF.ek}', \perp) \leftarrow \mathsf{LTF.Gen}(1^\lambda, \mathsf{loss})$.

3.4 All-But-Many Lossy Trapdoor Functions

Our definition mainly follows the original definition given by Hofheinz [29], and maintains the same tagging mechanism. That is, a tag tag is divided into two parts: the primary part $\mathbf{t}_\mathsf{p}$ and the auxiliary part $\mathbf{t}_\mathsf{a}$. The auxiliary part is usually just a random string. For any $\mathbf{t}_\mathsf{a}$, given a lossy tag generation trapdoor, one can compute $\mathbf{t}_\mathsf{p}$ to make $\mathsf{tag} = (\mathbf{t}_\mathsf{p}, \mathbf{t}_\mathsf{a})$ a lossy tag. As in [29], the auxiliary part helps us to embed auxiliary information (e.g., a one-time signature verification key).

One difference between our definition (and construction) and that of Hofheinz, is that we divide a tag set into *three* disjoint subsets: (1) a lossy tag set, (2) an injective tag set and (3) an invalid tag set. This is because in our lattice-based construction, some tags can simultaneously make the function injective and disable the inversion trapdoor. We will need to make sure that those tags are generally hard to find (except when knowing a trapdoor).

We now define ABM-LTFs. An all-but-many lossy trapdoor function with domain D consists of four PPT algorithms:

- $\mathsf{ABM.Gen}(1^\lambda)$: a key generation algorithm. It takes as input a security parameter, and outputs an evaluation key $\mathsf{ABM.ek}$, an inversion key $\mathsf{ABM.ik}$, and a lossy tag generation key $\mathsf{ABM.tk}$. The evaluation key $\mathsf{ABM.ek}$ defines the tag space $\mathcal{T} = \mathbf{t}_\mathsf{p} \times \{0,1\}^*$ consisting of three disjoint sets: injective tags $\mathcal{T}_\mathsf{inj}$, lossy tags $\mathcal{T}_\mathsf{loss}$, and invalid tags $\mathcal{T}_\mathsf{invalid}$. All tags have form $\mathsf{tag} = (\mathbf{t}_\mathsf{p}, \mathbf{t}_\mathsf{a})$ where $\mathbf{t}_\mathsf{p}$ is the primary part of the tag, and $\mathbf{t}_\mathsf{a} \in \{0,1\}^*$ is the auxiliary part of the tag.
- $\mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, X)$: an evaluation algorithm. It takes as input $\mathsf{ABM.ek}$, a tag $\mathsf{tag} \in \mathcal{T}$, and $X \in \mathsf{D}$. It produces $Y = \mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, X)$.
- $\mathsf{ABM.Inv}(\mathsf{ABM.ik}, \mathsf{tag}, Y)$: an inversion algorithm. It takes as input $\mathsf{ABM.ik}$, a injective tag $\mathsf{tag} \in \mathcal{T}_\mathsf{inj}$ and Y, where $Y = \mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, X)$. It outputs $X = \mathsf{ABM.Inv}(\mathsf{ABM.ik}, \mathsf{tag}, Y)$.
- $\mathsf{ABM.LTag}(\mathsf{ABM.tk})$: a lossy tag generation algorithm. It uses $\mathsf{ABM.tk}$ to generate a lossy tag $\mathsf{tag} \in \mathcal{T}_\mathsf{loss}$.

We require the following properties of ABM-LTFs.

Invertibility. The invertibility property consists of two sub-properties. Firstly, it requires that randomly sampled tags be injective tags with all but negligible probability, i.e.,

$$\Pr\left[\mathsf{tag} \in \mathcal{T}_\mathsf{inj} \mid \mathsf{tag} \xleftarrow{\$} \mathcal{T}\right] \geq 1 - \mathsf{negl}(\lambda)$$

for some negligible function $\mathsf{negl}(\lambda)$ in the security parameter λ. Secondly, it requires that for all injective tags, the ABM-LTF be invertible with all bat negligible probability. That is, for all $(\mathsf{ABM.ek}, \mathsf{ABM.ik}, \mathsf{ABM.tk}) \leftarrow \mathsf{ABM.Gen}(1^\lambda)$, $\mathsf{tag} \in \mathcal{T}_\mathsf{inj}$, $X \in \mathsf{D}$, and $Y = \mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, X)$ we have

$$\Pr\left[\mathsf{ABM.Inv}(\mathsf{ABM.ik}, \mathsf{tag}, Y) = X\right] = 1 - \mathsf{negl}(\lambda)$$

Lossiness. An ABM-LTF is ℓ-lossy if for all $(\mathsf{ABM.ek}, \mathsf{ABM.ik}, \mathsf{ABM.tk}) \leftarrow \mathsf{ABM.Gen}(1^\lambda)$, and all $\mathsf{tag} \in \mathcal{T}_\mathsf{loss}$, the image set $\mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, \mathsf{D})$ has size $\leq |\mathsf{D}|/2^\ell$.

Indistinguishability. The indistinguishability property requires that even multiple lossy tags be indistinguishable from random tags. That is, for all PPT adversary $\mathcal{A}$'s, the advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda)$ given by

$$\left|\Pr\left[\mathcal{A}^{\mathsf{ABM.LTag}(\mathsf{ABM.tk},\cdot)}(1^\lambda,\mathsf{ABM.ek})=1\right]-\Pr\left[\mathcal{A}^{\mathcal{O}_{\mathcal{T}}(\cdot)}(1^\lambda,\mathsf{ABM.ek})=1\right]\right|$$

is negligible in λ, where $(\mathsf{ABM.ek},\mathsf{ABM.ik},\mathsf{ABM.tk})\leftarrow\mathsf{ABM.Gen}(1^\lambda)$, the call $\mathsf{ABM.LTag}(\mathsf{ABM.tk},\cdot)$ returns a lossy tag, and $\mathcal{O}_{\mathcal{T}}(\cdot)$ returns a random tag in $\mathcal{T}$.

Evasiveness. Evasiveness asks that lossy and invalid tags be computationally hard to find, even given multiple lossy tags. That is, for all PPT adversary $\mathcal{A}$, for $(\mathsf{ABM.ek},\mathsf{ABM.ik},\mathsf{ABM.tk})\leftarrow\mathsf{ABM.Gen}(1^\lambda)$, $\mathcal{A}$ has negligible advantage

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda)=\Pr\left[\mathcal{A}^{\mathsf{ABM.LTag}(\mathsf{ABM.tk},\cdot),\mathcal{O}(\cdot)}(1^\lambda,\mathsf{ABM.ek})=\mathsf{tag}\in\mathcal{T}_{\mathsf{loss}}\cup\mathcal{T}_{\mathsf{invalid}}\right]$$

where the oracle $\mathcal{O}(\cdot)$ takes as input a tag tag output from $\mathcal{A}$ and returns answers "lossy/invalid" and "injective" indicating the type of tag.

4 All-But-Many Lossy Trapdoor Function from LWE

We now present our main construction, which borrows and combines various ideas from many different sources, primarily [29,10,7,14]; we also credit an anonymous source for suggesting the marriage of weak PRFs with chameleon hashing.

4.1 Basic LTF from [10]

We recall the lattice-based LTF proposed by Bellare et al. [10], which is the basis of our ABM-LTF construction.

Let $c>1$ and $b\geq 2$ be two constants. Let $n_1\geq 2$ be an integer, $q\geq 2$ be a large enough prime. Let $n=cn_1$ and $w=\log_b q$. Let $\bar{m}$ be any integer such that $\bar{m}>n\log_b q+\omega(\log n)$, and $m=\bar{m}+2nw=\Theta(n\log_b q)$. Let β and γ be integers such that $1<\gamma<\beta<q$. Define $I_\beta=\{0,1,\cdots,\beta-1\}$ and $I_\gamma=\{0,1,\cdots,\gamma-1\}$. Let $\hat{\mathbf{G}}\in\mathbb{Z}_q^{2n\times 2nw}$ be the gadget matrix.

- $\mathsf{LTF.Gen}(1^\lambda,\mathsf{loss})$ The lossy function generation algorithm dose the following:
  1. Sample $\mathbf{A'}\in\mathbb{Z}_q^{n_1\times\bar{m}}$, $\mathbf{E}_1\leftarrow\chi^{\bar{m}\times(n-n_1)}$, $\mathbf{E}_2\leftarrow\chi^{n_1\times(n-n_1)}$.
  2. Compute $\mathbf{A}=\begin{bmatrix}\mathbf{A'}\\ \mathbf{E}_1^t+\mathbf{E}_2^t\mathbf{A'}\end{bmatrix}\in\mathbb{Z}_q^{n\times\bar{m}}$.
  3. Sample $\mathbf{R}\leftarrow D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$.
  4. Set $\mathsf{LTF.ek}$: $\mathbf{F}=[\mathbf{A}|\mathbf{AR}]\in\mathbb{Z}_q^{n\times(\bar{m}+2nw)}$.
- $\mathsf{LTF.Gen}(1^\lambda,\mathsf{inj})$ The injective trapdoor function generation algorithm does the following:
  1. Sample $\mathbf{A'}\in\mathbb{Z}_q^{n_1\times\bar{m}}$, $\mathbf{E}_1\leftarrow\chi^{\bar{m}\times(n-n_1)}$, $\mathbf{E}_2\leftarrow\chi^{n_1\times(n-n_1)}$.
  2. Compute $\mathbf{A}=\begin{bmatrix}\mathbf{A'}\\ \mathbf{E}_1^t+\mathbf{E}_2^t\mathbf{A'}\end{bmatrix}\in\mathbb{Z}_q^{n\times\bar{m}}$.

3. Sample $\mathbf{R} \leftarrow D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$ and $\mathbf{H} \in \mathbb{Z}_q^{n\times 2n}$ with rank n.
4. Set $\mathsf{LTF.ek} = \mathbf{F} = [\mathbf{A}|\mathbf{AR}+\mathbf{H}\hat{\mathbf{G}}] \in \mathbb{Z}_q^{n\times(\bar{m}+2nw)}$ and $\mathsf{LTF.ik} = (\mathbf{R}, \mathbf{H}, \hat{\mathbf{G}})$

– $\mathsf{LTF.Eval}(\mathsf{LTF.ek}, \mathbf{x})$ For $\mathbf{x} \in I_\beta^{m+n_1} \times I_\gamma^{n-n_1}$, the evaluation algorithm returns

$$\mathbf{y}^t = g_{\mathbf{F}}(\mathbf{x}) = \mathbf{x}^t \begin{bmatrix} \mathbf{I}_m \\ \mathbf{F} \end{bmatrix} \bmod q$$

– $\mathsf{LTF.Inv}(\mathsf{LTF.ik}, \mathbf{y})$ Given $\mathbf{y}$, the inversion algorithm outputs $\mathbf{x} = \mathsf{Invert}(\mathbf{F}, \mathbf{R}, \mathbf{H}, \mathbf{y})$.

The invertibility of the basic lossy trapdoor function directly relies on Lemma 7. The lossiness and the indistinguishability of the function $g_{\mathbf{F}}(\cdot)$ relies on the following two lemmas.

Lemma 9 (Lemma 5.4, [9]). *Let $\mathbf{F} = [\mathbf{A}|\mathbf{AR}] \in \mathbb{Z}_q^{n\times m}$ be as generated by $\mathsf{LTF.Gen}(1^\lambda, \mathsf{loss})$ under the conditions $\gamma^{c-1} \geq 2^{\Omega(m/n_1)}$ and $\beta \geq \gamma \cdot s_1(\tilde{\mathbf{E}})$ where $\tilde{\mathbf{E}}^t = [\mathbf{E}_1^t|\mathbf{E}_1^t \cdot \mathbf{R}|\mathbf{E}_2^t]$. The function $g_{\mathbf{F}}(\mathbf{x}) = \mathbf{x}^t \begin{bmatrix} \mathbf{I}_m \\ \mathbf{F} \end{bmatrix} \bmod q$, where $\mathbf{x} \in I_\beta^{m+n_1} \times I_\gamma^{n-n_1}$, is an $\Omega(m)$-lossy function.*

Lemma 10 (Lemma 5.7, [9]). *For any PPT adversary $\mathcal{A}$ against the indistinguishability of above LTF with advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{A}}(\lambda)$, there exists an adversary $\mathcal{B}$ against $\mathsf{LWE}_{n_1,q,\chi}$ such that*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{A}}(\lambda) \leq 2 \cdot \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{B}} + \mathsf{negl}(\lambda)$$

for some negligible probability $\mathsf{negl}(\lambda)$.

4.2 Our Construction of ABM-LTF

Let $n_1 \geq 2$, $\bar{m} \geq 2$ be integers, $q \geq 2$ be a prime. Let $n = cn_1$, $w = \log_b q$ for constants c and b. Set $m = \bar{m} + 2nw$. Let β and γ be integers such that $1 < \gamma < \beta < q$. Define $I_\beta = \{0, 1, \cdots, \beta - 1\}$ and $I_\gamma = \{0, 1, \cdots, \gamma - 1\}$. Let $\mathsf{CH} = (\mathsf{CH.Gen}, \mathsf{CH.Eval}, \mathsf{CH.Equiv})$ be a secure chameleon hash function with equivocation indistinguishability. Let $\mathcal{UH} = \{\mathsf{UH}_{\mathbf{s}} : \{0,1\}^{\ell'} \to \{0,1\}^{\ell}\}$ for $\mathbf{s} \in \{0,1\}^{t'}$.

– $\mathsf{ABM.Gen}(1^\lambda, d)$ The key generation algorithm does the following steps:

1. Choose $\mathbf{A}' \xleftarrow{\$} \mathbb{Z}_q^{n_1\times\bar{m}}$, $\mathbf{E}_2 \xleftarrow{\$} \chi^{n_1\times(n-n_1)}$, $\mathbf{E}_1 \leftarrow \chi^{\bar{m}\times(n-n_1)}$ and set

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}' \\ \mathbf{E}_2^t\mathbf{A}' + \mathbf{E}_1^t \end{bmatrix} \in \mathbb{Z}_q^{n\times\bar{m}}$$

2. Select a weak PRF $\mathsf{WPRF} : \{0,1\}^t \times \{0,1\}^\ell \to \{0,1\}$. Select $\mathbf{K} \xleftarrow{\$} \{0,1\}^{h\times t}$. We denote by $\mathbf{k}_i \in \{0,1\}^t$ the i-th row of $\mathbf{K}$, to serve as an independent key for WPRF. We denote by $k_{i,j} \in \{0,1\}$ the j-th bit of $\mathbf{k}_i$. Select a universal hash function $\mathsf{UH}_{\mathbf{s}} \xleftarrow{\$} \mathcal{UH}$ with hidden key $\mathbf{s} = s_1 \ldots s_{t'} \in \{0,1\}^{t'}$. Express the function $\mathsf{WPRF}(\cdot, \mathsf{UH}.(\cdot))$ as a Boolean circuit C_{WPRF} with gate fan-in 2 and depth d.

3. Sample a set of low-norm matrices $\{\mathbf{R}_{k_{i,j}}\}_{i\in[h],j\in[t]}$, $\{\mathbf{R}_{s_i}\}_{i\in[t']}$ from the distribution $D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$. Compute $\mathbf{C}_{k_{i,j}} = \mathbf{A}\mathbf{R}_{k_{i,j}} + k_{i,j}\mathbf{G}$ and $\mathbf{C}_{s_i} = \mathbf{A}\mathbf{R}_{s_i} + s_i\mathbf{G}$. [5]
4. Sample a set of low-norm matrices $\{\mathbf{R}_{\mathbf{H}_i}\}_{i\in[h]}$ for $\mathbf{R}_{\mathbf{H}_i} \leftarrow D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$. Sample a set of random rank-n matrices $\{\mathbf{H}_i\}_{i\in[h]}$ for $\mathbf{H}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times 2n}$. Compute $\hat{\mathbf{C}}_{\mathbf{H}_i} = \mathbf{A}\mathbf{R}_{\mathbf{H}_i} + \mathbf{H}_i\hat{\mathbf{G}} \in \mathbb{Z}_q^{n\times 2nw}$ for $i \in [h]$. [6]
5. Select $\mathbf{Z} \leftarrow D^{\bar{m}\times 2nw}_{\mathbb{Z},b\cdot\omega(\sqrt{\log n})}$, and compute $\hat{\mathbf{C}}_{\mathbf{Z}} = \mathbf{A}\mathbf{R}_{\mathbf{Z}} + \mathbf{Z}\hat{\mathbf{G}}$.
6. Run $\mathsf{CH.Gen}(1^\lambda)$ to generate a chameleon hash key Hk and a trapdoor Td. Assume this chameleon hash function has message space $\mathcal{X}_{\mathsf{CH}} = \{0,1\}^*$, randomness space $\mathcal{R}_{\mathsf{CH}}$ and output space $\{0,1\}^{\ell'}$.
7. Set the public evaluation key

$$\mathsf{ABM.ek} = \begin{pmatrix} \mathsf{WPRF}, C_{\mathsf{WPRF}}, \mathbf{A}, \{\mathbf{C}_{k_{i,j}}\}_{i\in[h],j\in[t]}, \\ \{\mathbf{C}_{s_i}\}_{i\in[t']}, \{\hat{\mathbf{C}}_{\mathbf{H}_i}\}_{i\in[h]}, \hat{\mathbf{C}}_{\mathbf{Z}}, \mathsf{Hk} \end{pmatrix}$$

the private inversion key

$$\mathsf{ABM.ik} = \begin{pmatrix} \mathsf{WPRF}, C_{\mathsf{WPRF}}, \mathbf{K}, \mathbf{s}, \{\mathbf{R}_{k_{i,j}}\}_{i\in[h],j\in[t]}, \\ \{\mathbf{R}_{s_i}\}_{i\in[t']}, \{\mathbf{H}_i\}_{i\in[h]}, \{\mathbf{R}_{\mathbf{H}_i}\}_{i\in[h]}, \mathbf{Z}, \mathbf{R}_{\mathbf{Z}} \end{pmatrix}$$

and the lossy tag generation key

$$\mathsf{ABM.tk} = \big(\mathsf{WPRF}, C_{\mathsf{WPRF}}, \mathbf{K}, \mathbf{s}, \{\mathbf{H}_i\}_{i\in[h]}, \mathbf{Z}, \mathsf{Td}\big)$$

– **Tags.** A tag has form $\mathsf{tag} = (\mathbf{t}_\mathsf{p}, \mathbf{t}_\mathsf{a})$. The primary tag part $\mathbf{t}_\mathsf{p} = (\mathbf{D}, R) \in \mathbb{Z}_q^{2n\times 2n} \times \mathcal{R}_{\mathsf{CH}}$ and the auxiliary tag part $\mathbf{t}_\mathsf{a} \in \{0,1\}^*$. Set the tag space as $\mathcal{T} = \mathbb{Z}_q^{2n\times 2n} \times \mathcal{R}_{\mathsf{CH}} \times \{0,1\}^*$. With a tag $\mathsf{tag} = ((\mathbf{D}, R), \mathbf{t}_\mathsf{a})$, we can compute $\mu = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}, \mathbf{t}_\mathsf{a}); R) \in \{0,1\}^{\ell'}$. Let

$$\mathbf{H} = \mathbf{Z}\mathbf{D} - \sum_{i=1}^{h} \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_\mathbf{s}(\mu)) \cdot \mathbf{H}_i \pmod q$$

We define

$$\mathsf{tag} \in \begin{cases} \mathcal{T}_{\mathsf{inj}} & \text{if } \mathbf{H} \text{ has rank } n; \\ \mathcal{T}_{\mathsf{loss}} & \text{if } \mathbf{H} = \mathbf{0}; \\ \mathcal{T}_{\mathsf{invalid}} & \text{if } \mathbf{H} \text{ has rank} \neq 0 \text{ and} \neq n. \end{cases}$$

– $\mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, \mathbf{x})$ For input $\mathbf{x} \in I_\beta^{m+n_1} \times I_\gamma^{n-n_1}$, the algorithm does:
1. Let $\mathsf{tag} = (\mathbf{t}_\mathsf{p}, \mathbf{t}_\mathsf{a}) = ((\mathbf{D}, R), \mathbf{t}_\mathsf{a}) \in \mathcal{T}$, compute $\mu = \mathsf{CH.Eval}\left((\mathbf{D}, \mathbf{t}_\mathsf{a}); R\right) \in \{0,1\}^{\ell'}$.
2. Let $\mu_i \in \{0,1\}$ be the i-th bit of μ. Compute

$$\begin{aligned} \tilde{\mathbf{C}}_i &= \mathsf{Eval}_{\mathsf{BV}}(C_{\mathsf{WPRF}}, \mathbf{C}_{k_{i,1}}, \ldots, \mathbf{C}_{k_{i,t}}, \mathbf{C}_{s_1}, \ldots, \mathbf{C}_{s_{t'}}, \mu_1\mathbf{G}, \ldots, \mu_{\ell'}\mathbf{G}) \\ &= \mathbf{A}\tilde{\mathbf{R}}_i + \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_\mathbf{s}(\mu))\mathbf{G} \pmod q \end{aligned}$$

for some low-norm $\tilde{\mathbf{R}}_i \in \mathbb{Z}^{\bar{m}\times 2nw}$ and $i \in [h]$.

[5] $\mathbf{G}$ is the gadget matrix with dimensions n-by-$2nw$.

[6] $\hat{\mathbf{G}}$ is the gadget matrix with dimensions $2n$-by-$2nw$.

3. Compute $\bar{\mathbf{C}} = \hat{\mathbf{C}}_{\mathbf{Z}}\hat{\mathbf{G}}^{-1}(\mathbf{D}\hat{\mathbf{G}}) = \mathbf{A}(\mathbf{R}_{\mathbf{Z}}\hat{\mathbf{G}}^{-1}(\mathbf{D}\hat{\mathbf{G}})) + (\mathbf{ZD})\hat{\mathbf{G}} = \mathbf{A}\bar{\mathbf{R}} + (\mathbf{ZD})\mathbf{G}$, where $\bar{\mathbf{R}} \in \mathbb{Z}^{\bar{m}\times 2nw}$ is of low norm.
4. Set

$$\begin{aligned}\mathbf{F} &= [\mathbf{A}|\bar{\mathbf{C}}] - [\mathbf{0}|\sum\nolimits_{i=1}^{h} \tilde{\mathbf{C}}_i \cdot \mathbf{G}^{-1}(\hat{\mathbf{C}}_{\mathbf{H}_i})] \bmod q \\ &= [\mathbf{A}|\mathbf{AR} + (\mathbf{ZD} - \sum\nolimits_{i=1}^{h}(\mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{H}_i)\hat{\mathbf{G}}] \bmod q \\ &= [\mathbf{A}|\mathbf{AR} + \mathbf{H}\hat{\mathbf{G}}] \bmod q\end{aligned}$$

for the unknown low-norm $\mathbb{Z}^{\bar{m}\times 2nw}$-matrix

$$\mathbf{R} = \bar{\mathbf{R}} - \sum\nolimits_{i=1}^{h}\left(\tilde{\mathbf{R}}_i \cdot \mathbf{G}^{-1}(\hat{\mathbf{C}}_{\mathbf{H}_i}) + \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{R}_{\mathbf{H}_i}\right) \quad (1)$$

Notice that here $\mathbf{R}$ is unknown to the the function evaluator, and, however, is known to the inversion algorithm $\mathsf{ABM.Inv}$ which has the knowledge of $\mathsf{ABM.ik}$.
5. Compute the output of the function $\mathbf{y}^t = g_{\mathbf{F}}(\mathbf{x}) = \mathbf{x}^t \begin{bmatrix}\mathbf{I}_{\bar{m}+2nw} \\ \mathbf{F}\end{bmatrix} \bmod q$.

– $\mathsf{ABM.Inv}(\mathsf{ABM.ik}, \mathsf{tag}, \mathbf{y})$ The inversion algorithm takes as input an inversion key $\mathsf{ABM.ik}$, an injective tag $\mathsf{tag} \in \mathcal{T}_{\mathsf{inj}}$ and an image $\mathbf{y}$. It does the following:
  1. Let $\mathsf{tag} = ((\mathbf{D}, R), \mathbf{t}_{\mathsf{a}})$, compute $\mu = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}, \mathbf{t}_{\mathsf{a}}); R) \in \{0,1\}^{\ell'}$.
  2. Compute $\mathbf{F} = [\mathbf{A}|\mathbf{AR} + \mathbf{H}\hat{\mathbf{G}}]$ as the algorithm $\mathsf{ABM.Eval}$.
  3. Use the knowledge of $\mathsf{ABM.ik}$ to compute the low-norm $\mathbf{R}$ by the formula 1 and compute $\mathbf{H} = \mathbf{ZD} - \sum_{i=1}^{h} \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{H}_i \pmod q$. Notice $\mathbf{H}$ has rank n.
  4. Call the algorithm $\mathsf{Invert}(\mathbf{F}, \mathbf{R}, \mathbf{H}, \mathbf{y})$ to get $\mathbf{x}$.
– $\mathsf{ABM.LTag}(\mathsf{ABM.tk})$ The lossy tag generation algorithm takes as input the lossy tag generation key $\mathsf{ABM.tk}$. It does the following:
  1. Randomly select a tag $\mathsf{tag}' = ((\mathbf{D}', R'), \mathbf{t}'_{\mathsf{a}}) \in \mathcal{T}$ and compute $\mu = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}', \mathbf{t}'_{\mathsf{a}}); R')$.
  2. Solve for $\mathbf{D} \in \mathbb{Z}_q^{2n\times 2n}$ such that $\mathbf{ZD} = \sum_{i=1}^{h}(\mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{H}_i) \pmod q$.
  3. Randomly select $\mathbf{t}_{\mathsf{a}} \in \{0,1\}^*$.
  4. Compute $R = \mathsf{CH.Equiv}(\mathsf{Td}, ((\mathbf{D}', \mathbf{t}'_{\mathsf{a}}), R'), \mathbf{D})$ and output $\mathsf{tag} = ((\mathbf{D}, R), \mathbf{t}_{\mathsf{a}})$.

  It is easy to check that the algorithm indeed outputs a lossy tag.

4.3 Correctness

We show in the following theorems that our ABM-LTFs are invertible with injective tags and lossy with lossy tags.

Theorem 1. *For our construction, randomly sampled tags are injective tags with all but negligible probability. In addition, for any injective tag $\mathsf{tag} \in \mathcal{T}_{inj}$, the function $g_{\mathbf{F}}(\cdot)$ is invertible with overwhelming probability, where $\mathbf{F} = [\mathbf{A}|\mathbf{AR} + \mathbf{H}\hat{\mathbf{G}}] \in \mathbb{Z}_q^{n\times m}$ was computed via $\mathsf{ABM.Eval}$ with tag.*

Proof. Let $\mathsf{tag} = ((\mathbf{D}, R), \mathbf{t}_\mathsf{p})$ be a randomly sampled tag; that is, $\mathbf{D} \xleftarrow{\$} \mathbb{Z}_q^{2n\times 2n}$, $R \xleftarrow{\$} \mathcal{R}_\mathsf{CH}$ and $\mathbf{t}_\mathsf{p} \xleftarrow{\$} \{0,1\}^*$. We have $\mathbf{ZD} \bmod q$ is uniformly random over $\mathbb{Z}_q^{n\times 2n}$, thus, so is $\mathbf{H}$. By Lemma 1, $\mathbf{H}$ has rank n except negligible probability. Hence, $\mathsf{tag} = ((\mathbf{D}, R), \mathbf{t}_\mathsf{p})$ is an injective tag.

Since $\|\mathbf{x}\| \le \beta \cdot \sqrt{m}$, we can bound β (with large enough q) to ensure that $\|\mathbf{x}\| \le q/\Theta(b \cdot s_1(\mathbf{R}))$. We then apply Lemma 7 to conclude the proof. □

Theorem 2. *With our parameter restrictions (see also parameter selection in Section 4.4), for any lossy tag $\mathsf{tag} \in \mathcal{T}_{\mathsf{loss}}$, the function $g_\mathbf{F}(\cdot)$ is $\Omega(m)$-lossy, where $\mathbf{F} = [\mathbf{A}|\mathbf{AR}] \in \mathbb{Z}_q^{n\times m}$ computed via $\mathsf{ABM.Eval}$ using tag, and $m = \Theta(n \log_b q)$.*

Proof. This proof borrows from the proof of Lemma 9 which follows directly from the proof of Lemma 5.4 of [9].

By the construction of $\mathbf{F} \in \mathbb{Z}_q^{n\times m}$ we have

$$\begin{aligned} g_\mathbf{F}(\mathbf{x}) = \mathbf{x}^t \begin{bmatrix} \mathbf{I}_m \\ \mathbf{F} \end{bmatrix} \bmod q &= (\mathbf{x}^t \begin{bmatrix} \mathbf{I}_{\bar{m}} & & \\ & \mathbf{I}_{2nw} & \\ & & \mathbf{I}_{n_1} \\ \mathbf{E}_1^t & \mathbf{E}_1^t \cdot \mathbf{R} & \mathbf{E}_2^t \end{bmatrix}) \begin{bmatrix} \mathbf{I}_{\bar{m}} & \\ & \mathbf{I}_{2nw} \\ \mathbf{A}' & \mathbf{A}'\mathbf{R} \end{bmatrix} \bmod q \\ &= (\mathbf{x}^t \begin{bmatrix} \mathbf{I}_{m+n_1} \\ \tilde{\mathbf{E}}^t \end{bmatrix}) \begin{bmatrix} \mathbf{I}_m \\ \mathbf{F}' \end{bmatrix} \bmod q \end{aligned}$$

It suffices to bound the number of possible values of $\mathbf{x}^t \begin{bmatrix} \mathbf{I}_{m+n_1} \\ \tilde{\mathbf{E}}^t \end{bmatrix} \in \mathbb{Z}^{n_1+m}$.

By the triangle inequality, we have

$$\left\| \mathbf{x}^t \begin{bmatrix} \mathbf{I}_{m+n_1} \\ \tilde{\mathbf{E}}^t \end{bmatrix} \right\| \le \beta\sqrt{n_1+m} + s_1(\tilde{\mathbf{E}}) \cdot \gamma\sqrt{n-n_1} \le \sqrt{n_1+m} \cdot (\beta + \gamma \cdot s_1(\tilde{\mathbf{E}}))$$

Define $N_d(r)$ to be the number of integer points in a d-dimensional Euclidean ball of radius r. For $r \ge \sqrt{d}$, from the volume of the ball and Stirling's approximation, we have $N_d(r) = O(r/\sqrt{d})^d$. So the number of possible values of $\mathbf{x}^t \begin{bmatrix} \mathbf{I}_{m+n_1} \\ \tilde{\mathbf{E}}^t \end{bmatrix}$ is $O(\beta + \gamma \cdot s_1(\tilde{\mathbf{E}}))^{n_1+m}$.

By the structure of $\mathbf{F}$, $\gamma \ge 2^{\Omega(m/n_1)}$ and $\gamma \le q^{1/C}$, the base-2 logarithm of the domain of the function $g_\mathbf{F}(\cdot)$ is

$$(n_1+m)\log\beta + n_1 \log\gamma^{c-1} \ge (n_1+m)\log\beta + \Omega(m)$$

Since $\beta \ge \gamma \cdot s_1(\tilde{\mathbf{E}})$, the base-2 logarithm of the range of the function $g_\mathbf{F}(\cdot)$ is at most

$$(n_1+m)\log O(\beta + \gamma \cdot s_1(\tilde{\mathbf{E}})) = (n_1+m)\log\beta + O(m)$$

By choosing a sufficiently large constant in the Ω notation, we have $\log|\mathsf{D}| - \log|\mathsf{R}| = \Omega(m)$. We conclude that the function $g_\mathbf{F}(\cdot)$ is $\Omega(m)$-lossy. □

Setting β and γ. The restrictions on β and γ originate from two lemmas.

Firstly, for invertibility (Lemma 7), we need $\|\mathbf{x}\| \leq \beta\sqrt{m} < q/\Theta(b \cdot s_1(\mathbf{R}))$.

Secondly, for lossiness (Lemma 9), we need $\gamma^{c-1} \geq 2^{\Omega(m/n_1)}$ where $m = \Theta(n \log_b q) = \Theta(cn_1 \log_b q)$, and $\gamma \cdot s_1(\tilde{\mathbf{E}}) \leq \beta$; hence $\gamma \geq q^{\Theta(1/\log b)\cdot c/(c-1)}$.

For any desired constant $C > 1$, we can set up constants $c > 1$ and $b \geq 2$ so that $\gamma \leq q^{1/C}$. This gives

$$q^{1/C} \cdot s_1(\tilde{\mathbf{E}}) \leq \beta \leq q/\Theta(b \cdot s_1(\mathbf{R}) \cdot \sqrt{m}) \tag{2}$$

Therefore, it is sufficient to take q large enough such that

$$q^{1-1/C} \geq \Omega\left(s_1(\mathbf{R}) \cdot s_1(\tilde{\mathbf{E}}) \cdot \sqrt{m}\right) \tag{3}$$

4.4 Parameter Selections

An instance of parameter selection that meets all requirements of correctness and security properties is given here.

Firstly, to enable the statistical argument for security, i.e., Lemma 5 and Lemma 3, we set $\bar{m} > n \log_b q + \omega(\log n)$, and for any $\epsilon > 0$, set $h = \mathsf{poly}(\lambda)$ such that $\log(q/(\epsilon^2)) \leq h$.

We set the constant $C = 6$ for Equation (2), which we can do by picking a suitable constant c and logarithm radix b.

Instantiating WPRF by the weak PRF from [7], which has fan-in-2 Boolean circuit implementation in class NC^1, and a universal hash function from Corollary 1, we can get the fan-in-2 Boolean circuit C_{WPRF} in class NC^1, i.e., C_{WPRF} has input length $\ell' + t' + t = \mathsf{poly}(\lambda)$, and depth $\eta \log(\ell' + t' + t) = \eta \log(\mathsf{poly}(\lambda))$, for some constant $\eta > 0$.

We now bound the norm of $\mathbf{R} \in \mathbb{Z}^{\bar{m} \times 2nw}$ per the formula 1. Firstly we have

$$\begin{aligned}
& s_1\left(\textstyle\sum_{i=1}^{h} \tilde{\mathbf{R}}_i \cdot \mathbf{G}^{-1}(\hat{\mathbf{C}}_{\mathbf{H}_i}) + \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{R}_{\mathbf{H}_i}\right) \\
\leq\ & O\left(h \cdot 4^d \cdot \bar{m}^{3/2}\right) \cdot ((b-1) \cdot 2nw) \quad \text{(Lemma 8, 6)} \\
\leq\ & O\left(h \cdot 4^d \cdot \bar{m}^{3/2} \cdot \bar{m}\right) \qquad ((b-1) \cdot 2nw \leq O(\bar{m})) \\
\leq\ & O\left(h \cdot 4^d \cdot \bar{m}^2\right)
\end{aligned}$$

and

$$\begin{aligned}
s_1(\bar{\mathbf{R}}) &\leq 3 \cdot b \cdot \omega(\sqrt{\log n}) \cdot O(\sqrt{\bar{m}}) \cdot (b-1) \cdot 2nw \\
&\leq \tilde{O}(\bar{m}^{3/2})
\end{aligned}$$

So we have

$$\begin{aligned}
s_1(\mathbf{R}) &\leq s_1\left(\sum_{i=1}^{h} \tilde{\mathbf{R}}_i \cdot \mathbf{G}^{-1}(\hat{\mathbf{C}}_{\mathbf{H}_i}) + \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu))) \cdot \mathbf{R}_{\mathbf{H}_i}\right) + s_1(\bar{\mathbf{R}}) \\
&\leq O\left(h \cdot 4^d \cdot \bar{m}^2\right) + \tilde{O}(\bar{m}^{3/2}) \\
&= \tilde{O}(h \cdot 4^d \cdot n_1^2)
\end{aligned} \tag{4}$$

We now choose the LWE noise distribution $\chi = D_{\mathbb{Z},2\sqrt{n_1}}$ for accommodating the average-case to worst-case hardness reduction from classical lattice problems, e.g. SIVP, given by [41]. We bound $s_1(\tilde{\mathbf{E}})$ where $\tilde{\mathbf{E}}^t = [\mathbf{E}_1^t | \mathbf{E}_1^t \cdot \mathbf{R} | \mathbf{E}_2^t]$ according to Lemma 9.

$$\begin{aligned} s_1(\tilde{\mathbf{E}}) &\leq s_1(\mathbf{E})(1 + s_1(\mathbf{R})) \\ &\leq 2\sqrt{n_1} \cdot (\sqrt{\bar{m} + n_1} + \sqrt{n - n_1})(1 + s_1(\mathbf{R})) \qquad \text{(by Lemma 4)} \\ &= \tilde{O}(h \cdot 4^d \cdot n_1^3) \end{aligned}$$

We now set q through Equation (3) as

$$\begin{aligned} q &= \Theta\left((s_1(\mathbf{R}) \cdot s_1(\tilde{\mathbf{E}}) \cdot \sqrt{m})^{C/(C-1)}\right) \\ &= \tilde{\Theta}\left((h \cdot 4^d \cdot n_1^3 \cdot h \cdot 4^d \cdot n_1^2 \cdot n_1^{0.5})^{C/(C-1)}\right) \\ &= \tilde{\Theta}\left(h^{2.4} \cdot 2^{4.8d} \cdot n_1^{6.6}\right) \end{aligned}$$

Lastly we fix $\gamma = \tilde{O}\left((h^2 \cdot 2^{4d} \cdot n_1^{5.5})^{1/(C-1)}\right) = \tilde{O}\left(h^{0.4} \cdot 2^{0.8d} \cdot n_1^{1.1}\right) \leq q^{1/C}$. To fix β we have $\gamma \cdot s_1(\tilde{\mathbf{E}}) = \tilde{O}\left(h^{1.4} \cdot 2^{2.8} \cdot n_1^{4.1}\right)$ and $q/\Theta(b \cdot s_1(\mathbf{R})\sqrt{m}) = \tilde{O}\left(h^{2.4} \cdot 2^{2.8d} \cdot n_1^{4.1}\right)$, so to satisfy Equation (2), we set

$$\gamma \cdot s_1(\tilde{\mathbf{E}}) \leq \beta = \tilde{\Theta}\left(h^{2.4} \cdot 2^{4.8d} \cdot n_1^{6.6}\right) \leq q/\Theta(b \cdot s_1(\mathbf{R})\sqrt{m})$$

Summing up, an example of parameter selection per the foregoing, is:

$$d = O\left(\log(\mathsf{poly}(\lambda)\right) \quad ; \quad q = \tilde{\Theta}\left(h^{2.4} \cdot 2^{4.8d} \cdot n_1^{6.6}\right) \quad ; \quad m = \Theta(n_1 \log_b q)$$

$$\beta = \tilde{\Theta}\left(h^{2.4} \cdot 2^{4.8d} \cdot n_1^{6.6}\right) \quad ; \quad \gamma = \tilde{O}\left(h^{0.4} \cdot 2^{0.8d} \cdot n_1^{1.1}\right)$$

4.5 Security Proofs

Theorem 3 (Indistinguishability). *For any PPT adversary $\mathcal{A}$ against indistinguishablity of the above ABM-LTF with advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},\mathcal{A}}(\lambda)$, there exist two adversaries $\mathcal{A}_1$, $\mathcal{A}_2$ and a negligibly small error $\mathsf{negl}(\lambda)$ such that*

$$\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda) + h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \mathsf{negl}(\lambda) + \epsilon$$

Proof. We proceed with the proof using a game sequence. Let S_i be the event that $\mathcal{A}$ outputs 1 in the game **Game** i. In **Game** 1, all algorithms work exactly the same as the real scheme. $\mathcal{A}$ interacts with $\mathsf{ABM.LTag}(\mathsf{ABM.tk}, \cdot)$ which outputs lossy tags. So we have

$$\Pr[S_1] = \Pr\left[\mathcal{A}^{\mathsf{ABM.LTag}(\mathsf{ABM.tk},\cdot)}(1^\lambda, \mathsf{ABM.ek}) = 1\right]$$

In **Game** 2, we change the way of generating public matrix $\mathbf{A}$. Particularly, we sample $\mathbf{A}$ from $\mathbb{Z}_q^{n \times \bar{m}}$ uniformly at random. Because $\mathbf{A}$ does not affect the

output distribution of $\mathsf{ABM.LTag}$, by the LWE assumption, this change is not noticeable to $\mathcal{A}$, lest it give an LWE distinguisher. So we have

$$|\Pr[S_2] - \Pr[S_1]| \leq \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda)$$

for a suitable $\mathsf{LWE}_{n_1,q,\chi}$ adversary $\mathcal{A}_1$.

In **Game** 3, the public evaluation key of the ABM-LTF is set as

$$\mathsf{ABM.ek} = \left(\mathsf{WPRF}, C_{\mathsf{WPRF}}, \mathbf{A}, \{\mathbf{C}_{k_{i,j}}\}_{i\in[h],j\in[t]}, \{\mathbf{C}_{s_i}\}_{i\in[t']}, \{\hat{\mathbf{C}}_{\mathbf{H}_i}\}_{i\in[h]}, \hat{\mathbf{C}}_{\mathbf{Z}}, \mathsf{Hk}\right)$$

where $\{\mathbf{C}_{k_{i,j}}\}_{i\in[h],j\in[t]}$, $\{\mathbf{C}_{s_i}\}_{i\in[t']}$, $\{\hat{\mathbf{C}}_{\mathbf{H}_i}\}_{i\in[h]}$, and $\hat{\mathbf{C}}_{\mathbf{Z}}$ are chosen uniformly random from $\mathbb{Z}_q^{n\times 2nw}$. Accordingly, the low-norm secret matrices in $\mathsf{ABM.ik}$, which include $\{\mathbf{R}_{k_{i,j}}\}_{i\in[h],j\in[t]}$, $\{\mathbf{R}_{s_i}\}_{i\in[t']}$, $\{\mathbf{R}_{\mathbf{H}_i}\}_{i\in[h]}$, and $\mathbf{R}_{\mathbf{Z}}$ are no longer needed. It is easy to see that this change does not affect the (output distribution of) algorithm $\mathsf{ABM.LTag}$. Moreover, by Lemma 5, $\mathsf{ABM.ek}$ in **Game** 3 has a distribution that is statistically close to the distribution of $\mathsf{ABM.ek}$ in **Game** 2. So for some negligibly small statistical error $\mathsf{negl}(\lambda)$, we have

$$|\Pr[S_3] - \Pr[S_2]| \leq \mathsf{negl}(\lambda)$$

In **Game** 4, we change the algorithm $\mathsf{ABM.LTag}$. Specifically, in step 2 of $\mathsf{ABM.LTag}$, we compute $\mathsf{r}_i(\mathsf{UH}_{\mathbf{s}}(\mu))$ with random functions $\mathsf{r}_i : \{0,1\}^{\ell} \to \{0,1\}$ instead of $\mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu))$ for $i \in [h]$. (Note this does not affect $\mathsf{ABM.Eval}$ which still uses C_{WPRF}.) As μ is uniformly random, for a PPT adversary $\mathcal{A}_2$ against WPRF, a straightforward hybrid argument shows that

$$|\Pr[S_4] - \Pr[S_3]| \leq h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda)$$

In **Game** 5, we randomly sample a matrix $\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n\times 2n}$ instead of computing $\mathbf{S} = \sum_{i=1}^{h} \mathsf{r}_i(\mathsf{UH}_{\mathbf{s}}(\mu))\,\mathbf{H}_i \bmod q$ as in **Game** 4. By Corollary 1 with $h \geq \log(q/(\epsilon^2))$, the statistical distance between the distribution of the random variable $\sum_{i=1}^{h} \mathsf{r}_i(\mathsf{UH}_{\mathbf{s}}(\mu)) \cdot \mathbf{H}_i \bmod q$ and the uniform distribution over $\mathbb{Z}_q^{n\times 2n}$ is less than ϵ. Hence, we have

$$|\Pr[S_5] - \Pr[S_4]| \leq \epsilon$$

On the other hand, in **Game** 5, $\mathbf{H} = \mathbf{ZD} - \mathbf{S} \bmod q$ with random $\mathbf{S}$. Thus the pair $(\mathbf{D}, R)$ is independent of $\mathbf{H}$. Therefore all tags generated in **Game** 5 are random tags. So we have

$$\Pr[S_5] = \Pr\left[\mathcal{A}^{\mathcal{O}\tau(\cdot)}(1^{\lambda}, \mathsf{ABM.ek}) = 1\right]$$

Summing up, we find that adversary $\mathcal{A}$'s advantage $\mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda)$ is

$$\begin{aligned}&\left|\Pr\left[\mathcal{A}^{\mathsf{ABM.LTag}(\mathsf{ABM.tk},\cdot)}(1^{\lambda}, \mathsf{ABM.ek}) = 1\right] - \Pr\left[\mathcal{A}^{\mathcal{O}\tau(\cdot)}(1^{\lambda}, \mathsf{ABM.ek}) = 1\right]\right| \\ &\leq \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda) + h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \mathsf{negl}(\lambda) + \epsilon \qquad (5)\end{aligned}$$

which completes the proof. □

Theorem 4 (Evasiveness). *For any PPT adversary $\mathcal{A}$ against the evasiveness of the above ABM-LTF with advantage $\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda)$, there exist $\mathcal{A}_1$, $\mathcal{A}_2$, $\mathcal{A}_3$ and a negligible function $\mathsf{negl}(\lambda)$ such that*

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda) \leq \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda)+h\cdot\mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda)+\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{A}_3}(\lambda)+\epsilon+\mathsf{negl}(\lambda)$$

Proof. We prove the theorem using a game sequence. Let S_i be the event that $\mathcal{A}$ outputs a lossy or invalid tag in **Game** i. We further consider two types of (lossy or invalid) tag output by $\mathcal{A}$. We say that a tag $\mathsf{tag} = ((\mathbf{D}^*, R^*), \mathbf{t}^*_{\mathsf{a}})$ has Type I if μ^*, which is equal to $\mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}^*, \mathbf{t}^*_{\mathsf{a}}); R^*)$, is also the chameleon hash output of some previously generated tag. A tag $\mathsf{tag} = ((\mathbf{D}^*, R^*), \mathbf{t}^*_{\mathsf{a}})$ has Type II if $\mu^* = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}^*, \mathbf{t}^*_{\mathsf{a}}); R^*)$ is not the chameleon hash output of any previously generated tag. W.l.o.g., we assume that the adversary gets $N = \mathsf{poly}(\lambda)$ lossy tags $\{\mathsf{tag}_i\}_{i\in[N]} = \{(\mathbf{D}_i, R_i), \mathbf{t}_{\mathsf{a}i}\}_{i\in[N]}$ generated by the lossy tag generation oracle. Then the adversary adaptively comes up with $N' = \mathsf{poly}(\lambda)$ tags $\{\mathsf{tag}^*_i\}_{i\in[N']} = \{(\mathbf{D}^*_i, R^*_i), \mathbf{t}_{\mathsf{a}i}{}^*\}_{i\in[N']}$ and gets answers "lossy/invalid" or "injective" from the oracle $\mathcal{O}$ indicating whether theses tags are lossy/invalid or injective.

In **Game** 1, $\mathcal{A}$ interacts with $\mathsf{ABM.LTag}(\mathsf{ABM.tk}, \cdot)$ which works exactly as in the real system. By hypothesis, we have

$$\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda) = \Pr[S_1]$$

In **Game** 2, we sample the public matrix $\mathbf{A}$ randomly from $\mathbb{Z}_q^{n\times\bar{m}}$. This does not affect the output distribution of $\mathsf{ABM.LTag}$. By the LWE assumption, the change is not noticeable to $\mathcal{A}$; if it is, there is an LWE distinguisher. So we have

$$|\Pr[S_2] - \Pr[S_1]| \leq \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda)$$

for a suitable LWE adversary $\mathcal{A}_1$.

In **Game** 3, the public evaluation ABM-LTFs is set as

$$\mathsf{ABM.ek} = \Big(\mathsf{WPRF}, C_{\mathsf{WPRF}}, \mathbf{A}, \{\mathbf{C}_{k_{i,j}}\}_{i\in[h],j\in[t]}, \{\mathbf{C}_{s_i}\}_{i\in[t']}, \{\hat{\mathbf{C}}_{\mathbf{H}_i}\}_{i\in[h]}, \hat{\mathbf{C}}_{\mathbf{Z}}, \mathsf{Hk}\Big)$$

where $\{\mathbf{C}_{k_{i,j}}\}_{i\in[h],j\in[t]}$, $\{\mathbf{C}_{s_i}\}_{i\in[t']}$, $\{\hat{\mathbf{C}}_{\mathbf{H}_i}\}_{i\in[h]}$, and $\hat{\mathbf{C}}_{\mathbf{Z}}$ are chosen uniformly random from $\mathbb{Z}_q^{n\times 2nw}$. Accordingly, the low-norm secret matrices in $\mathsf{ABM.ik}$, including $\{\mathbf{R}_{k_{i,j}}\}_{i\in[h],j\in[t]}$, $\{\mathbf{R}_{s_i}\}_{i\in[t']}$, $\{\mathbf{R}_{\mathbf{H}_i}\}_{i\in[h]}$, $\mathbf{R}_{\mathbf{Z}}$, are not needed anymore. It is easy to see that this change does not affect the (output distribution of) algorithm $\mathsf{ABM.LTag}$. Moreover, by Lemma 5, $\mathsf{ABM.ek}$ in **Game** 3 has a distribution that is statistically close to the distribution of $\mathsf{ABM.ek}$ in **Game** 2. So for some negligibly small statistical error $\mathsf{negl}_1(\lambda)$, we have

$$|\Pr[S_3] - \Pr[S_2]| \leq \mathsf{negl}_1(\lambda)$$

In **Game** 4, we make the following changes. In step 2 of $\mathsf{ABM.LTag}$, for any μ, instead of computing $\mathbf{S} = \sum_{i=1}^{h} \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu))\mathbf{H}_i \bmod q$, we sample $\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n\times 2n}$. For all queries $\{\mathsf{tag}^*_i\}_{i\in[N']}$ to $\mathcal{O}$, we return the answer "injective".

We prove by induction that distinguishing between this game and the previous one implies a distinguisher for the WPRF. Notice that since the μ_i for all the issued lossy tags are random according to $\mathsf{ABM.LTag}$, their images $\mathsf{UH}_{\mathbf{s}}(\mu_i)$ are also random.

For the base step, suppose $N' = 1$ (the case $N' = 0$ is vacuous). In **Game** 3, $\mathcal{O}$ answers honestly by computing $\mathbf{H} = \mathbf{ZD} - \sum_{i=1}^{h} \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu_1^*))\,\mathbf{H}_i \bmod q$. Since $\mathsf{UH}_{\mathbf{s}}(\mu_1^*)$ is random and jointly random with all independently sampled $\mathsf{UH}_{\mathbf{s}}(\mu_i)$, by Corollary 1 and the security of WPRF, the **Game**-3 distribution $\{\sum_{j=1}^{h} \mathsf{WPRF}(\mathbf{K}_j, \mathsf{UH}_{\mathbf{s}}(\mu_i))\mathbf{H}_i j\}_{i\in[N]} \cup \{\sum_{j=1}^{h} \mathsf{WPRF}(\mathbf{K}_j, \mathsf{UH}_{\mathbf{s}}(\mu_1^*))\mathbf{H}_j\}$ and the **Game**-4 distribution $\{\mathbf{S}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times 2n}\}_{i\in[N]} \cup \{\mathbf{S} \xleftarrow{\$} \mathbb{Z}_q^{n\times 2n}\}$ are computationally distinguishable with probability at most $h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \epsilon$ for a suitable WPRF adversary $\mathcal{A}_2$. Moreover, since for μ_1^* from tag_1^* in **Game** 4 the matrix $\mathbf{S}$ is random, so is $\mathbf{H}$, the adversary always gets the answer "injective" except with negligible probability ε. This shows that $|\Pr[S_4] - \Pr[S_3]| \le h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \epsilon + N' \cdot \varepsilon$ when $N' = 1$.

For the inductive step, assume that the above holds for $k = N' - 1 \ge 1$. Accordingly, in **Game** 4, for tags $\{\mathsf{tag}_i^*\}_{i\in[k]}$, we simply answer "injective" without even looking at the query μ_i^*; we look at the N'-th query tag tag_{k+1}^*. In **Game** 3, we honestly derived the same "injective" answers for the first k guesses, and the last answer is computed as $\mathbf{H} = \mathbf{ZD} - \sum_{i=1}^{h} \mathsf{WPRF}(\mathbf{k}_i, \mathsf{UH}_{\mathbf{s}}(\mu_{k+1}^*))\,\mathbf{H}_i \bmod q$. Since WPRF in **Game** 4 is only evaluated on $\{\mathsf{UH}_{\mathbf{s}}(\mu_i)\}_{i\in[N]} \cup \{\mathsf{UH}_{\mathbf{s}}(\mu_{k+1}^*)\}$ which by construction is jointly uniformly random, and since in **Game** 3 by inductive hypothesis the answers were all "injective", the inductive hypothesis continues to hold, and we have $|\Pr[S_4] - \Pr[S_3]| \le h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \epsilon + k \cdot \varepsilon + \varepsilon$. Therefore we have for all $N' = \mathsf{poly}(\lambda)$, and taking $N' \cdot \varepsilon = \mathsf{negl}_2(\lambda)$,

$$|\Pr[S_4] - \Pr[S_3]| \le h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) + \epsilon + \mathsf{negl}_2(\lambda)$$

Notice that $\{\mathsf{tag}_i\}_{i\in[N]}$ generated in **Game** 4 are distributed as random tags.

In **Game** 5, the trapdoor Td of the chameleon hash function is not available. All primary tags are generated randomly, i.e, $(\mathbf{D}, R) \xleftarrow{\$} \mathbb{Z}_q^{2n\times 2n} \times \mathcal{R}_{\mathsf{CH}}$. Hence,

$$\Pr[S_5] = \Pr[S_4]$$

Moreover, for any fresh μ that was not derived from previous queries, $\mathbf{S} \in \mathbb{Z}_q^{n\times 2n}$ will be chosen randomly and independently. In other words, there does not exist an adversary that outputs Type II tags with more than some negligible probability $\mathsf{negl}_2(\lambda)$. So we have $\Pr[S_{5,\mathrm{I}}] \le \mathsf{negl}_3(\lambda)$. Any Type I output breaches the collision-resistance of the chameleon hash function, therefore $\Pr[S_{5,\mathrm{I}}] \le \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{A}_3}(\lambda)$ for some adversary $\mathcal{A}_3$. Since $\Pr[S_5] \le \Pr[S_{5,\mathrm{I}}] + \Pr[S_{5,\mathrm{II}}]$, we obtain

$$\Pr[S_5] \le \mathsf{negl}_3(\lambda) + \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{A}_3}(\lambda)$$

To sum up, letting $\mathsf{negl}(\lambda) = \mathsf{negl}_1(\lambda) + \mathsf{negl}_2(\lambda) + \mathsf{negl}_3(\lambda) + \epsilon$, we have

$$\begin{aligned}\mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{A}}(\lambda) \le \mathsf{Adv}^{\mathsf{LWE}_{n_1,n-n_1,q,\chi}}_{\mathsf{NF},\mathcal{A}_1}(\lambda) + h \cdot \mathsf{Adv}^{\mathsf{WPRF}}_{\mathcal{A}_2}(\lambda) \\ + \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{A}_3}(\lambda) + \mathsf{negl}(\lambda)\end{aligned} \tag{6}$$

This concludes the proof. □

5 IND-SO-CCA2 Secure PKE from Lattices

Using the constructions from [27,29] as a guide, we build the first LWE-based IND-SO-CCA2-secure public-key encryption scheme with our LWE-based ABM-LTF. In our construction, we take the advantage of the chameleon hash function embedded in our ABM-LTF. Our apprach also draws the idea from [42] in which transformations from tag-based PKE schemes to IND-CCA2 PKE schemes are proposed with the help of chameleon hashing.

5.1 Definition of IND-SO-CCA2 Security

A public-key encryption scheme Π consists of three PPT algorithms: KeyGen, $\mathsf{Encrypt}$ and $\mathsf{Decrypt}$. $\mathsf{KeyGen}(1^\lambda)$ takes as input a security parameter λ, outputs a public key pk and a private key sk. We define the message space $\mathcal{M}_\lambda$, randomness space $\mathcal{R}_\lambda$ and the ciphertext space $\mathcal{C}_\lambda$ in the obvious way. $\mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}; r)$ encrypts a message $\mathsf{m} \in \mathcal{M}_\lambda$ using pk and randomness $r \xleftarrow{\$} \mathcal{R}_\lambda$, and outputs a ciphertext ct. $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$ recovers the message m from ct using sk. The correctness of a PKE scheme requires that for all $\mathsf{m} \in \mathcal{M}_\lambda$, valid randomness $r \in \mathcal{R}_\lambda$, and $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$,

$$\Pr\left[\mathsf{m} = \mathsf{Decrypt}\left(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}; r)\right)\right] \geq 1 - \mathsf{negl}(\lambda)$$

for some negligible function $\mathsf{negl}(\lambda)$.

Selective Opening Security. Suppose that a vector of messages, coming from some joint distribution dist, has been encrypted into a vector of ciphertexts, and sent out. A "selective opening" attack allows an adversary to choose a subset of these ciphertexts and have them "opened", revealing their messages and the random coins used during encryption.

The opened messages, random coins, and distribution dist might help the adversary to learn information about the remaining messages, in the unopend ciphertexts. Selective opening security means that the content of the unopend ciphertexts remains secure in that scenario.

There are a few different ways of formalising selective opening security. As in [29], we are considering the indistinguishability-based definition of security against chosen-ciphertext attacks (referred to as IND-SO-CCA2) with respect to joint message distributions that are efficiently re-sampleable.

Definition 4 (Efficient Resampling). *Let $N = N(\lambda) > 0$, let $\mathcal{M}_\lambda$ be the message space, and let dist be a joint distribution over $\mathcal{M}_\lambda^N$. We say that dist is efficiently re-samplable if there is a PPT algorithm ReSamp such that for any $\mathcal{I} \subset [N]$ and any partial vector $(\mathsf{m}'^{(i)})_{i \in \mathcal{I}} \in \mathcal{M}_\lambda^{|\mathcal{I}|}$, ReSamp samples from the distribution dist, conditioned on $\mathsf{m}^{(i)} = \mathsf{m}'^{(i)}$ for all $i \in \mathcal{I}$.*

The IND-SO-CCA2 security essentially requires that no efficient adversary can distinguish the unopened messages from fresh messages drawn from the same joint distribution conditioned on the opened messages.

Definition 5 (IND-SO-CCA2 Security). *A public-key encryption scheme* $\Pi = (\mathit{KeyGen}, \mathit{Encrypt}, \mathit{Decrypt})$ *has IND-SO-CCA2 security iff for every polynomial* $N = N(\lambda)$*, and every PPT adversary* $\mathcal{A}$*, we have that*

$$\mathit{Adv}^{\mathit{ind\text{-}so\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr\left[\mathit{Exp}^{\mathit{ind\text{-}so\text{-}cca\text{-}b}}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - 1/2\right|$$

is negligible, where the experiment $\mathit{Exp}^{\mathit{ind\text{-}so\text{-}cca\text{-}b}}_{\Pi,\mathcal{A}}(\lambda)$ *is defined in Figure 1.*

Experiment $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\Pi,\mathcal{A}}(\lambda)$

1. $b \xleftarrow{\$} \{0,1\}$
2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
3. $(\mathsf{dist}, \mathsf{ReSamp}) \leftarrow \mathcal{A}^{\mathsf{Decrypt}(\mathsf{sk},\cdot)}(\mathsf{pk})$
4. $\mathbf{m}_0 = (\mathsf{m}^{(i)})_{i\in[N]} \leftarrow \mathsf{dist}$
5. $\mathbf{r} = (r^{(i)})_{i\in[N]} \leftarrow (\mathcal{M}_\lambda)^N$
6. $\mathbf{c} = \left(\mathsf{ct}^{(i)}\right)_{i\in[N]} = \left(\mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}^{(i)}; r^{(i)})\right)_{i\in[N]}$
7. $\mathcal{I} \leftarrow \mathcal{A}^{\mathsf{Decrypt}(\mathsf{sk},\cdot)}(\mathsf{pk}, \mathbf{c})$
8. $\mathbf{m}_1 = \mathsf{ReSamp}(\mathsf{dist}, \mathbf{m}_\mathcal{I})$
9. $b' \leftarrow \mathcal{A}^{\mathsf{Decrypt}(\mathsf{sk},\cdot)}\left(\mathbf{m}_b, \{\mathsf{m}^{(i)}, r^{(i)}\}_{i\in\mathcal{I}}\right)$
10. Return 1 if $b' = b$, and 0 otherwise

Fig. 1: Security experiment of IND-SO-CCA2 security

The adversary $\mathcal{A}$ *is required to output the resampling algorithm* ReSamp *as per Figure 1, and never to submit any challenge ciphertext* $\mathit{ct}^{(i)}$ *to the decryption oracle* $\mathit{Decrypt}(\mathit{sk}, \cdot)$*.*

5.2 Construction of IND-SO-CCA2 PKE

Let λ be the security parameter and $\kappa = \omega(\log\lambda)$. Let $\mathsf{ABM\text{-}LTF} = (\mathsf{ABM.Gen}, \mathsf{ABM.Eval}, \mathsf{ABM.Inv})$ be an l-lossy ABM-LTF with domain $\mathsf{D} = I_\beta^{m+n_1} \times I_\gamma^{n-n_1}$ as constructed before. Assume $\mathsf{X} = I_\beta^{n_1} \times I_\gamma^{n-n_1}$. Let $\mathsf{LTF} = (\mathsf{LTF.Gen}, \mathsf{LTF.Eval}, \mathsf{LTF.Inv})$ be an l'-lossy LTF with domain D. Without loss of generality, we assume $l \geq l'$. Let $\mathcal{UH}$ be a family of universal hash functions from $\mathsf{D} \times I_\beta^m \to \{0,1\}^\tau$ with $\tau \leq (l + l' - \log|\mathsf{X}| - 2\lambda) - 2\log(1/\epsilon)$ for some negligible $\epsilon = \mathsf{negl}(\lambda)$ [7]. Let $B = \Theta(b \cdot s_1(\mathbf{R}))$ as in Lemma 7. The message space is $\{0,1\}^\tau$. The PKE scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is as follows.

[7] We can satisfy this condition with large enough l, l' from the LTF and our ABM-LTF.

– $\mathsf{KeyGen}(1^\lambda)$ The key generation algorithm does:
  1. Run $(\mathsf{ABM.ek}, \mathsf{ABM.ik}, \mathsf{ABM.tk}) \leftarrow \mathsf{ABM.Gen}(1^\lambda)$.
  2. Run $(\mathsf{LTF.ek}, \mathsf{LTF.ik}) \leftarrow \mathsf{LTF.Gen}(1^\lambda, \mathsf{inj})$.
  3. Set the public key $\mathsf{pk} = (\mathsf{LTF.ek}, \mathsf{ABM.ek})$ and private key $\mathsf{sk} = (\mathsf{LTF.ik}, \mathsf{ABM.ik})$.
– $\mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}; r)$ To encrypt $\mathsf{m} \in \{0,1\}^\tau$, the encryption algorithm does:
  1. Randomly select $\mathbf{e}_1, \mathbf{e}_2 \xleftarrow{\$} I_\beta^m$, $\mathbf{x} \xleftarrow{\$} I_\beta^{n_1} \times I_\gamma^{n-n_1}$; Set $\mathbf{x}_1^t = [\mathbf{e}_1^t | \mathbf{x}^t]$, $\mathbf{x}_2^t = [\mathbf{e}_2^t | \mathbf{x}^t] \in \mathsf{D}$.
  2. Randomly select a universal hash function $\mathsf{UH}_\mathbf{k} \xleftarrow{\$} \mathcal{UH}$. [8]
  3. Compute $\mathbf{y}_1 = \mathsf{LTF.Eval}(\mathsf{LTF.ek}, \mathbf{x}_1)$ and $\rho = \mathsf{UH}_\mathbf{k}(\mathbf{x}, \mathbf{e}_1, \mathbf{e}_2) \oplus \mathsf{m}$.
  4. Set $\mathsf{tag} = (\mathbf{t}_\mathsf{p}, \mathbf{t}_\mathsf{a})$ for randomly sampled $\mathbf{t}_\mathsf{p} = (\mathbf{D}, R)$ and $\mathbf{t}_\mathsf{a} = (\mathsf{UH}_\mathbf{k}, \rho, \mathbf{x}_2)$, then compute $\mu = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}, \mathbf{t}_\mathsf{a}, \mathbf{y}_1); R)$.
  5. Use μ as the input of the step 2 of the algorithm $\mathsf{ABM.Eval}$, and compute the output of ABM-LTF: $\mathbf{y}_2 = \mathsf{ABM.Eval}(\mathsf{ABM.ek}, \mathsf{tag}, \mathbf{x}_2)$.
  6. Set the ciphertext $\mathsf{ct} = (\mathbf{y}_1, \mathbf{y}_2, \mathbf{t}_\mathsf{p}, \mathsf{UH}_\mathbf{k}, \rho, \mu)$.

  Note the randomness of this encryption $r = \mathsf{tag}$ where all elements in tag are public except $\mathbf{x}_2$.
– $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$ The decryption algorithm does:
  1. Parse the ciphertext as $\mathsf{ct} = (\mathbf{y}_1, \mathbf{y}_2, \mathbf{t}_\mathsf{p}, \mathsf{UH}_\mathbf{k}, \rho, \mu)$.
  2. Run $\mathsf{LTF.Inv}(\mathsf{LTF.ik}, \mathbf{y}_1)$ to get $\mathbf{x}_1^t = [\mathbf{e}_1^t | \mathbf{x}^t]$; Reject if $\|\mathbf{e}_1\| > B$.
  3. Let $\mathbf{F}$ be the matrix derived at the step 2 of $\mathsf{ABM.Inv}$. Compute $\mathbf{e}_2^t = \mathbf{y}_2^t - \mathbf{x}^t\mathbf{F}$; Reject if $\|\mathbf{e}_2\| > B$; Otherwise, go to the next step.
  4. Compute $\mu' = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}, \mathbf{t}_\mathsf{a}, \mathbf{y}_1); R)$ where $\mathbf{t}_\mathsf{a} = (\mathsf{UH}_\mathbf{k}, \rho, \mathbf{x}_2)$; if $\mu' \neq \mu$, reject; Otherwise go to the next step.
  5. Output the message $\mathsf{m} = \rho \oplus \mathsf{UH}_\mathbf{k}(\mathbf{x}, \mathbf{e}_1, \mathbf{e}_2)$.

The correctness of decryption algorithm can be easily checked.

5.3 Security Proof

Theorem 5. *Suppose that the ABM-LTF specified above is secure. Then the PKE scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ is IND-SO-CCA2 secure. In particular, for every PPT adversary $\mathcal{A}$ against Π with advantage $\mathsf{Adv}^{\mathsf{ind\text{-}so\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda)$, there exist PPT adversaries $\mathcal{B}_1$, $\mathcal{B}_2$ and $\mathcal{B}_3$ such that $\mathsf{Adv}^{\mathsf{ind\text{-}so\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda)$*

$$\leq \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{B}_4}(\lambda) + \mathsf{negl}(\lambda)$$

for the same chameleon hash function CH used in the construction of ABM-LTF, where $\mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{B}_1}(\lambda)$ is the advantage of $\mathcal{B}_1$ against CH that is used in ABM-LTF.

Proof. Recall that in the IND-SO-CCA2 security game (Figure 1), we have N challenge ciphertexts. We denote the i-th challenge ciphertext by

$$\mathsf{ct}^{(i)} = (\mathbf{y}_1^{(i)}, \mathbf{y}_2^{(i)}, \mathbf{t}_\mathsf{p}^{(i)}, \mathsf{UH}_{\mathbf{k}^{(i)}}, \rho^{(i)}, \mu^{(i)})$$

[8] Note the family of universal hash functions is used for masking the message and not the one used in the construction of ABM-LTF.

where $\mathbf{t}_{\mathsf{p}}^{(i)} = (\mathbf{D}^{(i)}, R^{(i)})$. Also recall $\mathbf{t}_{\mathsf{a}} = (\mathsf{UH}_{\mathbf{k}}, \rho, \mathbf{x}_2)$ for some $\mathbf{x}_2^t = [\mathbf{e}_2^t | \mathbf{x}^t]$. And $\mathbf{x}_2$ is applied to $\mathsf{ABM.Eval}$ with $\mathsf{tag} = (\mathbf{t}_{\mathsf{p}}, \mathbf{t}_{\mathsf{a}}, \mu)$ to generate $\mathbf{y}_2$.

We prove the theorem through a game sequence. Let S_i be the event that $\mathcal{A}$ outputs 1 in **Game** i. The first game **Game** 1 is the same as the experiment $\mathsf{Exp}^{\mathsf{ind\text{-}so\text{-}cca\text{-}b}}_{\Pi,\mathcal{A}}(\lambda)$. By definition we have

$$|\Pr[S_1] - 1/2| = \mathsf{Adv}^{\mathsf{ind\text{-}so\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda).$$

In **Game** 2, we reject all the decryption queries in which the component μ has already appeared in one of the challenge ciphertexts. If the adversary makes a decryption query on ciphertext $\mathsf{ct} = (\mathbf{y}_1, \mathbf{y}_2, \mathbf{t}_{\mathsf{p}} = (\mathbf{D}, R), \mathsf{UH}_{\mathbf{k}}, \rho, \mu^{(i)})$ where $\mu^{(i)}$ is from some $\mathsf{ct}^{(i)} = ({\mathbf{y}_1}^{(i)}, \mathbf{y}_2^{(i)}, \mathbf{t}_{\mathsf{p}}^{(i)}, \mathsf{UH}_{\mathbf{k}^{(i)}}, \rho^{(i)}, \mu^{(i)})$, we argue that such query will be rejected unless the collision resistant property of the chameleon hash function is broken. Notice that R is the randomness, $\mathbf{y}_2$ is the only ciphertext component that is *not* a part of the message of the chameleon hash function. Let $\mathbf{t}_{\mathsf{a}} = (\mathsf{UH}_{\mathbf{k}}, \rho, \mathbf{x}_2)$ and $\mathbf{t}_{\mathsf{a}}^{(i)} = (\mathsf{UH}_{\mathbf{k}^{(i)}}, \rho^{(i)}, \mathbf{x}_2^{(i)})$. There are three cases:

- If $\mathbf{y}_2 = \mathbf{y}_2^{(i)}$ and$(\mathbf{t}_{\mathsf{p}}, \mathsf{UH}, \rho) = (\mathbf{t}_{\mathsf{p}}^{(i)}, \mathsf{UH}_{\mathbf{k}^{(i)}}, \rho^{(i)})$: In this case the query is exactly the i-th challenge ciphertext which is invalid.
- If $\mathbf{y}_2 = \mathbf{y}_2^{(i)}$ and $(\mathbf{t}_{\mathsf{p}}, \mathsf{UH}, \rho) \neq (\mathbf{t}_{\mathsf{p}}^{(i)}, \mathsf{UH}_{\mathbf{k}^{(i)}}, \rho^{(i)})$: The decryption algorithm will output $\mathbf{x}_2$ in the step 3 (when the ciphertext passes through all test up to step 3) and recompute μ'. We would have $\mu \neq \mu$, thus reject the query, unless $\mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}, \mathbf{t}_{\mathsf{a}}, \mathbf{y}_1); R) = \mathsf{CH.Eval}(\mathsf{Hk}, (\mathbf{D}^{(i)}, \mathbf{t}_{\mathsf{a}}^{(i)}, \mathbf{y}_1^{(i)}); R^{(i)})$, which corresponds to a collision to the chameleon hash function.
- If $\mathbf{y}_2 \neq \mathbf{y}_2^{(i)}$: Recall that $\mu = \mu^{(i)}$ is derived from an injective tag. If the query makes decryption algorithm output $\mathbf{x}_2$ at step 3, we must have $\mathbf{x}_2 \neq \mathbf{x}_2^{(i)}$ and, thus, $\mathbf{t}_{\mathsf{a}} \neq \mathbf{t}_{\mathsf{a}}^{(i)}$. Then the query will be reject at step 4 unless an explicit collision, $((\mathbf{D}, \mathbf{t}_{\mathsf{a}}, \mathbf{y}_1); R)$ and $(\mathbf{D}^{(i)}, \mathbf{t}_{\mathsf{a}}^{(i)}, \mathbf{y}_1^{(i)}); R^{(i)})$, happens to the chameleon hash function.

So **Game** 2 and **Game** 1 behave the same unless the collision resistancy of the chameleon hashing is broken. Thus we have

$$|\Pr[S_2] - \Pr[S_1]| \leq \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{B}_1}(\lambda)$$

for some suitable adversary $\mathcal{B}_1$.

In **Game** 3, lossy tags are generated using $\mathsf{ABM.LTag}$ for all challenge ciphertexts, i.e., $\mathsf{ct}^{(i)}$ for $i \in [N]$. Notice that here we allow the decryption queries made with lossy tags in which $\mu \neq \mu^{(i)}$. (Of course it is computationally hard to come up with such queries by the evasiveness of ABM-LTF, which we have not used yet.) This is because the decryption algorithm in **Game** 3 does not use ABM-LTF to invert to get $\mathbf{x}$. Instead, $\mathbf{x}$ is recovered by LTF from $\mathbf{y}_1$ and then $\mathbf{e}_2$ can be uniquely recovered from $\mathbf{x}$ and $\mathbf{y}_2$. By tag indistinguishability of the $\mathsf{ABM\text{-}LTF}$,

$$|\Pr[S_3] - \Pr[S_2]| \leq \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_2}(\lambda)$$

for some suitable adversary $\mathcal{B}_2$.

Recall that in **Game** 3, we use LTF to invert $\mathbf{y}_1$ to get $\mathbf{x}_1^t = [\mathbf{e}_1^t|\mathbf{x}^t]$ and use $\mathbf{y}_2$ and $\mathbf{x}$ to recover $\mathbf{e}_2$ and, thus $\mathbf{x}_2$. In **Game** 4, we directly use $\mathsf{ABM.ik}$ to invert $\mathbf{y}_2$ and get $\mathbf{x}_2$. By our correctness of LTF and $\mathsf{ABM\text{-}LTF}$, this gives the same result unless μ in the decryption query is from one of the challenge ciphertexts, or the queries are made with lossy or invalid tags. The first case is already excluded in **Game** 3. The latter case would not happen under the evasiveness of $\mathsf{ABM\text{-}LTF}$. So we have

$$|\Pr[S_4] - \Pr[S_3]| \leq \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_3}(\lambda)$$

for some suitable adversary $\mathcal{B}_3$.

In **Game** 5, we generate a lossy evaluation key for LTF. We have

$$|\Pr[S_5] - \Pr[S_4]| \leq \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{B}_4}(\lambda)$$

for some suitable adversary $\mathcal{B}_4$.

In **Game** 6, we produce the ρ component in each challenge ciphertext by randomly sampling a string $\mathtt{r} \xleftarrow{\$} \{0,1\}^\tau$ and setting $\rho = \mathtt{r} \oplus \mathsf{m}$. As in **Game** 5, the $\mathbf{y}_2$ components are computed from $\mathsf{ABM\text{-}LTF}$ with lossy tags on $\mathbf{x}_2 \in \mathsf{D}$ for all challenge ciphertexts. Let $|\mathsf{E}_2|$ and $|\mathsf{X}|$ be the number of possible values of $\mathbf{e}_2$ and $\mathbf{x}$ respectively [9]. Recall $\mathbf{x}_1^t = [\mathbf{e}_1^t|\mathbf{x}^t]$ and $\mathbf{x}_2^t = [\mathbf{e}_2^t|\mathbf{x}^t]$. By the parameter selection and Lemma 2, we have

$$\begin{aligned}
\tilde{H}_\infty(\mathbf{x}_1, \mathbf{x}_2|\mathbf{y}_1, \mathbf{y}_2, \mu) &= \tilde{H}_\infty(\mathbf{x}, \mathbf{e}_1, \mathbf{e}_2|\mathbf{y}_1, \mathbf{y}_2, \mu) \\
&\geq H_\infty(\mathbf{x}, \mathbf{e}_1, \mathbf{e}_2) - (\log|\mathsf{D}| - l) - (\log|\mathsf{D}| - l') - 2\lambda \\
&\geq \log|\mathsf{D}| + \log|\mathsf{E}_2| - (\log|\mathsf{D}| - l) - (\log|\mathsf{X}| + \log|\mathsf{E}_2| - l') - 2\lambda \\
&= l + l' - \log|\mathsf{X}| - 2\lambda
\end{aligned}$$

Consequently, by the hypothesis that $\tau \leq (l - 2\lambda) - 2\log(1/\epsilon)$ and Lemma 3,

$$\Delta\left((\mathbf{y}_1, \mathbf{y}_2, \mu, \mathsf{UH}_{\mathbf{k}}, \mathsf{UH}_{\mathbf{k}}(\mathbf{x})), (\mathbf{y}_1, \mathbf{y}_2, \mu, \mathsf{UH}_{\mathbf{k}}, \mathcal{U}_\tau)\right) \leq \epsilon = \mathsf{negl}(\lambda)$$

where $\mathcal{U}_\tau$ stands for the uniform distribution over $\{0,1\}^\tau$. So we get

$$|\Pr[S_6] - \Pr[S_5]| \leq \mathsf{negl}(\lambda)$$

In **Game** 6, as all challenge messages are masked by an one-time pad, $\mathcal{A}$ gets no information about them. The original message vector $\mathbf{m}_0$ and the conditionally resampled message vector $\mathbf{m}_1$ come from the same distribution, thus

$$\Pr[S_6] = 1/2$$

Summing up, we obtain that $\mathsf{Adv}^{\mathsf{ind\text{-}so\text{-}cca}}_{\Pi,\mathcal{A}}(\lambda)$

$$\leq \mathsf{Adv}^{\mathsf{coll}}_{\mathsf{CH},\mathcal{B}_1}(\lambda) + \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_2}(\lambda) + \mathsf{Adv}^{\mathsf{eva}}_{\mathsf{ABM\text{-}LTF},\mathcal{B}_3}(\lambda) + \mathsf{Adv}^{\mathsf{ind}}_{\mathsf{LTF},\mathcal{B}_4}(\lambda) + \mathsf{negl}(\lambda)$$

which completes the proof. □

[9] Recall that $\mathbf{x}$, $\mathbf{e}_1$, $\mathbf{e}_2$ are chosen uniformly at random from certain intervals.

5.4 Tightly Secure IND-CCA2 PKE

The above PKE scheme is also a tightly secure PKE scheme with respect to the multi-ciphertext IND-CCA2 definition adopted by Gay et al. [24] (Definition 6). One can easily modify the IND-SO-CCA2 security proofs into a tight security proof with respect to the IND-CCA2 definition, where the security loss is independent of the number of decryption queries and the number of encryption queries.

Particularly, such a reduction is able to answer all the decryption queries and construct all challenge ciphertexts with lossy tags simultaneously, making the challenge ciphertexts information-theoretically unrecoverable. This IND-CCA2 secure PKE scheme we just outlined is thus the first tightly secure PKE scheme in the multi-ciphertext IND-CCA2 security model based on the LWE assumptions (or more generally without using quantumly broken assumptions).

Definition 6 (Multi-ciphertext IND-CCA2 security). *A PKE scheme* $\Pi = (\mathsf{KeyGen}, \mathsf{Encrypt}, dec)$ *is IND-CCA2 secure in the multi-ciphertext setting if for every PPT adversary* $\mathcal{A}$*, we have* $\mathcal{A}$*'s advantage*

$$\mathsf{Adv}^{\mathsf{ind\text{-}cc2a}}_{\Pi,\mathcal{A}}(\lambda) = \left|\Pr\left[\mathsf{Exp}^{\mathsf{ind\text{-}cca2}}_{\Pi,\mathcal{A}}(\lambda) = 1\right] - 1/2\right|$$

is negligible in λ *where the experiment* $\mathsf{Exp}^{\mathsf{ind\text{-}cca2}}_{\Pi,\mathcal{A}}(\lambda)$ *is defined in Fig. 2.*

Experiment $\mathsf{Exp}^{\mathsf{ind\text{-}cca2}}_{\Pi,\mathcal{A}}(\lambda)$	$\mathcal{O}_{\mathsf{enc}}(\mathsf{m}_0, \mathsf{m}_1)$	$\mathcal{O}_{\mathsf{dec}}(\mathsf{ct})$
1. $\mathcal{L} \leftarrow \emptyset$ 2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}((1^\lambda)$ 3. $b \xleftarrow{\$} \{0,1\}$ 4. $b' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathsf{enc}}(\cdot,\cdot),\mathcal{O}_{\mathsf{dec}}(\cdot)}(\mathsf{pk})$ 5. return 1 if $b' = b$, 0 otherwise	1. If $\lvert\mathsf{m}_0\rvert \neq \lvert\mathsf{m}_1\rvert$ return $\perp$ 2. $r \xleftarrow{\$} \mathcal{R}_\Pi$ 3. $\mathsf{ct} \leftarrow \mathsf{Encrypt}(\mathsf{pk}, \mathsf{m}_b; r)$ 4. $\mathcal{L} \leftarrow \mathcal{L} \cup \mathsf{ct}$	1. If $\mathsf{ct} \in \mathcal{L}$, return $\perp$ 2. return $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$

Fig. 2: Security experiment of IND-CCA2 security

6 Conclusion

In this paper, we have proposed the first All-But-Many Lossy Trapdoor Function based on lattice assumptions. ABM-LTFs are a very powerful primitive with potentially many applications in the construction of multi-challenge or multi-user cryptosystems. Our result answers the two open questions of constructing, from lattices, ABM-TF (originally posed by Alperin-Sheriff and Peikert [5]) and ABM-LTF (posed by Hofheinz [29]).

In addition, we have constructed an IND-SO-CCA2-secure PKE scheme from lattices by taking our ABM-LTF along the path of [27,29]. Our PKE scheme

enjoys a tight security reduction, in the sense that the reduction is independent of all adversarial queries, including decryption, opening, and challenge ciphertexts. This gives the first tightly IND-CCA2 secure PKE scheme from LWE, and an alternative solution, lattice-based, to the problem of constructing tightly secure CCA PKE without bilinear or multilinear parings [24].

Acknowledgement

We thank Benoît Libert and Damien Stehlé and the anonymous reviewers for useful comments.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. EUROCRYPT 2010, Springer (2010)
2. Ajtai, M.: Generating hard instances of lattice problems (extended abstract). STOC 1996, ACM (1996)
3. Akavia, A., Bogdanov, A., Guo, S., Kamath, A., Rosen, A.: Candidate weak pseudorandom functions in AC0∘MOD2. ITCS 2014, ACM (2014)
4. Alperin-Sheriff, J.: Short signatures with short public keys from homomorphic trapdoor functions. PKC 2015, Springer (2015)
5. Alperin-Sheriff, J., Peikert, C.: Circular and kdm security for identity-based encryption. PKC 2012, Springer (2012)
6. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. CRYPTO 2009,
7. Banerjee, A., Peikert, C., Rosen, A.: Pseudorandom functions and lattices. EUROCRYPT 2012, Springer (2012)
8. Bellare, M., Hofheinz, D., Yilek, S.: Possibility and impossibility results for encryption and commitment secure under selective opening. In: EUROCRYPT 2009.
9. Bellare, M., Kiltz, E., Peikert, C., Waters, B.: Identity-based (lossy) trapdoor functions and applications. Cryptology ePrint Archive, Report 2011/479 (2011)
10. Bellare, M., Kiltz, E., Peikert, C., Waters, B.: Identity-based (lossy) trapdoor functions and applications. EUROCRYPT 2012, Springer (2012)
11. Böhl, F., Hofheinz, D., Kraschewski, D.: On definitions of selective opening security. PKC 2012, Springer (2012)
12. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully key-homomorphic encryption, arithmetic circuit abe and compact garbled circuits. EUROCRYPT 2014, Springer (2014)
13. Boyen, X.: Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. PKC 2010, Springer (2010)
14. Boyen, X., Li, Q.: Towards tightly secure lattice short signature and Id-based encryption. ASIACRYPT 2016, Springer (2016)
15. Brakerski, Z., Vaikuntanathan, V.: Efficient fully homomorphic encryption from (standard) LWE. FOCS 2011, IEEE (2011)
16. Brakerski, Z., Vaikuntanathan, V.: Lattice-based FHE as secure as PKE. ITCS 2014, ACM (2014)
17. Brent, R.P., McKay, B.D.: Determinants and ranks of random matrices over zm. Discrete Mathematics 66(1), 35 – 49 (1987)

18. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai trees, or how to delegate a lattice basis. Journal of Cryptology 25(4), 601–639 (2012)
19. Damgård, I., Jurik, M.: A generalisation, a simpli. cation and some applications of paillier's probabilistic public-key system. PKC 2001, Springer (2001)
20. Dodis, Y., Kiltz, E., Pietrzak, K., Wichs, D.: Message authentication, revisited. EUROCRYPT 2012, Springer (2012)
21. Döttling, N., Müller-Quade, J.: Lossy codes and a new variant of the learning-with-errors problem. EUROCRYPT 2013, Springer (2013)
22. Fehr, S., Hofheinz, D., Kiltz, E., Wee, H.: Encryption schemes secure against chosen-ciphertext selective opening attacks. EUROCRYPT 2010, Springer (2010)
23. Fujisaki, E.: All-but-many encryption. ASIACRYPT 2014, Springer (2014)
24. Gay, R., Hofheinz, D., Kiltz, E., Wee, H.: Tightly cca-secure encryption without pairings. EUROCRYPT 2016, Springer (2016)
25. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. STOC 2008, ACM (2008)
26. Gentry, C., Sahai, A., Waters, B.: Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. CRYPTO 2013,
27. Hemenway, B., Libert, B., Ostrovsky, R., Vergnaud, D.: Lossy encryption: Constructions from general assumptions and efficient selective opening chosen ciphertext security. ASIACRYPT 2011, Springer (2011)
28. Hiromasa, R., Abe, M., Okamoto, T.: Packing messages and optimizing bootstrapping in GSW-FHE. PKC 2015, Springer (2015)
29. Hofheinz, D.: All-but-many lossy trapdoor functions. EUROCRYPT 2012, Springer (2012)
30. Hofheinz, D.: Circular chosen-ciphertext security with compact ciphertexts. In: EUROCRYPT 2013, Springer (2013)
31. Hofheinz, D., Rupp, A.: Standard versus selective opening security: separation and equivalence results. TCC 2014, Springer (2014)
32. Huang, Z., Liu, S., Qin, B.: Sender-equivocable encryption schemes secure against chosen-ciphertext attacks revisited. PKC 2013, Springer (2013)
33. Lyubashevsky, V., Masny, D.: Man-in-the-middle secure authentication schemes from LPN and weak PRFs. CRYPTO 2013, Springer (2013)
34. Libert, B., Sakzad, A., Stehl, D., Steinfeld, R.: All-But-Many Lossy Trapdoor Functions and Selective Opening Chosen-Ciphertext Security from LWE. CRYPTO 2017, Springer (2017)
35. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. EUROCRYPT 2012, Springer (2012)
36. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. CRYPTO 2013, Springer (2013)
37. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. EUROCRYPT 1999, Springer (1999)
38. Peikert, C., Vaikuntanathan, V., Waters, B.: A framework for efficient and composable oblivious transfer. CRYPTO 2008, Springer (2008)
39. Peikert, C., Waters, B.: Lossy trapdoor functions and their applications. SIAM Journal on Computing 40(6), 1803–1844 (2011)
40. Qin, B., Liu, S.: Leakage-resilient chosen-ciphertext secure public-key encryption from hash proof system and one-time lossy filter. ASIACRYPT 2013,
41. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. STOC 2005, ACM (2005)
42. Zhang, R.: Tweaking tbe/ibe to pke transforms with chameleon hash functions. ACNS 2007, Springer (2007)