Revocable cryptosystems from lattices

Zhang, Juanyang

2018

Zhang, J. (2018). Revocable cryptosystems from lattices. Doctoral thesis, Nanyang Technological University, Singapore.

<http://hdl.handle.net/10356/75927>

<https://doi.org/10.32657/10356/75927>

REVOCABLE CRYPTOSYSTEMS FROM LATTICES

JUANYANG ZHANG

School of Physical and Mathematical Sciences

2018

REVOCABLE CRYPTOSYSTEMS FROM LATTICES

JUANYANG ZHANG

School of Physical and Mathematical Sciences

A thesis submitted to the Nanyang Technological University
in partial fulfillment of the requirement for the degree of
Doctor of Philosophy

2018

To My Mommy and Daddy.

Acknowledgements

First of all, I want to express my deepest gratitude to my main supervisor, Prof. Huaxiong Wang, for his priceless guidance and support. I am honoured to accomplish my PhD studies under his supervision. He is a model for me, both in academia and in daily life.

I also want to sincerely thank my co-supervisor Prof. San Ling for his invaluable guidance and kindness.

I would like specially to give a deep thank to Dr. Khoa Ta Toan Nguyen for his helpful suggestions, his instructive comments, and his patient explanations. I deeply admire his enthusiastic attitude towards research and am profoundly grateful for his kindness. It is a great privilege for me to be working with him.

I also want to thank our cryptographic group members: Dr. Hyung Tae Lee, Dr. Frederic Ezerman Martianus, Dr. Fuchun Lin, Tan Hong Meng Benjamin, Yanhong Xu, Neng Zeng, and some visiting students and professors. I really learned a lot from the talks they gave and from the discussions with them.

I would like to thank my friends: Shantian Cheng, Shu Liu, Ivan Tjuawinata and Chen Yuan.

Last but not least, my deepest love goes to my family, including my father Zheng Zhang, mother Zhecui Nan, eldest brother Haoyang Zhang and second brother Haoli Zhang. Although they totally could not image PhD students' life, they still accompanied me throughout my studies, especially with their comfort and encouragement when I was frustrated.

List of Works

Below is the list of works done during my PhD studies in Nanyang Technological University, in chronological order.

1. Shantian Cheng and Juanyang Zhang. Adaptive-ID secure revocable identity-based encryption from lattices via subset difference method. In *ISPEC 2015*, volume 9065 of *Lecture Notes in Computer Science*, pages 283–297. Springer, 2015
2. Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Server-aided revocable identity-based encryption from lattices. In *CANS 2016*, volume 10052 of *Lecture Notes in Computer Science*, pages 107–123. Springer, 2016
3. San Ling, Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Revocable predicate encryption from lattices. In *ProvSec 2017*, volume 10592 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2017
4. San Ling, Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Server-aided revocable predicate encryption: definition and lattice-based instantiation. In *Submission to Designs, Codes and Cryptography*

Contents

1	Introduction	1
1.1	Revocable Cryptosystems	1
1.2	Lattice-based Cryptography	6
1.3	Contributions of the Thesis	8
1.4	Organization of the Thesis	14
2	Definitions and Preliminaries	16
2.1	General Notations	16
2.2	Preliminaries on Lattices	17
2.2.1	Lattices	17
2.2.2	Gaussian Distributions	19
2.2.3	The LWE Problem	20
2.3	Basic Cryptographic Primitives from Lattices	23
2.3.1	Sampling Algorithms	23
2.3.2	Public-key Encryption	25
2.3.3	Hierarchical Identity-based Encryption	28
2.3.4	Predicate Encryption	33
2.4	Revocable Cryptosystems	39
2.4.1	The Complete Subtree Method	39
2.4.2	Revocable Identity-based Encryption	41

3	Server-aided Revocable Identity-based Encryption from Lattices	47
3.1	Introduction	47
3.2	Definitions	51
3.3	A Lattice-based SR-IBE Scheme	55
3.3.1	Description of the Scheme	55
3.3.2	Correctness and Efficiency	59
3.3.3	Security	60
4	Server-aided Revocable Predicate Encryption	69
4.1	Introduction	69
4.2	The model of SR-PE	73
4.3	A Lattices-based SR-PE Scheme	78
4.3.1	Description of the Scheme	78
4.3.2	Correctness and Efficiency	82
4.3.3	Security	84
5	Revocable Predicate Encryption from Lattices	93
5.1	Introduction	93
5.2	Definitions	97
5.3	A Lattice-based RPE Scheme	99
5.3.1	Description of the Scheme	99
5.3.2	Correctness, Efficiency and Potential Implementation	102
5.3.3	Security	105
5.3.4	Extensions	113
6	Conclusions and Open Problems	115
	Bibliography	119

Abstract

In the last decade, lattices have become one of the most powerful tools in constructing cryptographic schemes, which enjoy conjectured resistance against quantum computers and strong security guarantees from worst-case to average-case reductions, as well as asymptotic efficiency. For a multi-user cryptosystem, user revocation has been a necessary but challenging problem. However, all known revocable schemes are either based on number-theoretic assumptions or lattice-based but less efficient compared to the art-of-date systems.

In this thesis, we focus on investigating user revocation model and the associated lattice-based instantiations. Our constructions have two goals: (i) to improve the existing revocable lattice-based cryptosystems in terms of efficiency and security; (ii) to consider the revocation functionality in new contexts from lattices. For the former, we carefully adapt the very recent revocation model into the lattice setting. The latter can be achieved either by using the existing revocation models (without concrete constructions from lattices) or by proposing new revocation models. We construct a series of cryptosystems supporting efficient revocations as follows.

- A revocable identity-based encryption (IBE) scheme, which is more efficient than all existing such schemes from lattices. We follow the architecture of the server-aided revocable encryptions, proposed by Qin et al. (ESORICS 2015). This paradigm provides significant efficiency advantages over previous revocation techniques in the setting of IBE. In the server-aided revocation model, most of the workloads on the user side are outsourced to an untrusted server, which can be untrusted since it does not possess any private information. With the help of this server, non-revoked users do not need to update anything when the system revokes other users. We equip Agrawal, Boneh, and Boyen's IBE (EUROCRYPT 2010) with the server-aided

revocation method. In the technical view, we observe that a “double encryption” mechanism is well-suited in such a server-aided system. We also show that our scheme is provably secure provided with the strong hardness of the Learning With Errors (LWE) problem.

- A revocation model called server-aided revocable predicate encryption (SR-PE) and an instantiation from lattices. We consider the server-aided revocation mechanism in the predicate encryption (PE) setting and formalize the notion of SR-PE with rigorous definitions and security model. Moreover, we introduce a construction of SR-PE for the scheme introduced by Agrawal, Freeman, and Vaikuntanathan (ASIACRYPT 2011) and prove that our scheme is selectively secure in the standard model. The correctness of our scheme relies on a special property of lattice-based encryption schemes.
- A lattice-based construction of predicate encryption following the direct revocation mechanism. In such a mechanism, it forces the ciphertexts to carry on the revocation information. Nieto, Manulis, and Sun (ACISP 2012) considered direct revocations in the PE setting and suggested the notion of full-hiding security for revocable PE schemes, which demands that the encrypted data keeps the privacy of not only the plaintext and the associated attribute, but also the revocation information. Following their pairing-based construction, we introduce a corresponding instantiation from lattice assumptions. Regarding efficiency, our lattice-based scheme is somewhat comparable to the construction by Nieto, Manulis, and Sun. Our scheme achieves the full-hiding security thanks to the anonymity of one IBE instance we additionally introduce into the system.

Chapter 1

Introduction

1.1 Revocable Cryptosystems

Cryptography has been playing a crucial role throughout the history of information theory. In the modern age, cryptography is effectively synonymous with encryption, where a sender encrypts the plaintext and a recipient decrypts the ciphertext. One important area of cryptography is the symmetric encryption, which requires the sender and the recipient agree on a secret key in advance. In this thesis, we focus on another field of cryptography, known as asymmetric encryption, where the two parties do not need to share the same key.

Public-Key Encryption. Asymmetric encryption, or public-key encryption (PKE), is a fundamental and revolutionary ingredient in cryptography, which is used as a mechanism to guarantee the authenticity, confidentiality and non-repudiability of data storage and electronic communications. The idea of PKE was discovered by Diffie and Hellman in [32]. In a PKE scheme, there are pairs of keys: public keys and private keys. Whereas the former can be distributed publicly, the latter are preserved by the owner secretly. Anyone is able to encrypt a message with the intended recipient's public key and the ciphertext can only be decrypted using the corresponding private key. One drawback of PKE is the need for a public key infrastructure (PKI) to manage

users' public keys.

In 1978, Rivest, Shamir, and Adleman provided the first implementation of PKE in [91] (known as the RSA system), based on the practical difficulty of the factorization of the product of two large prime numbers, the factoring problem. Since the 1970s, a number of PKE constructions and other relevant techniques have been developed in the area of public-key cryptography. ElGamal invented the ElGamal public-key cryptosystem [37] relying on the discrete logarithm problem. The introduction of pairing-based cryptography, initiated by Boneh [16] and Menezes et al. [68], has yielded many PKE systems, which are based on algebraic structure of elliptic curves over finite fields. Moreover, pioneered by Ajtai [6], Regev [90], and Gentry et al. [41], lattices have been playing an active role in public-key cryptography, which will be discussed later.

Recently, PKE has come across an archetype shift from traditional encryption for particular users to more flexible encryption systems, which offer fine-grained and role-based access to the ciphertext. These systems often rely on some functional relationships, which can be evaluated during the decryption phase, taking as input the information associated with the receiver's private key and the information under the encrypted data. Based on difference of the functionality, PKE encompasses concepts such as identity-based encryption, attribute-based encryption, hidden vector encryption, predicate encryption, functional encryption, etc.

Identity-Based Encryption. Introduced by Shamir [98], identity-based encryption (IBE) provides an important alternative way to avoid the need for PKI. It allows a sender to use the recipient's identity as a public key to encrypt a message, from which the corresponding private key is issued through a secret channel by a trusted authority called the key generation center (KGC). This implies that IBE is a single-receiver system since one ciphertext can only be decrypted by the intended recipient. The first realization of IBE was the work by Boneh and Franklin [18] based on the Bilinear Diffie-Hellman problem. Almost at the same time, Cocks [30] proposed a scheme using

quadratic residues modulo a composite. Moreover, the third class of IBE, pioneered by Gentry et al. [41] in 2008, is based on lattice assumptions.

Predicate Encryption. Formalized by Katz, Sahai, and Waters [51], predicate encryption (PE) gives fine-grained access control beyond conventional PKE. In a PE system, the recipient’s private key, issued by the KGC, is associated with a predicate f , while a ciphertext is bound to an attribute a . The success of decryption holds if and only if $f(a) = 1$, which follows that PE is a multi-receiver system since one ciphertext may be decrypted by many users. PE schemes can be regarded as a generalization of attribute-based encryption (ABE) [49, 93], which additionally keep privacy of the attribute bound to each ciphertext. Concrete proposals of PE systems usually focus on realizing special types of predicates. One typical category is for the inner-product predicates, where attribute a is represented by vector $\vec{y}$ and predicate f is determined by vector $\vec{x}$ such that $f(a) = 1$ if and only if $\langle \vec{x}, \vec{y} \rangle = 0$ ($\langle \vec{x}, \vec{y} \rangle$ denotes the inner product of vectors $\vec{x}$ and $\vec{y}$ over a ring or a field). In [51], it has been demonstrated that inner-product predicates can be used to evaluate a wide class of predicates such as equalities, conjunctions of comparison or subset tests, hidden vector predicate, polynomial evaluation, and arbitrary CNF/DNF formulae. In the scope of PE for inner-product predicates, Katz, Sahai, and Waters [51] proposed a fully attribute hiding scheme based on the composite-order pairings, and Agrawal et al. [4] constructed a weakly attribute hiding system based on lattices. Moreover, PE for circuits was proposed by Gorbunov et al. [46] using lattice assumptions.

Revocation Problem. As multi-user cryptosystems, equipping PKE schemes with efficient revocation mechanisms is necessary. We now illustrate with examples that there is a need to revoke users or keys in certain scenarios.

- **E-mail Systems.** Let us start with a classical example described in the work by Shamir [98] that initiates the study of IBE. When a user, say Alice, joins an e-mail system, she can obtain a unique e-mail address, such as “alice@university.sg”, together with a private key which is secretly kept by herself. Anyone in the

system is able to send emails to Alice, which are ciphertexts bound to the string “alice@university.sg”. Alice then can read the emails specified by her identity using her private key.

However, in some cases (e.g., after graduation) Alice’s ability of reading e-mails should be revoked from the system. Since Alice’s email address is distinct from others’, revoking the string “alice@university.sg” will only affect her decryption capability. This implies that there should be a mechanism for the service manager to update the e-mail system to revoke Alice’s email address. After updating, although Alice still holds the correct private key, she cannot access to any emails.

- **Credit Card Systems.** Next, we consider the credit card investigation problem given in the paper by Boneh and Waters [20] as an example to motivate PE. In a credit card system, the detailed information for a huge number of transactions is encrypted under a set of attributes (e.g., date, venue and costs). When an investigator, say Bob, needs to flag all transactions satisfying a certain predicate, such as $f = \text{“date is between 01/01/2018 and 12/31/2018”} \wedge \text{“venue is in Singapore”} \vee \text{“cost is over S$1000”}$, the system generates a token associated with f and the token is sent to Bob via a secure channel. Bob then can use this token to identify all the transactions satisfying f . Meanwhile, it is required that Bob should not learn anything else about these transactions (e.g., the corresponding date, venue and costs).

However, for both security and privacy concerns, Bob’s token should not be valid for a long time period. To this end, there should be a mechanism to revoke the decryption capacity of the token sent to Bob.

One notable difference from the previous example is that, here we consider revocation of Bob’s token (or key) instead of revocation of the predicate f . We feel this setting is more practicable because of two main reasons. First, the revocation problem is often motivated by the privacy challenge caused by leaking

Bob's token to some attacker. Second, the same predicate f would be queried by many investigators. If the system revokes Bob's decryption capability by revoking the associated f , this operation would affect other costumes who wants to investigate on f .

Accompany with the development of public-key cryptography, revocation has always been one of the most considerable functionalities. In the community, researchers care about both efficiency and security of revoking certain users or keys. Trivial solutions like resetting the whole system is inefficient since the system often consists of many users. The efficiency requirements in applications could be different. In the above e-mail system, users mainly care about the ciphertext size since the number of e-mails for each user might be huge, while in the credit card system, costumers prefer short keys for convenience's sake. To suit in different scenarios, many revocation paradigms have been considered: certificate-based revocation models for PKE schemes with PKI ([5,34,48,69,70],...), time-based revocation models with constant-size ciphertexts ([15,18,28,54,92],...), direct revocation models with constant-size private keys ([11,35,57,76],...), mediator-aided revocation models ([17,52,53,67,104],...), etc.

The problem of revocation can be combined with the capability of traitor tracing, which means that if a group of users combine their secret keys to produce a "pirate decoder", the KGC can trace at least one of the "traitors" given access to this decoder. In this thesis, we focus on revocable encryption schemes without traitor tracing.

A revocable cryptosystem usually builds on one specific original scheme (i.e., without revocation functionality), the security of which relies on hardness of some computational problems, like discrete logarithm and integer factorization. It is well-known that number-theoretic assumptions will be vulnerable on a sufficiently powerful quantum computer running Shor's algorithm [99]. In the area of post-quantum cryptography, which refers to the cryptographic algorithms (usually public-key algorithms) that are thought to be secure against attacks by both classical and quantum adversaries, lattices have become one of the most powerful tools.

1.2 Lattice-based Cryptography

Lattices, or full-rank additive subgroups over $\mathbb{R}^n$, have been studied since the 18th century. The computational aspect of lattices has attracted special attention over the last three decades along with the speedy development of computer science, and particularly, public-key cryptography. Since Ajtai [6] gave a breakthrough work in 1996, an excited research area known as *lattice-based cryptography*, has been developed rapidly in the last decade. The main goal is to design new cryptosystems from lattice assumptions and to improve the existing constructions in terms of efficiency and security. The security of most of these schemes relies on the conjectured hardness of computational problems on lattices.

The most basic lattice problem is the shortest vector problem (SVP), where the goal is to find the shortest non-zero vector in a given n -dimensional lattice. Cryptographers have also considered two variants of the SVP problem: the approximation version and the decision version. For the former, the goal is to find a lattice vector with length bounded by γ times the length of the shortest non-zero vector and the goal of the latter is to determine the length of the shortest non-zero vector is bounded by one or larger than γ . In both cases, $\gamma = \gamma(n)$ is an approximation factor which should be larger than one. The most widely studied and celebrated algorithm for lattice problems is the LLL algorithm, developed in 1982 by Lenstra, Lenstra, and Lovász [55]. They provided a polynomial time algorithm to solve SVP_γ (as well as other basic computational problems on lattices) with $\gamma = 2^{O(n)}$. Despite of the exponential approximation factor, the LLL algorithm somehow turns out to be amazingly useful in many applications, such as factoring polynomials over the rational numbers [55] as well as cryptosystems based on the knapsack problem [80]. Triggered by Lenstra, Lenstra, and Lovász, a lot of studies (e.g., [36, 95]) have been done trying to find better solutions for lattice problems.

Unfortunately, all these algorithms running in polynomial time only achieve ex-

ponential or sub-exponential approximation factors, whereas for polynomial factors, the attempted algorithms (e.g., [8, 55]) all run in exponential time. Furthermore, *quantum* algorithms for these problems have been considered, since Shor’s polynomial time algorithms [99] for integer factorization and discrete logarithms on a quantum computer. However, there has been little success so far. Therefore, it is reasonable and natural to have the following conjecture.

“There is no polynomial time (classical or quantum) algorithm that approximates lattice problems to within polynomial factors.”

This result justifies the potential use of lattice problems for post-quantum cryptography.

Nevertheless, the way to use lattices in cryptography is a big challenge until Ajtai [6] discovered a ground-breaking worst-case to average-case connection in 1996. On one hand, a problem is considered hard in the worst case if it is so for any instance of the problem. On the other hand, lattice-based cryptography is based on the the computational hardness of solving a random instance of some hard problems.

In the area of PKE, Regev [90] introduced an average-case problem, namely the Learning With Errors (LWE) problem, the decisional variant of which has served as the foundation for secure lattice based encryption systems. For given integers n, q, m and a noise distribution χ , the decision-version of LWE, is stated as follows: given a uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m$, where the vector $\mathbf{s}$ is uniformly random in $\mathbb{Z}_q^n$ and the noise $\mathbf{e}$ is sampled from χ^m , the goal is to distinguish between the pair $(\mathbf{A}, \mathbf{b})$ and a uniformly random pair over $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$. In [90], Regev also showed that the decision-LWE is at least as hard as quantum approximating SVP (and the shortest independent vectors problem) to within approximation factor $\gamma = \text{poly}(n)$. Later, Peikert [85] and Brakerski et al. [22] proved that under corresponding classical (non-quantum) reductions, the hardness of LWE could be built based on the hardness of the decision-variant of SVP up to polynomial approximation factors. In other words, the LWE problem is hard based on

the conjectured hardness of the worst-case lattice problems.

After Regev’s introduction, the LWE problem quickly got lots of attention from cryptographers and has become the basis of many lattice-based encryption schemes, such as PKE schemes that are secure against chosen plaintext attacks [41, 90] and chosen ciphertext attacks [72, 88], (hierarchical) IBE schemes [2, 3, 26, 103], ABE schemes [19, 21, 45, 47], PE schemes [4, 38, 44, 102], fully homomorphic encryption schemes [23, 24] (pioneered by Gentry’s work [40]), etc. All these cryptosystems enjoy provable security under worst-case hardness assumptions.

Beyond the conjectured resistance of quantum computers and strong security guarantees from worst-case hardness, lattice-based cryptosystems are asymptotically efficient as well. The schemes are often highly parallelizable and algorithmically simple, since they mainly consist of linear operations on vectors and matrices modulo relatively small integers. In contrast, for cryptosystems based on number-theoretic problems, the cost of the corresponding operations is much higher, such as generating huge prime numbers and exponentiating modulo these primes.

1.3 Contributions of the Thesis

Motivations. As discussed above, revocation is a necessary and imperative functionality in multi-user encryption systems and lattices have become one of the most exciting and powerful tools in the field, especially in the last decade. However, the study of revocable cryptosystems has not got much attention in the world of lattices. To the best of our knowledge, revocation was only considered in the context of lattice-based IBE.

In 2012, Chen et al. [28] initiated the study of lattice-based revocable IBE by providing a revocation mechanism for the scheme by Agrawal et al. [2]. They exploited the key-update revocation mechanism introduced by Boldyreva et al. [15], roughly modelled as follows. The KGC, who possesses the current list of revoked identities,

has to publish an update key at each time period so that only non-revoked users can update their private keys to decrypt ciphertexts bound to the same time slot. This mechanism is also known as indirect revocation, since the revocation information is not controlled by the message sender, but by the KGC. To make the KGC's workload scalable (i.e., being logarithmic in the maximum number of users N), Boldyreva et al. suggested to employ the complete subtree (CS) method [75] to handle key updating. In [28], to combine Agrawal et al.'s IBE scheme [2] and the CS method, the authors employed a syndrome splitting technique which can be seen as a variant of secret sharing [97]. Chen et al.'s construction has been improved in two directions. Cheng and Zhang [29] improved the efficiency using the subset difference method [75]. Takayasu and Watanabe [100] developed a scheme with enhanced security, which is, to some extent, resistant against a sophisticated threat called decryption key exposure attacks [96].

Very recently, a major related result was obtained by Agrawal et al. [1]. They considered user revocation under the direct revocation mechanism, where the revocation list is directly encrypted into each ciphertext. This approach eliminates the necessity of the key-update phase, while it produces ciphertexts of relatively large size, depending on the number of all users N and/or the number of revoked users r . In [1], the main result is an elegant generic construction from inner products functional encryption (IPFE) to identity-based trace-and-revoke system. Particularly, the authors observed that such a transformation requires the underlying FE be secure under a much weaker security model, based on which they provided an improved LWE-based IPFE.

On the contrary, there have been more investigations of the revocation functionality for pairing-based cryptosystems, not only in the IBE setting but also in the more general contexts of ABE and PE.

Boldyreva et al. [15] proposed the indirect revocation mechanism and gave instantiations for both IBE and ABE schemes (the main content of their paper is about revocable IBE and analysis of revocable ABE is only mentioned from scratch). Sub-

sequently, for their revocable ABE, there are concrete constructions in [10, 92]. For Boldyreva et al.’s revocable IBE, there were attempts to improve the security in [64, 96]. Specially, to reduce users’ computational burden, Qin et al. [89] introduced a server into the key-update revocation mechanism. The server functions as a publicly accessible computer with powerful computational capabilities, to which one can outsource most of users’ workloads. Such a revocation mechanism was considered in the settings of IBE and ABE by Qin et al. [89] and Cui et al. [31], respectively.

Under direct revocation model, Lewko et al. [57] proposed an identity-based revocation system with very small private keys, i.e., of size independent from N and r . Chen et al. [27] gave a generic construction from non-zero inner-products PE to identity-based revocation system, where both private keys and ciphertexts have constant sizes but the size of public parameters is linear in N . Beyond the IBE setting, Attrapadung and Imai [10] considered ABE supporting direct revocation mechanism and Nieto et al. [78] further extended it to the PE setting.

Our Contributions. Motivated by the above situation, in this thesis, we focus on two problems: equipping efficient revocation mechanism for LWE-based encryption schemes and designing new revocation models.

To solve the first problem, we investigated the revocation models employed in latest revocable cryptosystems (not lattice-based) and found some mechanisms are desirable to be adapted into the lattice world because of the significant advantages they offer. However, most of proposals under other assumptions (e.g., pairing assumptions) are not described in the modular way or with generic constructions that might have certain limitations compared with concrete schemes. It then follows that adapting new revocation models into lattice-based cryptosystems is non-trivial.

Furthermore, the techniques employed in pairing-based revocable schemes might not be applicable on lattices as algorithms under the two assumptions are fundamentally different. In particular, the dual encryption system [101] is well-studied on pairings while there is still no lattice category. Hence, designing lattice-based revocable

cryptosystems emerges the need of novel techniques.

To this end, we usually perform in two steps: (i) analyze the necessary building blocks the revocation methods require; (ii) find proper ways to combine the ingredients. At step (i), our instantiations often require one ordinary (non-revoked) lattice-based scheme and one method to realize revocations. At step (ii), we should take advantage of certain techniques from LWE-based encryption systems. In many cases, suitable modifications of the techniques are demanded to smoothly combine the building blocks.

To address the second problem, we start with some existing efficient revocation mechanisms and try to extend them to other settings. Reasonable extensions should take account of features of the contexts. Moreover, our new revocation model is feasible for modern applications (e.g., cloud computing with fine-grained access control).

More specifically, we have the following contributions.

Contribution 1. We construct a revocable IBE which is more efficient than all existing such schemes from lattices. We follow the architecture of sever-aided revocable identity-based encryption (SR-IBE), proposed by Qin et al. [89] in the year of 2015. The sever-aided revocation paradigm (an illustration picture is given in Figure 4.1) significantly improves efficiency of the key-update revocation mechanism [15]. In the server-aided revocation model, there is a computationally powerful server so that most of users' workloads, key updating, can be outsourced to this sever. Namely, the update keys are only sent to the sever rather than all users. Meanwhile, for each user, the sever is sent a corresponding token by the KGC. Such a server then aids users' decryptions in the sense that it will transform ciphertexts to "partially decrypted ciphertexts" before sending to users. Particularly, the sever can be untrusted since it does not possess any secret information, who just performs public storages and correct computations.

Attracted by the efficiency advantages of sever-aided revocations, we propose a lattice-based instantiation of SR-IBE. We make use of two building blocks: the hierarchical IBE (HIBE) by Agrawal et al. [2] and the revocable IBE (RIBE) by Chen et al. [28]. To combine these two ingredients, we observe that a "double encryption"

mechanism [59] is well-suited in such a server-aided system. Roughly speaking, our ciphertext is computed by two layers: running the HIBE encryption algorithm with the original message and generating the final ciphertext by using the underlying RIBE to encrypt the resulted ciphertext from HIBE. The sever, who holds the decryption key of the RIBE scheme, then can perform RIBE decryption to recover a valid ciphertext for the HIBE system. Finally, the recipient reveals the original message using his private key, which is obtained from the underlying HIBE. Under this circumstance, the double encryption mechanism perfectly makes the server aid in users' decryption procedure.

Contribution 2. We first formalize the model of server-aided revocable predicate encryption (SR-PE) and then put forward an instantiation of SR-PE from lattices. The model can be seen as a non-trivial adaptation of the sever-aided revocation mechanism into the setting of PE. We design a lattice-based SR-PE scheme that is proven secure (in a selective manner) in the standard model.

At a high level, our scheme employs two main building blocks: the ordinary PE scheme by Agrawal et al. [4] and the CS method due to Naor et al. [75]. Our first challenge is to enable a relatively sophisticated mechanism, in which an original PE ciphertext bound to an attribute (but not bound to any user's identifying information since the attribute may be shared by many users), after being transformed by the server, would become a partially decrypted ciphertext bound to the identifying information of the non-revoked recipient. To this end, we introduce the IBE instance from [2] into our SR-PE construction. Namely, we assign each user an identity id and embed id into the user-specific token in a way such that the partially decrypted ciphertext will be bound to id .

More specifically, we exploit a special property of some **LWE**-based encryption systems, observed by Boneh et al. [19], which allows to transform an encryption of a message under one key into an encryption of the same message under another key. Each user with identity id is issued a private key for a two-level hierarchical system consisting of one instance of the PE system from [4] as well as an additional IBE level

to deal with id , associated with a matrix $\mathbf{D}_{\text{id}}$. Meanwhile, the server-side token for id is generated by embedding $\mathbf{D}_{\text{id}}$ into another instance of the same PE system. A ciphertext in our scheme consists of two PE ciphertexts. If recipient id is not revoked, the server can transform the second PE ciphertext into an IBE ciphertext associated with $\mathbf{D}_{\text{id}}$, thanks to the special property mentioned above. Finally, the partially decrypted ciphertext, including the first PE ciphertext and the IBE ciphertext, can be fully decrypted using the private key of id .

Contribution 3. We propose another revocable PE scheme from lattices. The construction follows the direct revocation mechanism, which forces the ciphertext to carry on the current list of revoked keys so that users’ private keys does not need to be updated after they join the system. Such a mechanism was first considered in the PE setting by Nieto et al. [78], with paring-based instantiations. They defined a strong security notion for revocable PE schemes, the full-hiding security, which ensures that the ciphertext does not leak any information about the associated message and attribute, as well as the revocation information it carries on. The privacy of revocation information is useful in applications where such information is required to keep hidden from the recipient’s view.

Following their work, we present the first lattice-based PE supporting direct revocations. We employ the same building blocks as in the previous SR-PE scheme, namely, the PE scheme by Agrawal et al. [4] and the CS method due to Naor et al. [75]. However, new techniques for the combination are needed since sever-aided revocation and direct revocation models are essentially different.

At the heart of our constructor is the use of a variant of the Agrawal et al.’s IBE scheme [2], which admits the anonymous property. Informally, it is assumed that each user’s private key is issued associated with not only a predicate but also a unique index. The latter can be treated as an “identity” as in the IBE system. Meanwhile, the indexes of revoked keys are encoded into ciphertexts. To guarantee that revoked keys are useless in decrypting, we use a splitting technique [28] to combine this anonymous

IBE scheme with the original PE. The privacy of attributes and messages is based on the PE scheme and the privacy of revocation lists can be achieved thanks to the anonymity of the IBE scheme.

Applications. Recall that in Section 1.1, we described two examples: the e-mail system as an application of IBE systems, which demands short size of ciphertexts; the credit card fraud investigation problem for PE schemes, where customers most care about the size of private keys. It is necessary to equip both systems with efficient revocation mechanisms since users might leave for certain reasons or their keys should be expired after some time. Our construction of server-aided revocable IBE scheme seems suitable in the e-mail system and our two revocable PE schemes (one following the server-aided model and another based on the direct revocation model) can serve as good choices for credit card investigations.

1.4 Organization of the Thesis

The rest of the thesis is organized into the following 5 chapters:

- ◇ In Chapter 2, we first define some general notations, then we recall some lattice-based cryptographic definitions and techniques which will be used in our constructions introduced in the subsequent chapters. We also describe some related schemes from previous works.
- ◇ In Chapter 3, we construct an identity-based encryption supporting server-aided revocations.
- ◇ In Chapter 4, we introduce a new model to address user revocation for predicate encryption schemes and then give a lattice-based instantiation.
- ◇ In Chapter 5, we propose another revocable predicate encryption equipped with the direct revocation mechanism.

- ◊ In Chapter 6, we give a summary of the results and discuss about relevant open problems in the field. We also list several possible topics for future investigations.

To balance the main body of the thesis, the detailed security proofs for the schemes in Chapter 3, 4, 5 are given in the Appendix, while the high-level ideas are mentioned in each chapter.

First, the results presented in Chapter 3 is based on the paper from [77] co-authored with Prof. Huaxiong Wang and Dr. Khoa Nguyen, which was published in the proceedings of CANS 2016. Second, the results presented in Chapter 4 is based on the paper from [65] co-authored with Prof. San Ling, Prof. Huaxiong Wang, and Dr. Khoa Nguyen, which was submitted to Designs, Codes and Cryptography. Finally, the results presented in Chapter 5 is based on the paper from [66] co-authored with the same cryptographers as in [65], which was published in the proceedings of ProvSec 2017. This thesis author was the main investigator and author of these three papers.

Chapter 2

Definitions and Preliminaries

2.1 General Notations

In the thesis, we will employ a series of general notations, defined as follows:

Vectors, Matrices and Norms. We use bold lower-case letters (for instance, $\mathbf{x}, \mathbf{y}$) and bold upper-case letters (for instance, $\mathbf{A}, \mathbf{B}$) to denote column vectors and matrices, respectively. In particular, over-arrows $\overrightarrow{x}, \overrightarrow{y}$ are used to denote predicate and attribute vectors. For vector $\mathbf{x} \in \mathbb{R}^m$ and vector $\mathbf{y} \in \mathbb{R}^k$, we denote by $(\mathbf{x} \parallel \mathbf{y}) \in \mathbb{R}^{m+k}$ the concatenation of $\mathbf{x}$ and $\mathbf{y}$. In the similar manner, the column-concatenation of matrix $\mathbf{A} \in \mathbb{R}^{n \times k}$ and matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$ is denoted by $[\mathbf{A} \mid \mathbf{B}] \in \mathbb{R}^{n \times (k+m)}$.

Given a vector $\mathbf{x} = (x_1, \dots, x_m) \in \mathbb{R}^m$, the ℓ_p norm (for $1 \leq p < \infty$) of $\mathbf{x}$ is defined as $\|\mathbf{x}\|_p := (|x_1|^p + \dots + |x_m|^p)^{\frac{1}{p}}$ while the infinity norm ℓ_∞ of $\mathbf{x}$ is computed as $\|\mathbf{x}\|_\infty := \max_{i \in [m]} |x_i|$. To simplify the notation, $\|\cdot\|$ is used to denote the ℓ_2 norm.

Let $\mathbf{A} = [\mathbf{a}_1 \mid \dots \mid \mathbf{a}_m] \in \mathbb{R}^{n \times m}$, we denote by $\|\mathbf{A}\|$ the norm of $\mathbf{A}$, the highest ℓ_2 norm of any column of $\mathbf{A}$, i.e., $\|\mathbf{A}\| := \max_{i \in [m]} \|\mathbf{a}_i\|$. If $\mathbf{a}_1, \dots, \mathbf{a}_m$ are linearly independent, let $\widetilde{\mathbf{A}} := [\widetilde{\mathbf{a}}_1 \mid \dots \mid \widetilde{\mathbf{a}}_m]$ represent the Gram-Schmidt orthogonalization of matrix $\mathbf{A}$. Specifically, $\widetilde{\mathbf{a}}_1, \dots, \widetilde{\mathbf{a}}_m$ are defined iteratively as: $\widetilde{\mathbf{a}}_1 = \mathbf{a}_1$; and for each $i \in \{2, \dots, m\}$, $\widetilde{\mathbf{a}}_i = \mathbf{a}_i - \sum_{j \leq i} \frac{\langle \mathbf{a}_i, \widetilde{\mathbf{a}}_j \rangle}{\langle \widetilde{\mathbf{a}}_j, \widetilde{\mathbf{a}}_j \rangle} \cdot \widetilde{\mathbf{a}}_j$. In the geometrical sense, vector $\widetilde{\mathbf{a}}_i$ is the component of $\mathbf{a}_i$ orthogonal to the space S spanned by $\mathbf{a}_1, \dots, \mathbf{a}_{i-1}$, i.e., $S = \text{span}(\mathbf{a}_1, \dots, \mathbf{a}_{i-1})$.

Particularly, the notion $\|\widetilde{\mathbf{A}}\|$ is used to represent the Gram-Schmidt norm of $\mathbf{A}$.

Other Notations. For a finite set Ω , we use the notion $x \xleftarrow{\$} \Omega$ to indicate that element x is sampled uniformly at random from Ω . For convenience, we say that sampler $\vec{x}$ is uniformly random in set Ω to state that $\vec{x}$ is uniformly distributed random in Ω . The notion $x \leftarrow \mathcal{D}$ is used to denote that we sample x following the probability distribution $\mathcal{D}$. Furthermore, when x is the output of an algorithm named $\mathcal{B}$, we denote as $x \leftarrow \mathcal{B}$.

To evaluate the asymptotic time complexity of algorithms, we use the following notation: $f(n) = O(g(n))$ if $\exists k > 0, \exists n_0$ such that $\forall n > n_0, f(n) \leq kg(n)$. Moreover, the soft-O notation is used: $f(n) = \tilde{O}(g(n))$ if for some constant c , $f(n) = O(g(n) \cdot \log^c g(n))$.

We are going to use n to represent the corresponding security parameter of all the cryptosystems throughout this thesis, which exactly is the dimension of the lattices the computational problems relies on.

Algorithms are regarded as efficient if they run in probabilistic polynomial time (PPT). A function $d : \mathbb{N} \rightarrow \mathbb{R}$ is said to be negligible, if $d(n)$ is $O(n^{-c})$ for all constant $c > 0$. Over a discrete domain Γ , the statistical distance of two random variables X and Y is defined as $\Delta(X; Y) := \frac{1}{2} \sum_{s \in \Gamma} |\Pr[X = s] - \Pr[Y = s]|$. For random variables $X(n)$ and $Y(n)$, we say that variables X and Y are statistically close, or the distributions of X and Y are indistinguishable, if $\Delta(X(n); Y(n))$ is a negligible function in n .

For a positive integer k , the notation $[k]$ represents set $\{1, \dots, k\}$. If c is a real number, we use $\lfloor c \rfloor = \lfloor c - 1/2 \rfloor$ to denote the integer closest to c .

2.2 Preliminaries on Lattices

2.2.1 Lattices

In this thesis, we focus on full-rank lattices, formally defined in the following.

Definition 2.2.1 (Lattices). Let vectors $\mathbf{b}_1, \dots, \mathbf{b}_n \in \mathbb{R}^n$ be linearly independent. The lattice generated by $\mathbf{B} = [\mathbf{b}_1 \mid \dots \mid \mathbf{b}_n] \in \mathbb{R}^{n \times n}$ is the set:

$$\Lambda(\mathbf{B}) = \left\{ \mathbf{y} \in \mathbb{R}^n : \exists \mathbf{x} \in \mathbb{Z}^n, \mathbf{y} = \mathbf{B}\mathbf{x} \right\} = \left\{ \sum_{i=1}^n x_i \cdot \mathbf{b}_i : x_i \in \mathbb{Z} \right\},$$

which contains all the integer linear combinations of vectors $\mathbf{b}_1, \dots, \mathbf{b}_n$. The integer n is called the dimension of $\Lambda(\mathbf{B})$ and the matrix $\mathbf{B}$ is called a basis of this lattice.

Actually, for a given lattice Λ , there are infinitely many bases, the quality of which are typically measured using their norms. Given a particular basis $\mathbf{B}$ of the lattice Λ , the determinant $\det(\Lambda)$ of the lattice, defined as $\det(\mathbf{B}^\top \mathbf{B})$, is an invariant of Λ . Another invariant of the lattice, the successive minima, is defined as follows.

Definition 2.2.2 (Successive minima). Let Λ be an n -dimensional lattice. For each $i \in [n]$, the i -th successive minimum $\lambda_i(\Lambda)$ is the smallest number $r > 0$ such that there exist i linearly independent vectors $\mathbf{s}_1, \dots, \mathbf{s}_i \in \Lambda$ satisfying $\|\mathbf{s}_j\| \leq r$ for all $j \in [i]$.

Geometrically, the i -th successive minimum is the radius of the smallest ball containing i linearly independent lattice vectors. In particular, λ_1 is the length of the shortest non-zero vector in lattice Λ .

Next, we review two basic worst-case approximation problems on lattices. In both problems, $\gamma(n)$ is an approximation factor, which should not be less than 1.

Definition 2.2.3 (Shortest Vector Problem (Decision Version)). An input to the decision version of the shortest vector problem (**GapSVP**) is a basis $\mathbf{B}$ of a full-rank n -dimensional lattice $\Lambda(\mathbf{B})$. It is a YES instance if $\lambda_1(\Lambda(\mathbf{B})) \leq 1$ and is a NO instance if $\lambda_1(\Lambda(\mathbf{B})) > \gamma(n)$.

Definition 2.2.4 (Shortest Independent Vectors Problem). On input a full-rank basis $\mathbf{B}$ of an n -dimensional lattice $\Lambda(\mathbf{B})$, the goal of the shortest independent vectors

problem (SIVP) is to output n linearly independent vectors $\mathbf{s}_1, \dots, \mathbf{s}_n \in \Lambda(\mathbf{B})$ such that $\|\mathbf{S}\| \leq \gamma(n) \cdot \lambda_n(\Lambda(\mathbf{B}))$, where $\mathbf{S} = [\mathbf{s}_1 \mid \dots \mid \mathbf{s}_n]$.

The conjectured hardness of the worst-case problems is the fundamental security assumption for lattice-based cryptography. Specifically, it is conjectured that there is no polynomial-time (even quantum) algorithm that approximates lattice problems (e.g., SIVP, GapSVP) to within polynomial factors.

A lot of lattice-based encryption systems are essentially based on the average-case problem LWE, which is closely connected with the q -ary lattices.

Definition 2.2.5 (q -array lattices). For integer $q \geq 2$ and $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, define the q -ary lattices:

$$\begin{aligned}\Lambda_q^\perp(\mathbf{A}) &= \{\mathbf{r} \in \mathbb{Z}^m : \mathbf{A}\mathbf{r} = \mathbf{0} \bmod q\}, \\ \Lambda_q(\mathbf{A}) &= \{\mathbf{r} \in \mathbb{Z}^m : \mathbf{r} = \mathbf{A}^\top \mathbf{s} \bmod q \text{ for some } \mathbf{s} \in \mathbb{Z}_q^n\}.\end{aligned}$$

For any $\mathbf{u}$ in the image of $\mathbf{A}$, define the coset

$$\Lambda_q^\mathbf{u}(\mathbf{A}) = \{\mathbf{r} \in \mathbb{Z}^m : \mathbf{A}\mathbf{r} = \mathbf{u} \bmod q\}.$$

2.2.2 Gaussian Distributions

Next, we review one of the fundamental tools in lattice-based cryptography: Gaussian distributions (particularly, over $\mathbb{Z}^m$ and the q -ary lattices).

The (continuous) Gaussian distribution over $\mathbb{R}^m$, with center $\mathbf{c} \in \mathbb{R}^m$ and parameter $s > 0$, is defined by its density function

$$\forall \mathbf{x} \in \mathbb{R}^m : \rho_{s,\mathbf{c}}(\mathbf{x}) = \exp\left(-\pi \frac{\|\mathbf{x} - \mathbf{c}\|^2}{s^2}\right)$$

In the lattice world, we are especially interested in discrete Gaussian distribution over integer lattices, i.e., $\Lambda \subseteq \mathbb{Z}^m$. The discrete Gaussian distribution over lattice Λ

with center $\mathbf{c}$ and parametered by s , is given by the density function

$$\forall \mathbf{x} \in \Lambda : \mathcal{D}_{\Lambda, s, \mathbf{c}}(\mathbf{x}) = \frac{\rho_{s, \mathbf{c}}(\mathbf{x})}{\rho_{s, \mathbf{c}}(\Lambda)}.$$

where $\rho_{s, \mathbf{c}}(\Lambda) = \sum_{\mathbf{x} \in \Lambda} \rho_{s, \mathbf{c}}(\mathbf{x})$. When $\mathbf{c} = \mathbf{0}$, to simplify the notations, we often use ρ_s and $\mathcal{D}_{\Lambda, s}$.

In [41], Gentry et al. showed, for lattices with sufficiently short bases, we can efficiently obtain samples from discrete Gaussian distributions.

Lemma 2.2.6 ([41]). There exists a PPT algorithm $\text{SampleGaussian}(\mathbf{B}, s, \mathbf{c})$ that, taking as input a basis $\mathbf{B}$ of an m -dimensional lattice Λ , a Gaussian parameter $s \geq \|\tilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log m})$, and center $\mathbf{c} \in \mathbb{R}^m$, outputs a sample from a distribution that is statistically close to $\mathcal{D}_{\Lambda, s, \mathbf{c}}$.

Particularly, let $\mathbf{B}_z$ be the standard basis for $\mathbb{Z}^m$ and $s \geq \omega(\sqrt{\log m})$, then we can use $\text{SampleGaussian}(\mathbf{B}_z, s, \mathbf{0})$ to sample from distribution $\mathcal{D}_{\mathbb{Z}^m, s}$.

The discrete Gaussian distribution also enjoys nice properties, restated in the following lemma.

Lemma 2.2.7 ([41, 73]). Let $q \geq 2$ be a prime and $m \geq 2n \log q$ be an integer. Assume that $\mathbf{A}$ is a matrix in $\mathbb{Z}_q^{n \times m}$. Let $\mathbf{T}_{\mathbf{A}}$ be a basis of $\Lambda_q^\perp(\mathbf{A})$ and $s \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log m})$ be a Gaussian parameter. Then:

1. $\Pr [\mathbf{x} \leftarrow \Lambda_q^\perp(\mathbf{A}) : \|\mathbf{x}\| > s\sqrt{m}]$ is negligible in n .
2. For $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^m, s}$, the distribution of $\mathbf{u} = \mathbf{Ax} \bmod q$ is statistically close to uniform over $\mathbb{Z}_q^n$.

2.2.3 The LWE Problem

The Learning With Errors (LWE) problem, first introduced in the ground-breaking work by Regev [90], serves as a basis in lattice-based cryptography. The LWE problem, having the search and the decision versions, is defined below.

Definition 2.2.8 $((n, q, \chi)$ -LWE). Let $q \geq 2, m \geq 1$, and let χ be a probability distribution on $\mathbb{Z}$ (or over $\mathbb{Z}_q$). For $\mathbf{s} \in \mathbb{Z}_q^n$, distribution $\mathcal{A}_{\mathbf{s}, \chi}$ is obtained by sampling vector $\mathbf{a} \xleftarrow{\$} \mathbb{Z}_q^n$ and error $e \leftarrow \chi$, and returning the pair $(\mathbf{a}, \mathbf{a}^\top \mathbf{s} + e) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

1. The **Search-LWE** problem asks to find $\mathbf{s}$, given m independent samples from $\mathcal{A}_{\mathbf{s}, \chi}$.
2. The **Decision-LWE** problem requires to distinguish m samples chosen according to distribution $\mathcal{A}_{\mathbf{s}, \chi}$ (for some $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$) and a uniformly random sample on $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q$.

In lattice-based cryptosystems, we are mainly interested in the **LWE** problem with sufficiently small errors. Namely, there is a bound $B \ll q$ such that the probability that the norm exceeds this bound is negligibly small.

Definition 2.2.9 (B -bounded distribution, [23, 42]). We say that a distribution χ over integers is B -bounded, if the probability of a sample $e \leftarrow \chi$ satisfying $|e| > B$ is negligible.

In many cryptographic applications, we usually consider the (n, q, χ) -LWE problem in the matrix form. Let matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and vector $\mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m$ where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{e} \leftarrow \chi^m$. The **Search-LWE** problem asks to find $\mathbf{s}$, given the pair $(\mathbf{A}, \mathbf{b})$, and the **Decision-LWE** requires to distinguish the pair $(\mathbf{A}, \mathbf{b})$ from a uniformly random sample on $\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$. Under this setting, the **LWE** problem can be connected to the q -ary lattices as follows: assuming that χ is a B -bounded distribution over $\mathbb{Z}$ and B is sufficiently small, then vector $\mathbf{b}$ can be viewed as a perturbed point in the lattice $\Lambda_q^\top(\mathbf{A})$.

For the strong hardness of the **LWE** problem, we now review several known results. In 2005, Regev [90] showed that for a special error distribution, denoted by $\bar{\Psi}_\alpha$, the **Search-LWE** problem is at least as hard as approximating **SIVP** and **GapSVP** using a quantum reduction. The distribution $\bar{\Psi}_\alpha$ is defined to be the discretized Gaussian distribution over $\mathbb{Z}_q$, parametrized by $\alpha/\sqrt{2\pi}$. Later, Peikert [86] showed

that similar results hold for $\chi = \mathcal{D}_{\mathbb{Z}, \alpha q}$ with $\alpha \in (0, 1)$ such that $\alpha q > 2\sqrt{n}$ (note that the distribution $\mathcal{D}_{\mathbb{Z}, \alpha q}$ is B -bounded for $B > \alpha q \cdot \omega(\log n)$). In [90], Regev also showed that, for prime $q = \text{poly}(n)$, the **Search-LWE** and the **Decision-LWE** versions are essentially equivalent. The search-to-decision reduction was further studied in the later works [71, 72, 85], along with some improvements that enable a reduction that goes well with a wider class of error distributions and moduli. The following theorem, presented in [23, 42], summarizes the results about the quantum hardness of **LWE**.

Theorem 2.2.10 (Quantum hardness of **LWE**, [23, 42, 71, 72, 85, 90]). Set $q = q(n)$ to be either a product of small (i.e., size $\text{poly}(n)$) distinct primes or a prime power, and set the error bounded by $B \geq \sqrt{n} \cdot \omega(\log n)$. Then there exists a B -bounded distribution χ such that: if there is an efficient algorithm solving the average-case (n, q, χ) -**LWE** problem, then there is an efficient *quantum* algorithm to approximate the worst-case **SIVP** and **GapSVP** to within approximation factor $\gamma = \tilde{O}(nq/B)$.

Although it is conjectured that **SIVP** and **GapSVP** are hard even on quantum computers, the reduction in the above theorem is quite unsatisfactory. Alternatively, several works have been done to classically reduce the hardness for the **LWE** problem to worst-case problems. We summarize these results in the following theorem.

Theorem 2.2.11 (Classical hardness of **LWE**, [22, 85]). If there exists a PPT algorithm solving the (n, q, χ) -**LWE** problem, then:

1. For $q \geq 2^{n/2}$, there is a PPT algorithm for the **GapSVP** problem, on any n -dimensional lattice. [85]
2. For $q = \text{poly}(n)$, there is a PPT algorithm for the **GapSVP** problem, on any lattice with dimension $\sqrt{n}$. [22]

Note that whereas the reduction in [22] needs a quadratic loss in the dimension, the reduction in [85] requires q to be exponentially large. Until now, establishing a classical hardness for **LWE** with generally useful parameters (e.g., the parameters specified in Theorem 2.2.10) remains as one of the most interesting problems in the field.

2.3 Basic Cryptographic Primitives from Lattices

2.3.1 Sampling Algorithms

In lots of lattice-based cryptosystems, the following two important algorithms serve as bases: the trapdoor generation algorithm [7, 9, 72] and the preimage sampling algorithm [41]. The former generates a (statistically close) uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a sufficiently short basis $\mathbf{T}_{\mathbf{A}}$ of lattice $\Lambda_q^\perp(\mathbf{A})$, and the latter, taking as input a good basis of $\Lambda_q^\perp(\mathbf{A})$ and any vector $\mathbf{u}$, generates a short vector in coset $\Lambda_q^{\mathbf{u}}(\mathbf{A})$. In the following lemma, we review these two basic algorithms.

Lemma 2.3.1 ([7, 9, 41, 72]). Let $q \geq 2$ and $m \geq 2n \log q$.

1. There is an efficient algorithm $\text{TrapGen}(n, q, m)$ which generates a pair of matrices $(\mathbf{A}, \mathbf{T}_{\mathbf{A}})$ such that: matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is statistically close to uniform and matrix $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$ is a basis for $\Lambda_q^\perp(\mathbf{A})$ with

$$\|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \leq O(\sqrt{n \log q}) \text{ and } \|\mathbf{T}_{\mathbf{A}}\| \leq O(n \log q)$$

except for negligible probability in n .

2. Given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, a basis $\mathbf{T}_{\mathbf{A}}$ of lattice $\Lambda_q^\perp(\mathbf{A})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ which is in the range of $\mathbf{A}$, and a Gaussian parameter $s \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\log n)$, algorithm $\text{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}, s)$ outputs a vector $\mathbf{e} \in \mathbb{Z}^m$ that is sampled from a distribution negligible from $\mathcal{D}_{\Lambda_q^{\mathbf{u}}(\mathbf{A})}$.

In [72], Micciancio and Peikert introduced a very structured matrix $\mathbf{G}$, which is known as the *primitive matrix*. Matrix $\mathbf{G}$ simply admits a public short basis for lattice $\Lambda_q^\perp(\mathbf{G})$. We recall the properties of the primitive matrix $\mathbf{G}$ defined in [72].

Lemma 2.3.2 ([72]). Let $q \geq 2$ and $k = \lceil \log q \rceil$ be integers. Define

$$\mathbf{g} := (1, 2, \dots, 2^k) \in \mathbb{Z}^k \quad \text{and} \quad \mathbf{G} := \mathbf{I}_n \otimes \mathbf{g}.$$

Then there exists a known basis $\mathbf{T}_{\mathbf{G}} \in \mathbb{Z}^{nk \times nk}$ of lattice $\Lambda_q^\perp(\mathbf{G})$ satisfying $\|\widetilde{\mathbf{T}_{\mathbf{G}}}\| \leq \sqrt{5}$.

Moreover, let j be a positive integer. There is a deterministic polynomial-time algorithm $\mathbf{G}^{-1}$ that, given a matrix $\mathbf{U} \in \mathbb{Z}_q^{n \times j}$, returns a matrix $\mathbf{X} = \mathbf{G}^{-1}(\mathbf{U})$ where $\mathbf{X} \in \{0, 1\}^{nk \times j}$ and $\mathbf{GX} = \mathbf{U}$.

Finally, define $\text{bin} : \mathbb{Z}_q \rightarrow \{0, 1\}^k$ to be the function that maps $w \in \mathbb{Z}_q$ to the binary decomposition of w . We have $\langle \mathbf{g}, \text{bin}(w) \rangle = w$, for each $w \in \mathbb{Z}_q$.

In this thesis, it is simply assumed that the dimension of the primitive matrix $\mathbf{G}$ is the same as the matrix $\mathbf{A}$ outputted by algorithm $\text{TrapGen}(n, q, m)$.

In [2, 72], it was shown that, given a short basis for lattice $\Lambda_q^\perp(\mathbf{A})$, there are efficient algorithms to sample short vectors in the super-lattice, and to delegate equally efficient bases. In addition, without the basis, one can simulate such a sampling procedure, which will be employed to make the security proof works. Particularly, we will employ the sampling algorithms SampleLeft , SampleBasisLeft , and SampleRight , defined as follows.

SampleBasisLeft ($\mathbf{A}, \mathbf{M}, \mathbf{T}_{\mathbf{A}}, s$) : Taking as input a rank n matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ equipped with a basis $\mathbf{T}_{\mathbf{A}}$ of $\Lambda_q^\perp(\mathbf{A})$, a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times m_1}$ and a Gaussian parameter $s \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log(m + m_1)})$, it generates a basis $\mathbf{T}_{\mathbf{F}}$ of lattice $\Lambda_q^\perp(\mathbf{F})$ which preserves the Gram-Schmidt norm of the basis (i.e., $\|\widetilde{\mathbf{T}_{\mathbf{F}}}\| = \|\widetilde{\mathbf{T}_{\mathbf{A}}}\|$). Here, we define $\mathbf{F} := [\mathbf{A} \mid \mathbf{M}] \in \mathbb{Z}_q^{n \times (m+m_1)}$.

SampleLeft ($\mathbf{A}, \mathbf{M}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}, s$) : Taking as input full-rank matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ equipped with a basis $\mathbf{T}_{\mathbf{A}}$ of $\Lambda_q^\perp(\mathbf{A})$, a matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times m_1}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a Gaussian parameter $s \geq \|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \cdot \omega(\sqrt{\log(m + m_1)})$, it samples a vector $\mathbf{z} \in \mathbb{Z}^{(m+m_1)}$, which is sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^\perp(\mathbf{F}), s}$. Here we define $\mathbf{F} := [\mathbf{A} \mid \mathbf{M}] \in \mathbb{Z}_q^{n \times (m+m_1)}$.

SampleRight ($\mathbf{A}, \mathbf{R}, t, \mathbf{G}, \mathbf{T}_{\mathbf{G}}, \mathbf{u}, s$) : Taking as input matrices $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, $\mathbf{R} \in \mathbb{Z}^{m \times m}$, a scalar $t \in \mathbb{Z}_q \setminus \{0\}$, the primitive matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ equipped with a publicly

known trapdoor $\mathbf{T}_{\mathbf{G}}$ of $\Lambda_q^\perp(\mathbf{G})$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, and a Gaussian parameter $s \geq \|\widetilde{\mathbf{T}_{\mathbf{B}}}\| \cdot \|\mathbf{R}\| \cdot \omega(\sqrt{\log m})$, it outputs a vector $\mathbf{z} \in \mathbb{Z}^{2m}$, which is sampled from a distribution statistically close to $\mathcal{D}_{\Lambda_q^\perp(\mathbf{F}),s}$. Here we define $\mathbf{F} := [\mathbf{A} \mid \mathbf{A}\mathbf{R} + t\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$.

All these sampling algorithms can be easily adapted in applications where one takes as input a matrix $\mathbf{U} \in \mathbb{Z}_q^{n \times k}$ for certain integer $k \geq 1$, instead of giving a vector $\mathbf{u} \in \mathbb{Z}_q^n$. In this scenario, the above algorithms would output a matrix $\mathbf{Z}$ over $\mathbb{Z}^{2m \times k}$.

The following lemma from [2], which can be viewed as a variant of left-over hash lemma, will be used to show the security of our revocable encryption system in Chapter 5.

Lemma 2.3.3. Let $q > 2$ be a prime. Let $m > (n+1) \log q + \omega(\log n)$ and $\beta = \text{poly}(n)$. Select $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{R} \xleftarrow{\$} \{-1, 1\}^{m \times \beta}$, and $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n \times \beta}$. The joint distribution of $(\mathbf{A}, \mathbf{A}\mathbf{R}, \mathbf{R}^\top \mathbf{v})$ is indistinguishable from the distribution of $(\mathbf{A}, \mathbf{B}, \mathbf{R}^\top \mathbf{v})$, for any vector $\mathbf{v} \in \mathbb{Z}_q^m$.

The following lemma is also needed to demonstrate the correctness and security of our constructions in Chapter 4 and Chapter 5.

Lemma 2.3.4 (Adapted from [41, 72]). Let $m \geq 2n \log q$, $\kappa \geq 1$, and $q \geq 2$ be integers. Assume that $\mathbf{F} \in \mathbb{Z}_q^{n \times m}$ has rank n and $\mathbf{T}_{\mathbf{F}}$ is a basis of $\Lambda_q^\perp(\mathbf{F})$. If $s \geq \|\widetilde{\mathbf{T}_{\mathbf{F}}}\| \cdot \omega(\sqrt{\log n})$ and $\mathbf{Z} \leftarrow (\mathcal{D}_{\mathbb{Z}_q^m, s})^\kappa$, then the distribution of matrix $\mathbf{F}\mathbf{Z} \bmod q$ is indistinguishable from the uniform distribution over $\mathbb{Z}_q^{n \times \kappa}$.

Particularly, this lemma also holds if matrix $\mathbf{F}$ is the primitive matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ specified in Lemma 2.3.2 or if it is uniformly random over $\mathbb{Z}_q^{n \times m}$.

2.3.2 Public-key Encryption

In this section, we will review the syntax of public-key encryption (PKE) and describe one related lattice-based scheme.

Definition 2.3.5. A public-key encryption scheme $\mathcal{PKE} = (\text{Setup}, \text{Enc}, \text{Dec})$, associated with a message space $\mathcal{M}$, is a triple of probabilistic algorithms as follows:

- $\text{Setup}(1^n)$, given a security parameter n , outputs a pair of public key and secret key (pk, sk) .
- $\text{Enc}(\text{pk}, M)$, given a public key pk and a message $M \in \mathcal{M}$, outputs a ciphertext ct .
- $\text{Dec}(\text{sk}, \text{ct})$, given a secret key sk and a ciphertext ct , returns $M \in \mathcal{M} \cup \perp$.

Correctness. The correctness requirement for a PKE scheme requires that: for any $n \in \mathbb{N}$, any (pk, sk) generated by $\text{Setup}(1^n)$ and any message $M \in \mathcal{M}$,

$$\text{Dec}(\text{sk}, \text{Enc}(\text{pk}, M)) = M$$

holds with overwhelming probability (over the randomness of all algorithms).

Security Notions. The semantic security of PKE is defined as the indistinguishability of ciphertexts under chosen plaintext attacks. In other words, it guarantees that any efficient adversary cannot distinguish the ciphertexts encrypted from two distinct messages.

Definition 2.3.6 (IND-CPA Security). A PKE scheme is IND-CPA secure if any PPT algorithm $\mathcal{A}$ has negligible probability in the following game:

1. $\text{Setup}(1^n)$ is run to generate (pk, sk) and $\mathcal{A}$ is given pk .
2. $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathcal{M}$ and is given the challenge ciphertext $\text{ct}^* \leftarrow \text{Enc}(\text{pk}, M_b)$ for $b \xleftarrow{\$} \{0, 1\}$.
3. $\mathcal{A}$ returns a bit b' and wins if $b' = b$. Adversary $\mathcal{A}$'s advantage is defined as:

$$\text{Adv}_{\mathcal{A}, \mathcal{PKE}}^{\text{IND-CPA}}(n) = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

The probability is over the random bits b and b' .

Remark 2.3.7. The stronger security notion, indistinguishability of ciphertexts under chosen ciphertext attacks (IND-CCA1), grants the adversary access to the decryption oracle before outputting messages M_0, M_1 . There is another security notion, IND-CCA2, which demands that, even after receiving the challenge ciphertext $\mathbf{ct}^*$, the adversary is still allowed to make queries to the decryption oracle on any string that is not equal to $\mathbf{ct}^*$.

The Dual-Regev PKE Scheme

In 2008, Gentry et al. [41] gave a breakthrough work on encryption systems on lattices, which consists of a dual variant of Regev's PKE scheme [90]. Compared with Regev's scheme, the public key $\mathbf{pk}$ in this dual variant is “dense”, in the sense that the public syndrome $\mathbf{u} \in \mathbb{Z}_q^n$ might correspond to many equivalent decryption keys. In the following scheme, we describe the dual-Regev PKE scheme in details.

Scheme 2.3.8 (The Dual-Regev PKE, excerpted from [41]). The scheme works with following parameters: prime $q = O(n^4)$, integer $m = O(2n \log q)$, Gaussian parameter $s \geq \omega(\sqrt{\log m})$ and a B -bounded distribution χ where $B = \tilde{O}(\sqrt{n})$.

Setup(1^n): Sample $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and $\mathbf{r} \leftarrow \mathcal{D}_{\mathbb{Z}_m, s}$. Compute $\mathbf{u} = \mathbf{A}\mathbf{r} \bmod q$ and then output

$$\mathbf{pk} = (\mathbf{A}, \mathbf{u}); \quad \mathbf{sk} = \mathbf{r}.$$

Enc($\mathbf{pk}, M$): To encrypt a message $M \in \{0, 1\}$, choose $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$ and $e \leftarrow \chi$.

Output the ciphertext $\mathbf{ct} = (\mathbf{c}, \tau) \in \mathbb{Z}_q^m \times \mathbb{Z}_q$ where

$$\begin{cases} \mathbf{c} = \mathbf{A}^\top \mathbf{s} + \mathbf{e}, \\ \tau = \mathbf{u}^\top \mathbf{s} + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor. \end{cases}$$

$\text{Dec}(\mathbf{r}, (\mathbf{c}, \tau))$: Compute $d = \tau - \mathbf{r}^\top \mathbf{c} \in \mathbb{Z}_q$. Output $\lfloor \frac{2}{q} \cdot d \rfloor \in \{0, 1\}$.

Analysis. As mentioned in [41], the idea of including multi-syndromes can make the above cryptosystem to encrypt messages with polynomial length.

To demonstrate the correctness, we have:

$$\begin{aligned} d &= \tau - \mathbf{r}^\top \mathbf{c} \\ &= \mathbf{u}^\top \mathbf{s} + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor - \mathbf{r}^\top (\mathbf{A}^\top \mathbf{s} + \mathbf{e}) \\ &= M \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{e - \mathbf{r}^\top \mathbf{e}}_{\text{error}} \end{aligned}$$

By Lemma 2.2.7, vector $\mathbf{r}$ is sufficiently small, and so are the errors e and $\mathbf{e}$, the above error term can be showed to be bounded by $\frac{q}{5}$. Hence it can be shown that we can recover the correct message bit, except with negligible probability.

In [41], Gentry et al. also showed the IND-CPA security of their PKE scheme, restated as follows.

Theorem 2.3.9 (Excerpted from [41]). For the chosen parameters, the PKE described in Scheme 2.3.8 satisfies the IND-CPA security defined in Definition 2.3.6, under the (n, q, χ) -LWE assumption.

2.3.3 Hierarchical Identity-based Encryption

In this section, we will recall the syntax of hierarchical identity-based encryption (HIBE) and describe one LWE-based scheme related to the later construction in the thesis.

Definition 2.3.10. An identity-based encryption $\mathcal{IBE} = (\text{Setup}, \text{Extract}, \text{Enc}, \text{Dec})$, associated with a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$, consists of four algorithms as follows.

- **Setup**(1^n), given a security parameter n , outputs a master secret key msk and public parameters pp which is an implicit input of all other algorithms.

- $\text{Extract}(\text{msk}, \text{id})$, given the master secret key msk and an identity id , generates the corresponding private key sk_{id} .
- $\text{Enc}(\text{id}, M)$, given an identity id and a message $M \in \mathcal{M}$, outputs a ciphertext $\text{ct} \in \mathcal{C}$.
- $\text{Dec}(\text{sk}_{\text{id}}, \text{ct})$, given a private key sk_{id} and a ciphertext ct , return $M \in \mathcal{M} \cup \{\perp\}$.

In a hierarchical IBE $\mathcal{HIBE} = (\text{Setup}, \text{Extract}, \text{Derive}, \text{Enc}, \text{Dec})$, identities are vectors, and there is a fifth algorithm called **Derive**, defined as follows.

- $\text{Derive}(\text{id}, \text{sk}_{\text{id}|\ell})$, given an identity $\text{id} = (\text{id}_1, \dots, \text{id}_k)$ at depth k and a private key $\text{sk}_{\text{id}|\ell}$ of a parent identity $\text{id}|\ell = (\text{id}_1, \dots, \text{id}_\ell)$ for some $\ell < k$, outputs the private key sk_{id} , the distribution of which is the same as the one outputted by $\text{Extract}(\text{msk}, \text{id})$.

Moreover, algorithm **Setup** takes an additional input which is the maximum hierarchy depth d .

Correctness. The correctness requirement for an HIBE scheme states that: for any $n \in \mathbb{N}$, any (pp, msk) outputted by $\text{Setup}(1^n, 1^d)$, any sk_{id} outputted by $\text{Extract}(\text{msk}, \text{id})$ or $\text{Derive}(\text{id}, \text{sk}_{\text{id}|\ell})$, and any message $M \in \mathcal{M}$,

$$\text{Dec}(\text{sk}_{\text{id}}, \text{Enc}(\text{id}, M)) = M$$

holds with overwhelming probability (over the randomness of all algorithms).

Security Notions. In the standard HIBE security model [18], it allows an adversary to adaptively choose the identity it wishes to attack. The selective security [25], a weaker notion of HIBE, forces the attacker to commit the target identity before seeing the public parameters. Moreover, a privacy property called indistinguishability from random was considered in [2, 3], which requires that the challenge ciphertext is indistinguishable from a random element in the ciphertext space. More formally, we

consider the security notion of indistinguishability from randomness under a selective-identity chosen plaintext attacks (INDr-sID-CPA) as in the following security game.

Security Game. Let $\mathcal{A}$ be an algorithm who would attack the system.

Init: $\mathcal{A}$ is given the maximum hierarchy depth d and outputs a target identity

$$\text{id}^* = (l_1^*, \dots, l_k^*) \text{ where } k \leq d.$$

Setup: The challenger runs $\text{Setup}(1^\lambda, 1^d)$ and gives $\mathcal{A}$ the resulting public parameters pp . It keeps the master secret key msk to itself.

Phase 1: $\mathcal{A}$ adaptively issues queries on identities $\text{id}_1, \text{id}_2, \dots$ where no query is for a prefix of id^* . For each query, the challenger runs algorithm $\text{Extract}(\text{msk}, \text{id}_i)$ to obtain the private key sk_{id_i} and sends it to $\mathcal{A}$.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs a plaintext $M \in \mathcal{M}$ on which it will target. The challenger picks $b \xleftarrow{\$} \{0, 1\}$ and $\text{ct} \xleftarrow{\$} \mathcal{C}$. If $b = 0$, it sets the challenge ciphertext to $\text{ct}^* := \text{Enc}(\text{id}^*, M)$. If $b = 1$, it sets to $\text{ct}^* := \text{ct}$. It sends ct^* as the challenge to $\mathcal{A}$.

Phase 2: $\mathcal{A}$ issues additional adaptive private key queries as in Phase 1 and the challenger responds as before.

Guess: Finally, $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$ and wins if $b' = b$.

We refer to such an algorithm $\mathcal{A}$ as an INDr-sID-CPA adversary and define its advantage in attacking an HIBE scheme $\mathcal{HIBE}$ as

$$\text{Adv}_{\mathcal{A}, \mathcal{HIBE}}^{\text{INDr-sID-CPA}}(n) = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

Definition 2.3.11 (INDr-sID-CPA Security). We say that an HIBE system $\mathcal{HIBE}$ is INDr-sID-CPA secure if we have that $\text{Adv}_{\mathcal{A}, \mathcal{HIBE}}^{\text{INDr-sID-CPA}}(n)$ is a negligible function for all INDr-sID-CPA PPT algorithms $\mathcal{A}$.

Remark 2.3.12. The adaptive-identity counterparts to the above notions by removing the Init phase from the attack game, and allowing the adversary to wait until the Challenge phase to announce the challenge identity id^* . The same restriction holds as in the selective-identity game, i.e., the adversary never issued a private key query for a prefix of id^* during Phase 1 and Phase 2. The resulting security notion is defined using the modified game as in Definition 2.3.11 and is denoted IND_r-ID-CPA.

The Agrawal-Boneh-Boyen (H)IBE Scheme

In [2], Agrawal, Boneh, and Boyen (ABB) constructed a secure LWE-based IBE system in the standard model, which could be extended to the hierarchical setting as well. In their (H)IBE system, the key authority, possessing a short basis $\mathbf{T}_\mathbf{B}$ for a public lattice $\Lambda_q^\perp(\mathbf{B})$, can delegate short bases for a super-lattice by running `SampleBasisLeft`. Each super-lattice corresponds to a depth-1 identity. With such bases, the depth-1 identities then can compute short bases for more super lattices, corresponding to depth-2 identities. The ciphertexts can be generated via a variant of the Dual-Regev PKE (see Scheme 2.3.8), which can be decrypted using a short vector in the super-lattice. Such a short vector can be sampled by running algorithm `SamplePre`. Similarly, we can deal with identities with higher hierarchical level.

In this work, we will employ the two-level variant of the ABB HIBE, described as follows.

Scheme 2.3.13 (The ABB HIBE, adapted from [2, 72]). The scheme works with following parameters: prime $q = \tilde{O}(n^{2.5})$, integer $m = 2n \lceil \log q \rceil$, Gaussian parameter $s = \tilde{O}(m)$, LWE error distribution χ with bound $B = \tilde{O}(\sqrt{n})$, and efficient encoding function $\mathbf{H} : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^{n \times n}$, that is full-rank differences (FRD)(i.e., for all distinct $\mathbf{u}, \mathbf{w} \in \mathbb{Z}_q^n$, the difference $\mathbf{H}(\mathbf{u}) - \mathbf{H}(\mathbf{w})$ is a full-rank matrix in $\mathbb{Z}_q^{n \times n}$).

Setup_{HIBE}($1^n, 1^d$): Generate $(\mathbf{B}, \mathbf{T}_\mathbf{B}) \leftarrow \text{TrapGen}(n, q, m)$. Pick $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{B}_1, \mathbf{B}_2 \xleftarrow{\$}$

$\mathbb{Z}_q^{n \times m}$. Output

$$\text{pp}_{\text{HIBE}} = (\mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{v}); \quad \text{msk}_{\text{HIBE}} = \mathbf{T}_{\mathbf{B}}.$$

Extract_{HIBE}(msk, id): For a depth-1 identity $\text{id} \in \mathbb{Z}_q^n$, return the private key sk_{id} by running

$$\text{SampleBasisLeft}(\mathbf{B}, \mathbf{B}_1 + \text{H}(\text{id})\mathbf{G}, \mathbf{T}_{\mathbf{B}}, s).$$

Derive_{HIBE}(id, sk_{id'}): For a depth-2 identity $\text{id} = (\text{id}', \text{id}'') \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$, let $\text{sk}_{\text{id}'}$ be the private key of id' and $\mathbf{B}_{\text{id}'} = [\mathbf{B} \mid \mathbf{B}_1 + \text{H}(\text{id}')\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$. Return sk_{id} by running

$$\text{SampleBasisLeft}(\mathbf{B}_{\text{id}'}, \mathbf{B}_2 + \text{H}(\text{id}'')\mathbf{G}, \text{sk}_{\text{id}'}, s).$$

Enc_{HIBE}(id, M): To encrypt a message bit $M \in \{0, 1\}$ under an identity $\text{id} = (\text{id}', \text{id}'') \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$ at depth 2, pick $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, $e \leftarrow \chi$ and $\mathbf{S}_1, \mathbf{S}_2 \xleftarrow{\$} \{-1, 1\}^{m \times m}$. Set $\mathbf{B}_{\text{id}} = [\mathbf{B} \mid \mathbf{B}_1 + \text{H}(\text{id}')\mathbf{G} \mid \mathbf{B}_2 + \text{H}(\text{id}'')\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$. Output $\text{ct} = (\mathbf{c}_1, c_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q$ where:

$$\begin{cases} \mathbf{c}_1 = \mathbf{B}_{\text{id}}^\top \mathbf{s} + (\mathbf{e} \|\mathbf{S}_1^\top \mathbf{e} \|\mathbf{S}_2^\top \mathbf{e}) \in \mathbb{Z}_q^{3m}, \\ c_0 = \mathbf{v}^\top \mathbf{s} + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q. \end{cases}$$

Dec_{HIBE}(sk_{id}, ct): Sample $\mathbf{r}_{\text{id}} \leftarrow \text{SamplePre}(\mathbf{B}_{\text{id}}, \text{sk}_{\text{id}}, \mathbf{v}, s)$. Compute $\omega = c_0 - \mathbf{r}_{\text{id}}^\top \mathbf{c}_1 \in \mathbb{Z}_q$ and output $\left\lfloor \frac{2}{q} \cdot \omega \right\rfloor \in \{0, 1\}$.

Analysis. As mentioned in [2, 41], the idea of including multi-syndromes can make the above cryptosystem to encrypt messages with polynomial length.

To demonstrate the correctness, in algorithm Dec, we have: $\mathbf{B}_{\text{id}} \mathbf{r}_{\text{id}} = \mathbf{v}$. Thus the

following holds:

$$\begin{aligned}
\omega &= c_0 - \mathbf{r}_{\text{id}}^\top \mathbf{c}_1 \\
&= \mathbf{v}^\top \mathbf{s} + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor - \mathbf{r}_{\text{id}}^\top (\mathbf{B}_{\text{id}}^\top \mathbf{s} + (\mathbf{e} \|\mathbf{S}_1^\top \mathbf{e} \|\mathbf{S}_2^\top \mathbf{e})) \\
&= M \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{e - \mathbf{r}_{\text{id}}^\top (\mathbf{e} \|\mathbf{S}_1^\top \mathbf{e} \|\mathbf{S}_2^\top \mathbf{e})}_{\text{error}}
\end{aligned}$$

Since the errors $e, \mathbf{e}$ and the vector $\mathbf{r}_{\text{id}}$ are all sufficiently small, the magnitude of above error term can be bounded by $\frac{q}{5}$. It follows that the decryption is correct, except for negligible probability.

In [2], Agrawal, Boneh and Boyen also showed that their (H)IBE scheme is **INDr-sID-CPA**) secure in the standard model. Roughly speaking, the security proof makes use of the idea of partitioning technique: the challenger could use the target identity to reduce the scheme from **LWE**, and answer private key queries for all other identities. We restate their result in Theorem 2.3.14.

Theorem 2.3.14 (Excerpted from [2]). For the chosen parameters, the HIBE in Scheme 2.3.13 satisfies the security notion of **INDr-sID-CPA** defined in 2.3.11, provided that the (n, q, χ) -**LWE** assumption holds.

2.3.4 Predicate Encryption

Here, we review the definition for predicate encryption (PE) and describe one lattice-based instantiation which will be the building block of our constructions in Chapter 4 and Chapter 5.

Definition 2.3.15. A PE scheme $\mathcal{PE} = (\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Dec})$ consists of four algorithms and has an associated predicate space $\mathbb{P}$, a message space $\mathcal{M}$ and an attribute space $\mathbb{A}$.

Setup (1^n), given a security parameter λ , outputs a master secret key $\mathbf{msk}$, and a set of public parameters $\mathbf{pp}$ which is assumed to be an implicit input for all other

algorithms.

$\text{KeyGen}(\text{msk}, f)$, given the master secret key msk and a predicate $f \in \mathbb{P}$, outputs a private key sk_f .

$\text{Enc}(a, M)$, given an attribute $a \in \mathbb{A}$ and a message $M \in \mathcal{M}$, outputs a ciphertext ct .

$\text{Dec}(\text{ct}, \text{sk}_f)$, given a ciphertext ct and a private key sk_f , outputs a message M or the distinguished symbol $\perp$.

Correctness. The correctness for PE requires that: for all msk and pp generated by $\text{Setup}(1^n)$, all $f \in \mathbb{P}$, $a \in \mathbb{A}$, all $\text{sk}_f \leftarrow \text{KeyGen}(\text{msk}, f)$ and $\text{ct} \leftarrow \text{Enc}(a, M)$:

1. If $f(a) = 1$ then $\text{Dec}(\text{ct}, \text{sk}_f) = M$.
2. If $f(a) = 0$ then $\text{Dec}(\text{ct}, \text{sk}_f) = \perp$, except for negligible probability.

Security Notions. For PE systems, there are several notions of security. The most basic one is *payload hiding*, which guarantees the privacy of the encrypted message but not the privacy of the attribute associated with a ciphertext. A stronger notion is the *attribute hiding*, which guarantees in addition the privacy of the associated attribute. We consider an intermediate notion called *weak attribute hiding*. Our security definition is “selective”, in the sense that the adversary must announce its challenge attributes before seeing the public parameters. More formally, we recall the weak attribute hiding security in a selective setting against chosen plaintext attacks (**wAH-sA-CPA**) in the following definition.

Definition 2.3.16 (wAH-sA-CPA Security). A PE scheme is wAH-sA-CPA secure if any PPT adversary $\mathcal{A}$ has negligible advantage in the following game:

1. $\mathcal{A}$ first announces the challenge attributes $a^{(0)}, a^{(1)}$.
2. $\text{Setup}(1^n)$ is run to generate msk and pp . Then $\mathcal{A}$ is given pp .

3. $\mathcal{A}$ is allowed access to oracle **KeyGen**. For a query of a predicate f , $\mathcal{A}$ is given $\text{sk}_f \leftarrow \text{KeyGen}(\text{msk}, f)$, subject to restriction that $f(a^{(0)}) = f(a^{(1)}) = 0$.
4. $\mathcal{A}$ outputs two challenge messages $M^{(0)}, M^{(1)}$ with the same length. A uniformly random bit r is chosen and $\mathcal{A}$ is given the ciphertext $\text{ct}^* \leftarrow \text{Enc}(a^{(r)}, M^{(r)})$.
5. Subject to the same restriction as before, $\mathcal{A}$ may continue to make additional queries for private keys.
6. $\mathcal{A}$ outputs a bit r' and wins the game if $r' = r$. The advantage of $\mathcal{A}$ in the game is defined as:

$$\text{Adv}_{\mathcal{A}, \mathcal{PE}}^{\text{wAH-sA-CPA}}(n) = \left| \Pr[r' = r] - \frac{1}{2} \right|.$$

The Agrawal-Freeman-Vaikuntanathan Predicate Encryption Scheme

Here, we describe the lattice-based PE scheme, constructed by Agrawal, Freeman and Vaikuntanathan (AFV) [4] in 2011 and later improved by Xagawa [102]. They consider the inner-product predicates, which means that, for chosen parameters q and ℓ , each predicate $f_{\vec{x}}$ is associated with a vector $\vec{x} \in \mathbb{Z}_q^\ell$ and each attribute a is represented as a vector $\vec{y} \in \mathbb{Z}_q^\ell$. We say that $f_{\vec{x}}(\vec{y}) = 1$ if and only if $\langle \vec{x}, \vec{y} \rangle = 0$. The set $\mathbb{P} = \{f_{\vec{x}} \mid \vec{x} \in \mathbb{Z}_q^\ell\}$ is called the predicate space, and the set $\mathbb{A} = \mathbb{Z}_q^\ell$ is called the attribute space.

The AFV scheme adopts the similar matrix extension idea from the ABB HIBE system. Specially, the master secret of the authority is a short basis $\mathbf{T}_{\mathbf{A}}$ for lattice $\Lambda_q^\perp(\mathbf{A})$, generated by running algorithm **TrapGen**. Each predicate vector $\vec{x}$ corresponds to a super-lattice of $\Lambda_q^\perp(\mathbf{A})$. With the trapdoor $\mathbf{T}_{\mathbf{A}}$, the authority can use algorithm **SampleLeft** to efficiently sample a short vector in the super-lattice. The ciphertext is computed under an attribute vector $\vec{y}$, via a variant of the Dual-Regev PKE scheme described in Scheme 2.3.8. The short vector allows to decrypt such a ciphertext when $\langle \vec{x}, \vec{y} \rangle = 0$. In order for better efficiency, in [102], Xagawa suggested an enhanced variant that takes advantages of the primitive matrix $\mathbf{G}$ (see Lemma 2.3.2).

Before describing the detailed scheme, let us discuss a small correctness issue in all existing LWE-based PE schemes [4, 38, 46, 102]. Note that the correctness of PE schemes particularly demands that if $f_{\vec{x}}(\vec{y}) = 0$ then: when decrypting a ciphertext under $\vec{y}$, the decryption algorithm, taking as input the private key for $f_{\vec{x}}$, must output a failure symbol except for negligible probability. However, in the lattice-based PKE systems (e.g., Scheme 2.3.8), the decryption algorithm actually does not fail. Instead, it outputs a random element in the message space $\mathcal{M}$. To address this issue of correctness, the following idea was suggested: assume that the scheme can be extended to work with message space $\mathcal{M}'$ satisfying that $|\mathcal{M}|/|\mathcal{M}'| = \text{negl}(n)$. In this way, when encrypting a message from the original space $\mathcal{M}$, we first encode it to a message in the larger space $\mathcal{M}'$ and then perform the encryption procedure under this encoded message. Then, if $f_{\vec{x}}(\vec{y}) = 0$, we will obtain a random element in $\mathcal{M}'$ in the decryption algorithm. Since $\mathcal{M}'$ is sufficiently large, the probability that such a random element is a proper encoding for any message in $\mathcal{M}$ is negligible. This implies the correctness of the scheme with all but negligible case.

The AFV scheme simply works with message space $\mathcal{M} = \{0, 1\}$. Based on the idea discussed above, it is reasonable to define the encoding function $\text{encode} : \mathcal{M} \rightarrow \{0, 1\}^\kappa$ for $\kappa = \omega(\log n)$ as follows. Let $\text{encode}(b) = (b, 0, \dots, 0) \in \{0, 1\}^\kappa$ for each $b \in \mathcal{M}$, which is the binary vector having b as the first coordinate and zeros elsewhere. By this definitions, function encode satisfies the required condition, since $2/2^\kappa = 2^{-\omega(\log n)}$ is negligible in n .

We will review the AFV PE together with Xagawa's improvement in the following scheme.

Scheme 2.3.17 (The AFV PE, adapted from [4, 102]). The scheme works with the following parameters: predicate and attribute vector length $\ell = \text{poly}(n)$, prime $q = \tilde{O}(\ell^2 n^4)$, integer $m = 2n \lceil \log q \rceil$, integer $\kappa = \omega(\log n)$, Gaussian parameter $s = \tilde{O}(m)$, LWE error distribution χ with bound $B = \tilde{O}(\sqrt{n})$ and encoding function $\text{encode} : \{0, 1\} \rightarrow \{0, 1\}^\kappa$ defined as above.

Setup(1^n): Generate $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \text{TrapGen}(n, q, m)$. Pick $\mathbf{V} \xleftarrow{\$} \mathbb{Z}_q^{n \times \kappa}$ and for each $i \in [\ell]$, sample $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$. Output

$$\text{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{V}); \quad \text{msk} = \mathbf{T}_\mathbf{A}.$$

KeyGen($\text{msk}, \vec{x}$): For a predicate vector $\vec{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_q^\ell$, compute $\mathbf{A}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{A}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \in \mathbb{Z}_q^{n \times m}$ and output $\text{sk}_{\vec{x}} = \mathbf{Z}$ by running

$$\text{SampleLeft}(\mathbf{A}, \mathbf{A}_{\vec{x}}, \mathbf{T}_\mathbf{A}, \mathbf{V}, s).$$

Another way is to set $\text{sk}_{\vec{x}} = \mathbf{T}_{\vec{x}}$ by running $\text{SampleBasisLeft}(\mathbf{A}, \mathbf{A}_{\vec{x}}, \mathbf{T}_\mathbf{A}, s)$.

Note that we then can sample

$$\mathbf{Z} \leftarrow \text{SamplePre}([\mathbf{A} \mid \mathbf{A}_{\vec{x}}], \mathbf{T}_{\vec{x}}, \mathbf{V}, s).$$

Enc($\vec{y}, M$): To encrypt a bit $M \in \mathcal{M}$ under an attribute vector $\vec{y} = (y_1, \dots, y_\ell) \in \mathbb{Z}_q^\ell$, choose $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \xleftarrow{\chi^\kappa}$, $\mathbf{e}_1 \xleftarrow{\chi^m}$, $\mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for each $i \in [\ell]$, and then output the ciphertext $\text{ct} = (\mathbf{c}, \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]})$, where:

$$\begin{cases} \mathbf{c} = \mathbf{V}^\top \mathbf{s} + \mathbf{e} + \text{encode}(M) \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_0 = \mathbf{A}^\top \mathbf{s} + \mathbf{e}_1 \in \mathbb{Z}_q^m, \\ \mathbf{c}_i = (\mathbf{A}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{R}_i^\top \mathbf{e}_1 \in \mathbb{Z}_q^m, \forall i \in [\ell] \end{cases}$$

Dec(ct, sk_f): Set $\mathbf{c}_{\vec{x}} = \sum_{i=1}^{\ell} (\mathbf{G}^{-1}(x_i \cdot \mathbf{G}))^\top \mathbf{c}_i \in \mathbb{Z}_q^m$ and $\mathbf{d} = \mathbf{c} - \mathbf{Z}^\top (\mathbf{c}_0 \parallel \mathbf{c}_{\vec{x}}) \in \mathbb{Z}_q$. If for some $M' \in \mathcal{M}$, it holds that $\text{encode}(M') = \left\lfloor \frac{2}{q} \cdot \mathbf{d} \right\rfloor$, then output M' . Otherwise, output $\perp$.

Analysis. To demonstrate the correctness, if the scheme is operated as specified, we consider two cases as follows.

1. Case 1: Suppose that $\langle \vec{x}, \vec{y} \rangle = 0$. One then has:

$$\begin{aligned} \mathbf{c}_{\vec{x}} &= \sum_{i=1}^{\ell} \left(\mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \right)^{\top} \mathbf{c}_i \\ &= \sum_{i=1}^{\ell} \left(\mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \right)^{\top} \left((\mathbf{A}_i + y_i \cdot \mathbf{G})^{\top} \mathbf{s} + \mathbf{R}_i^{\top} \mathbf{e}_1 \right) \\ &= (\mathbf{A}_{\vec{x}})^{\top} \mathbf{s} + (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1 \end{aligned}$$

where $\mathbf{R}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{R}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$. One also has $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} = \mathbf{V}$. Thus the following holds:

$$\begin{aligned} \mathbf{d} &= \mathbf{c} - \mathbf{Z}^{\top}(\mathbf{c}_0 \parallel \mathbf{c}_{\vec{x}}) \\ &= \mathbf{V}^{\top} \mathbf{s} + \mathbf{e} + \text{encode}(M) \cdot \left\lfloor \frac{q}{2} \right\rfloor - \mathbf{Z}^{\top} \left([\mathbf{A} \mid \mathbf{A}_{\vec{x}}]^{\top} \mathbf{s} + (\mathbf{e}_1 \parallel (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1) \right) \\ &= \text{encode}(M) \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{\mathbf{e} - \mathbf{Z}^{\top}(\mathbf{e}_1 \parallel (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1)}_{\text{error}} \end{aligned}$$

Since the error $\mathbf{e}, \mathbf{e}_1$ and entries of $\mathbf{Z}$ are all bounded by $O(s\sqrt{n})$ with overwhelming probability, the above error term is bounded by $\frac{q}{5}$.

2. Case 2: Suppose that $\langle \vec{x}, \vec{y} \rangle \neq 0$. Then vector $\mathbf{c}_{\vec{x}}$ contains the non-zero term

$$\langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}^{\top} \mathbf{s}$$

which appears in vector $\mathbf{d}$ as well. By Lemma 2.3.4, the distribution of such a term is indistinguishable from uniform. It implies that the probability that all the last $\kappa - 1$ entries of vector $\left\lfloor \frac{2}{q} \cdot \mathbf{d} \right\rfloor$ are zeros is at most $2/2^{\kappa} = 2^{-\omega(\log n)} = \text{negl}(n)$.

Following the discussion above, the PE in Scheme 2.3.17 is correct, except for negligible probability.

Agrawal, Freeman and Vaikuntanathan proved the (wAH-sA-CPA) security for that their PE scheme, and Xagawa showed that the same assertion holds for his improved version. We thus have the following theorem.

Theorem 2.3.18 (Excerpted from [4,102]). For the given parameters, the PE described in Scheme 2.3.17 satisfies the **wAH-sA-CPA** security notion defined in Definition 2.3.16, based on the hardness of the (n, q, χ) -LWE problem.

2.4 Revocable Cryptosystems

In 2001, Naor, Naor, and Lotspiech [75] introduced the subset-cover framework to handle stateless user revocation in broadcast encryption, which is arguably one of the most well-studied and efficient revocation techniques for public key cryptosystems. In their framework, when certain users are revoked from the system, a set of disjoint subsets is generated to cover all the non-revoked users. Following this framework, they proposed two tree-based methods: complete subtree (CS) method and subset difference (SD) method. The former enjoys a smaller key size while the latter has a smaller ciphertext size.

In this thesis, we will consider to employ the CS method to realize efficient revocation, which was widely applied in many cryptographic primitives, such as revocable IBE [15, 54, 63, 96], revocable ABE [10, 92] and revocable group signature [60, 61]. Next, we will describe how the CS method works and review one **LWE**-based revocable IBE using the CS method, given in the work by Chen et al. [28].

2.4.1 The Complete Subtree Method

The CS method is built on a binary-tree **BT** with number of leaf being at least N : the number of users (or keys) that is supported by the system. It follows that each user (or key) can be assigned to a unique leaf node. On the tree **BT**, we use the following notations:

1. For a non-leaf node θ , notation θ_ℓ is used to denote the left child of θ and θ_r is used to denote the corresponding right child.

Figure 2.1: Suppose that $RL = \{3, 4\}$. It follows that $\{9, 14\} \leftarrow \text{KUNodes}(\text{BT}, RL)$. For leaf node $2 \notin RL$, the set $\text{Path}(2) = \{\text{root}, 13, 9, 2\}$ is intersected with $\text{KUNode}(\text{BT}, RL)$ at node 9. For leaf node $4 \in RL$, no ancestor of it is contained in the output of algorithm **KUNodes**.

2. For a leaf node ν , the set $\text{Path}(\nu)$ represents the collection of nodes on the path from ν to the root node (both the root and ν are included).

The CS method then is based on a node selection algorithm **KUNodes**, which takes as input the binary tree **BT** and a list of revoked users or keys **RL**, and outputs a set of nodes Y . Algorithm **KUNodes** is described in the following and an illustration example is given in Figure 2.1.

```

KUNodes(BT, RL)
 $X, Y \leftarrow \emptyset$ 
 $\forall \nu \in RL : \text{add Path}(\nu) \text{ to } X$ 
 $\forall \theta \in X :$ 
    If  $\theta_\ell \notin X$ , then add  $\theta_\ell$  to  $Y$ 
    If  $\theta_r \notin X$ , then add  $\theta_r$  to  $Y$ 
    If  $Y = \emptyset$ , then add root to  $Y$ 
Return  $Y$ 

```

Note that the outputted set Y is the smallest subset of nodes that contains an ancestor of all the leafs corresponding to non-revoked users (or keys). In [75], it is showed that the size of the subset generated by **KUNodes** is bounded by $r \log \frac{N}{r}$, where r stands for the size of **RL**.

The CS method is employed as one building block of our schemes in Chapter 3, Chapter 4, and Chapter 5. Particularly, in Chapter 3, Chapter 4, the revocation list is associated with a time period t and algorithm **KUNodes** additionally takes as input a time slot t .

2.4.2 Revocable Identity-based Encryption

The syntax of RIBE was formally defined by Boldyreva et al. [15].

Definition 2.4.1. Associated with a message space $\mathcal{M}$, a revocable identity-based encryption $\mathcal{RIBE} = (\text{Setup}, \text{PrivKG}, \text{UpdKG}, \text{DecKG}, \text{Enc}, \text{Dec}, \text{Revoke})$, consists of seven algorithms as follows.

- $\text{Setup}(1^n)$, given a security parameter n , outputs a master secret key msk and public parameters, the initial revocation list $\text{RL} := \emptyset$, and a state st . Here, pp is an implicit input of all other algorithms.
- $\text{PrivKG}(\text{msk}, \text{id}, \text{st})$, given msk , an identity id , and st , generates the private key sk_{id} and updates st .
- $\text{UpdKG}(\text{msk}, t, \text{RL}, \text{st})$, given msk , a key update time t with the corresponding revocation list RL , and st , generates an update key uk_t .
- $\text{DecKG}(\text{sk}_{\text{id}}, \text{uk}_t)$, given a private key sk_{id} and an update key uk_t , generates a decryption key $\text{dk}_{\text{id},t}$.
- $\text{Enc}(\text{id}, t, M)$, given an id , a t , and a message $M \in \mathcal{M}$, outputs a ciphertext ct .
- $\text{Dec}(\text{dk}_{\text{id},t}, \text{ct})$, given a private key sk_{id} and a ciphertext ct , return $M \in \mathcal{M} \cup \{\perp\}$.
- $\text{Revoke}(\text{id}, t, \text{RL})$, given an id to be revoked at a time t , and the current RL , returns an updated RL .

Correctness. The correctness condition for an RIBE scheme demands that: for any $n \in \mathbb{N}$, all possible state $\mathbf{st}$, and revocation list $\mathbf{RL}$, if $\mathbf{id}$ is not revoked at a time $\mathbf{t}$, then

$$\text{Dec}(\text{dk}_{\mathbf{id}, \mathbf{t}}, \text{Enc}(\mathbf{id}, \mathbf{t}, M)) = M$$

holds except for negligible probability.

Security Notions. Boldyreva et al. [15] define the selective security for RIBE, which takes account the privacy challenges in establishing revocation. More formally, we recall the security notion of indistinguishability from randomness under a selective-identity chosen plaintext attacks (INDr-sRID-CPA) as follows.

Definition 2.4.2 (INDr-sRID-CPA Security). Let $\mathcal{A}$ be an adversary algorithm and let $\mathcal{O}$ be a set of oracles defined as follows:

- $\text{PrivKG}(\cdot)$: When querying on an identity $\mathbf{id}$, algorithm $\mathcal{A}$ is given a private key $\mathbf{sk}_{\mathbf{id}}$ by running $\text{PrivKG}(\mathbf{msk}, \mathbf{id}, \mathbf{st})$.
- $\text{UpdKG}(\cdot)$: When querying on a time period $\mathbf{t}$, algorithm $\mathcal{A}$ is given an update key $\mathbf{uk}_{\mathbf{t}}$ by running algorithm $\text{UpdKG}(\mathbf{msk}, \mathbf{t}, \mathbf{RL}, \mathbf{st})$.
- $\text{Revoke}(\cdot, \cdot)$: When querying on an identity $\mathbf{id}$ and a time $\mathbf{t}$, algorithm $\mathcal{A}$ is given the updated $\mathbf{RL}$ by running $\text{Revoke}(\mathbf{id}, \mathbf{t}, \mathbf{RL}, \mathbf{st})$. Note that this oracle cannot be queried on time $\mathbf{t}$ if $\text{UpdKG}(\cdot)$ has been queried on time $\mathbf{t}$.

An RIBE scheme is INDr-sRID-CPA secure if any PPT adversary $\mathcal{A}$ has negligible advantage in the following experiment:

$$\boxed{\text{Exp}_{\mathcal{A}, \mathcal{RIBE}}^{\text{INDr-sRID-CPA}}}(n)$$

$$\text{id}^*, \mathbf{t}^* \leftarrow \mathcal{A}$$

$$(\text{pp}, \text{msk}, \text{RL}, \text{st}) \leftarrow \text{Setup}(1^n)$$

$$M_0, M_1 \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp})$$

$$r \xleftarrow{\$} \{0, 1\}$$

$$\text{ct}^* \leftarrow \text{Enc}(\text{id}^*, \mathbf{t}^*, M_b)$$

$$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\text{ct}^*)$$

Return 1 if $b' = b$ and 0 otherwise.

Except the requirement that the two challenge messages M_0, M_1 have the same length, the following two restrictions must hold:

1. Case 1: if the challenge identity id^* has been queried to $\text{PrivKG}(\cdot)$, then id^* must be revoked at the challenge time $\mathbf{t}^*$.
2. Case 2: if id^* is not revoked at $\mathbf{t}^*$, then id^* should not be queried to the $\text{PrivKG}(\cdot)$ oracle.

The advantage of $\mathcal{A}$ in the experiment is defined as:

$$\text{Adv}_{\mathcal{A}, \mathcal{RIBE}}^{\text{INDr-sRID-CPA}}(n) = \left| \Pr \left[\text{Exp}_{\mathcal{A}, \mathcal{RIBE}}^{\text{INDr-sRID-CPA}} = 1 \right] - \frac{1}{2} \right|.$$

Remark 2.4.3. In [63], the adaptive security for RIBE was investigated, where the adversary is not required to specify id^* and $\mathbf{t}^*$ before seeing the public parameters pp . In [96], the resistance of decryption key exposure attacks was considered, where the adversary is allowed additional access to oracle DecKG .

Chen et al.'s RIBE

In [28], Chen et al. proposed an LWE-based RIBE based on the CS method that employs the ABB IBE (see Scheme 2.3.13) as a building block. Their RIBE scheme employs another instance of the ABB IBE to handle the key update time. To apply the CS method, for each node θ , the public syndrome $\mathbf{u} \in \mathbb{Z}_q^n$ is split into two random vectors $\mathbf{u}_{1,\theta}, \mathbf{u}_{2,\theta}$. Next, we describe a variant of Chen et al.'s RIBE scheme encrypting multi-bit messages instead of a bit.

Scheme 2.4.4 (Chen et al.'s RIBE, adapted from [28]). The scheme works with following parameters: maximal number of identities $N = \text{poly}(n)$, prime $q = \tilde{O}(n^{2.5})$, integer $m = 2n \lceil \log q \rceil$, message length $k = \text{poly}(n)$, Gaussian parameter $s = \tilde{O}(m)$, LWE error distribution χ with bound $B = \tilde{O}(\sqrt{n})$, and FRD function $\mathbf{H} : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^{n \times n}$.

Setup(1^n): Generate $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \text{TrapGen}(n, q, m)$. Pick $\mathbf{A}_1, \mathbf{A}_2 \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$. Obtain the initial revocation list $\text{RL} := \emptyset$ and initial state $\text{st} := \text{BT}$ where BT is a binary tree with at least N leaf nodes. Return

$$\text{RL}; \text{st}; \text{pp}_{\text{RIBE}} = (\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{U}); \text{msk}_{\text{RIBE}} = \mathbf{T}_\mathbf{A}.$$

PrivKG($\text{msk}, \text{id}, \text{st}$): Randomly assign id to an unassigned leaf node ν_{id} of BT . For each node $\theta \in \text{Path}(\nu_{\text{id}})$, if $\mathbf{U}_{1,\theta}$ and $\mathbf{U}_{2,\theta}$ are undefined, then choose $\mathbf{U}_{1,\theta} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$ and let $\mathbf{U}_{2,\theta} = \mathbf{U} - \mathbf{U}_{1,\theta}$. Output $\text{sk}_{\text{id}} = (\theta, \mathbf{E}_{1,\theta})_{\theta \in \text{Path}(\nu_{\text{id}})}$ by running

$$\mathbf{E}_{1,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_1 + \mathbf{H}(\text{id})\mathbf{G}, \mathbf{T}_\mathbf{A}, \mathbf{U}_{1,\theta}, s).$$

UpdKG($\text{msk}, \text{t}, \text{RL}, \text{st}$): For each node $\theta \in \text{KUNodes}(\text{BT}, \text{RL})$, obtain $\mathbf{U}_{2,\theta}$ included in st . Return $\text{uk}_{\text{t}} = (\theta, \mathbf{E}_{2,\theta})_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})}$ by running

$$\mathbf{E}_{2,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_2 + \mathbf{H}(\text{t})\mathbf{G}, \mathbf{T}_\mathbf{A}, \mathbf{U}_{2,\theta}, s).$$

DeckG(sk_{id}, uk_t): If there exists $\theta \in \text{Path}(\nu_{\text{id}}) \cap \text{KUNodes}(\text{BT}, \text{RL})$, then return the corresponding pair $(\mathbf{E}_{1,\theta}, \mathbf{E}_{2,\theta})$. Otherwise, return $\perp$.

Enc(id, t, M): To encrypt a k -bit message $M \in \{0, 1\}^k$ under an identity id and a t time, pick $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, $\mathbf{e}' \leftarrow \chi^k$ and $\mathbf{S}_1, \mathbf{S}_2 \xleftarrow{\$} \{-1, 1\}^{m \times m}$. Set $\mathbf{A}_{\text{id}} = [\mathbf{A} \mid \mathbf{A}_1 + \text{H}(\text{id})\mathbf{G} \mid \mathbf{A}_2 + \text{H}(\text{t})\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$. Output $\mathbf{ct} = (\mathbf{c}_1, \mathbf{c}_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$ where:

$$\begin{cases} \mathbf{c}_1 = \mathbf{A}_{\text{id}, \text{t}}^\top \mathbf{s} + (\mathbf{e} \parallel \mathbf{S}_1^\top \mathbf{e} \parallel \mathbf{S}_2^\top \mathbf{e}) \in \mathbb{Z}_q^{3m} \\ \mathbf{c}_0 = \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + M \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q^k. \end{cases}$$

Dec(dk_{id,t}, ct): Compute $\mathbf{d} = \mathbf{c}_0 - [\mathbf{E}_{1,\theta} \mid \mathbf{E}_{2,\theta}]^\top \mathbf{c}_1 \in \mathbb{Z}_q^k$ and output $\left\lfloor \frac{2}{q} \cdot \mathbf{d} \right\rfloor \in \{0, 1\}^k$.

Revoke(id, t, RL): Update RL by adding all nodes associated with id to RL.

Analysis. To demonstrate the correctness, if id is not revoked at t, in algorithm DeckG, we have:

$$[\mathbf{A} \mid \mathbf{A}_1 + \text{H}(\text{id})\mathbf{G}] \cdot \mathbf{E}_{1,\theta} + [\mathbf{A} \mid \mathbf{A}_2 + \text{H}(\text{t})\mathbf{G}] \cdot \mathbf{E}_{2,\theta} = \mathbf{U},$$

Thus the following holds:

$$\begin{aligned} \mathbf{d} &= \mathbf{c}_0 - [\mathbf{E}_{1,\theta} \mid \mathbf{E}_{2,\theta}]^\top \mathbf{c}_1 \\ &= \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + M \cdot \left\lfloor \frac{q}{2} \right\rfloor - [\mathbf{E}_{1,\theta} \mid \mathbf{E}_{2,\theta}]^\top (\mathbf{A}_{\text{id}}^\top \mathbf{s} + (\mathbf{e} \parallel \mathbf{S}_1^\top \mathbf{e} \parallel \mathbf{S}_2^\top \mathbf{e})) \\ &= M \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{\mathbf{e}' - [\mathbf{E}_{1,\theta} \mid \mathbf{E}_{2,\theta}]^\top (\mathbf{e} \parallel \mathbf{S}_1^\top \mathbf{e} \parallel \mathbf{S}_2^\top \mathbf{e})}_{\text{error}} \end{aligned}$$

Since the errors $\mathbf{e}, \mathbf{e}'$ and the matrices $\mathbf{E}_{1,\theta}, \mathbf{E}_{2,\theta}$ are all sufficiently small, the magnitude of above error term can be bounded by $\frac{q}{5}$. It follows that the decryption is correct, except for negligible probability.

Chen et al. also showed that one-bit variant of their RIBE scheme is IND-sRID-CPA secure. The security proof can be easily adapted to deal with the k -bit case based on the techniques from [2, 41]. We thus have the following theorem.

Theorem 2.4.5 (Excerpted from [28]). For the chosen parameters, Chem et al.'s RIBE described in Scheme 2.4.4 satisfies the security notion of IND-sRID-CPA defined in Definition 2.4.2, based on the hardness of the (n, q, χ) -LWE problem.

Chapter 3

Server-aided Revocable Identity-based Encryption from Lattices

3.1 Introduction

Introduced by Shamir [98], Identity-Based Encryption (IBE) provides an important alternative way to avoid the need for a Public Key Infrastructure (PKI). It allows a sender to use the recipient's identity as a public key to encrypt a message, from which the corresponding private key is issued through a secret channel by a trusted authority called the key generation center (KGC). The first realization of IBE was the work by Boneh and Franklin [18] based on the Bilinear Diffie-Hellman problem. Almost at the same time, Cocks [30] proposed a scheme using quadratic residues modulo a composite. The third class of IBE, pioneered by Gentry et al. [41] in 2008, is based on lattice assumptions.

As for many multi-user cryptosystems, an efficient revocation mechanism is necessary and imperative in the IBE setting. If some identities have been revoked due

to certain reasons (e.g., the user misbehaves or his private key is stolen), the mechanism should ensure that: (i) the revoked identities no longer possess the decryption capability; (ii) the workloads of the KGC and the non-revoked users in updating the system are “small”. Designing an IBE scheme supported by efficient revocation turned out to be a challenging problem. A naïve solution, suggested by Boneh and Franklin in their seminal work [18], requires users to periodically renew their private keys by communicating with the KGC per time epoch, via a secure channel. This solution, while yielding a straightforward revocation method (i.e., revoked identities are not given new keys), is too impractical to be used for large-scale system, as the workload of the KGC grows linearly in the number of users N . Later on, Boldyreva, Goyal and Kumar (BGK) [15] formally defined the notion of revocable identity-based encryption (RIBE), and employed the tree-based revocation techniques from [75] to construct the first scalable RIBE in which the KGC’s workload is only logarithmic in N . In the BGK model, however, the non-revoked users have to communicate with the KGC regularly to receive the update keys. Although this key updating process can be done through a public channel, it is somewhat inconvenient and bandwidth-consuming.

To improve the situation, Qin et al. [89] recently proposed server-aided revocable identity-based encryption (SR-IBE) - a new revocation approach in which almost all workloads on users are outsourced to a server, and users can compute decryption keys at any time period without having to communicate with either the KGC or the server. Moreover, the server can be untrusted (in the sense that it does not possess any secret information) and should just perform correct computations. More specifically, an SR-IBE scheme functions as follows. When setting up the system, the KGC issues a long-term private key to each user. The update keys are sent only to the server (via a public channel) rather than to all users. The ciphertexts also go through the server who transforms them to “partially decrypted ciphertexts” which are forwarded to the intended recipients. The latter then can recover the messages using decryption keys derived from their long-term keys. This is particularly well-suited for applications

such as secure email systems, where email addresses represent users' identities and the (untrusted) email server performs most of the computations. In [89], apart from introducing this new model, Qin et al. also described a pairing-based instantiation of SR-IBE.

In this work, inspired by the advantages and potentials of SR-IBE, we put it into the world of lattice-based cryptography, and design the first SR-IBE scheme from lattice assumptions.

Related Works. The subset cover framework, originally proposed by Naor, Naor and Lotspiech (NNL) [75] in the context of broadcast encryption, is arguably the most well-known revocation technique for multi-user systems. It uses a binary tree, each leaf of which is designated to each user. Non-revoked users are partitioned into disjoint subsets, and are assigned keys according to the Complete Subtree (CS) method or the Subset Difference (SD) method. This framework was first considered in the IBE setting by Boldyreva et al. [15]. Subsequently, several pairing-based RIBE schemes [54, 63, 96] were proposed, providing various improvements. Among them, the work by Seo and Emura [96] suggested a strong security notion for RIBE, that takes into account the threat of decryption key exposure attacks. The NNL framework also found applications in the context of revocable group signatures [60, 61].

The study of IBE with outsourced revocation was initiated by Li et al. [58], who introduced a method to outsource the key update workload of the trusted KGC to a semi-trusted KGC. Indeed, revocation mechanisms with an online semi-trusted third party (called mediator) had appeared in earlier works [13, 17, 33, 62]. However, all these approaches are vulnerable against collusion attacks between revoked users and the semi-trusted KGC or the mediator.

Our Results and Techniques. We introduce the first construction of lattice-based SR-IBE. We inherit the main efficiency advantage of Qin et al.'s model over the BGK model for RIBE: the system users do not have to communicate with any party to get

		Public Params. Size	Token Size	Private Key Size	Update Key Size	Ciphertext Size	Model
	[28]	$\tilde{O}(n^2)$	–	$O(\log N) \cdot \tilde{O}(n)$	$r \log \frac{N}{r} \cdot \tilde{O}(n)$	$\tilde{O}(n)$	Selective
	[29]	$\tilde{O}(n^{2+\epsilon})$	–	$O(\log^2 N) \cdot \tilde{O}(n)$	$(2r-1) \cdot \tilde{O}(n)$	$\tilde{O}(n^{1+\epsilon})$	Adaptive
Ours	Server	$\tilde{O}(n^2)$	$O(\log N) \cdot \tilde{O}(n)$	–	$r \log \frac{N}{r} \cdot \tilde{O}(n)$	$\tilde{O}(n)$	Selective
	User		–	$\tilde{O}(n^2)$	–	$\tilde{O}(n)$	

Table 3.1: Comparison among known lattice-based revocable IBE schemes. Here, n is the security parameter, N is the maximum number of users, r is the number of revoked users. For the scheme from [29], the number ϵ is a small constant such that $\epsilon < 1/2$. The notation “–” means that such an item does not exist in the corresponding scheme.

update keys, as they are capable of computing decryption keys for any time period on their own. As for previous lattice-based RIBE schemes [28, 29], our proposal works with one-bit messages, but multi-bit variants can be achieved with small overhead, using standard techniques [2, 41]. The public parameters and the ciphertexts produced by the scheme have bit-sizes comparable to those of [28, 29]. The long-term private key of each user has size constant in the number of all users N , but to enable the delegation of decryption keys, it has to be a trapdoor matrix with relatively large size. The full efficiency comparison among the schemes from [28, 29] and ours is given in Table 3.1.

As a high level, our design approach is similar to the pairing-based instantiation by Qin et al., in the sense that we also employ an RIBE scheme [28] and a two-level HIBE scheme [2] as the building blocks. In our setting, the server simultaneously plays two roles: it is the decryptor in the RIBE block (i.e., it receives ciphertexts from senders and performs the decryption mechanism of RIBE - which is called “partial decryption” here), and at the same time, it is the sender in the HIBE block. The users (i.e., the message recipients), on the other hand, only work with the HIBE block. Their identities are placed at the first level of the hierarchy, while the time periods are put at the second level. This enables the user with private key for id to delegate a decryption key for an ordered pair of the form (id, t) .

However, looking into the details, it is not straightforward to make the two building blocks operate together. Qin et al. address this problem by using a key splitting

technique which currently seems not available in the lattice setting. Instead, we adapt a double encryption mechanism, recently employed by Libert et al. [59] in the context of lattice-based group signatures with message-dependent opening [94], which works as follows. The sender encrypts the message under the HIBE to obtain an initial ciphertext of the form $(\mathbf{c}_2, c_0)$, where c_0 is an element of $\mathbb{Z}_q$ (for some $q > 2$) and is the ciphertext component carrying the message information. Next, he encrypts the binary representation of c_0 , i.e., vector $\text{bin}(c_0) \in \{0, 1\}^{\lceil \log q \rceil}$, under the RIBE to obtain $(\mathbf{c}_1, \hat{\mathbf{c}}_0)$. The final ciphertext is then set as $(\mathbf{c}_1, \mathbf{c}_2, \hat{\mathbf{c}}_0)$ and is sent to the server. The latter will invert the second step of the encryption mechanism to get back to the initial ciphertext $(\mathbf{c}_2, c_0)$. Receiving $(\mathbf{c}_2, c_0)$ from the server, the user should be able to recover the message.

The security of our SR-IBE scheme relies on that of the two lattice-based building blocks, i.e., the ABB HIBE (described in Scheme 2.3.13) and Chen et al.'s RIBE (described in Scheme 2.4.4). Both of them are selectively secure in the standard model, assuming the hardness of the Learning with Errors (LWE) problem - so is our scheme.

Organization. The rest of this chapter is organized as follows. Section 3.2 provides definitions of SR-IBE. Our construction of lattice-based SR-IBE and its analysis are presented in Section 3.3.

3.2 Definitions

In this section, we recall the definition and security model of SR-IBE, put forward by Qin et al. [89].

Definition 3.2.1. A server-aided revocable identity-based encryption scheme $\mathcal{SRIBE}$ = (Sys, Setup, Token, UpdKG, TranKG, PrivKG, DeckKG, Enc, Transform, Dec, Revoke) involves 4 parties: KGC, sender, recipient, and server. Algorithms among the parties are as follows:

$\text{Sys}(1^n)$ is run by the KGC. It takes as input a security parameter n and outputs the system parameters **params**.

$\text{Setup}(1^n)$ is run by the KGC. It takes as input the system parameters **params** and outputs a set of public parameters **pp**, a master secret key **msk**, a revocation list **RL** (initially empty), and state **st**. We assume that **pp** is an implicit input of all other algorithms.

$\text{Token}(\text{msk}, \text{id}, \text{st})$ is run by the KGC. It takes as input the master secret key **msk**, an identity **id**, and state **st**. It outputs an updated state **st** and a token τ_{id} for **id**. The latter is sent to the server through a public channel.

$\text{UpdKG}(\text{msk}, t, \text{RL}, \text{st})$ is run by the KGC. It takes as input the master secret key **msk**, a time t , the up-to-date revocation list **RL**, and state **st**. It outputs an update key uk_t , which is sent to the server through a public channel.

$\text{TranKG}(\tau_{\text{id}}, \text{uk}_t)$ is run by the server. It takes as input a token τ_{id} and an update key uk_t , and outputs a transformation key $\text{tk}_{\text{id},t}$. The transform key $\text{tk}_{\text{id},t}$ is stored by the sever itself.

$\text{PrivKG}(\text{msk}, \text{id})$ is run by the KGC. It takes as input the master key **msk** and an identity **id**, and outputs a private key sk_{id} of **id**, which is sent to the recipient through a *secret* channel.

$\text{DecKG}(\text{sk}_{\text{id}}, t)$ is run by the recipient. It takes as input the private key sk_{id} and a time t . It outputs a decryption key $\text{dk}_{\text{id},t}$, which is reserved by the recipient himself.

$\text{Enc}(\text{id}, t, M)$ is run by the sender. It takes as input a recipient's identity **id**, a time t , and a message M . It outputs a ciphertext $\text{ct}_{\text{id},t}$, which is sent to the server through a public channel.

$\text{Transform}(\text{ct}_{\text{id},t}, \text{tk}_{\text{id},t})$ is run by the sever. It takes as input a ciphertext $\text{ct}_{\text{id},t}$, and a transformation key $\text{tk}_{\text{id},t}$ with the same **id**, t . It outputs a partially decrypted

ciphertext $\text{ct}'_{\text{id},\text{t}}$, which is sent to the recipient through a public channel.

$\text{Dec}(\text{ct}_{\text{id},\text{t}}, \text{dk}_{\text{id},\text{t}})$ is run by the recipient. On input a partially decrypted ciphertext $\text{ct}'_{\text{id},\text{t}}$ and a decryption key $\text{dk}_{\text{id},\text{t}}$, this algorithm outputs a message M or the distinguished symbol $\perp$.

$\text{Revoke}(\text{id}, \text{t}, \text{RL}, \text{st})$ is run by the KGC. It takes as input an identity id to be revoked, a revocation time t , the up-to-date revocation list RL , and a state st . It outputs an updated revocation list RL .

Correctness. For an SR-IBE scheme, it requires that: For any $n \in \mathbb{N}$, all possible state st , and any revocation list RL , if id is not revoked on a time t , and if all parties follow the prescribed algorithms, then

$$\text{Dec}(\text{ct}_{\text{id},\text{t}}, \text{dk}_{\text{id},\text{t}}) = M.$$

Security. Qin et al. [89] defined semantic security against adaptive-identity chosen plaintext attacks for SR-IBE. Here, we will consider the weaker selective-identity security suggested by Boldyreva et al. [15], in which the adversary must announce the challenge identity id^* and time t^* before the execution of algorithm **Setup**.

Definition 3.2.2 (SR-sID-CPA Security). Let $\mathcal{O}$ be the set of the following oracles:

1. $\text{Token}(\cdot)$: On input an identity id , return π_{id} by running $\text{Token}(\text{msk}, \text{id}, \text{st})$.
2. $\text{UpdKG}(\cdot)$: On input a time t , return uk_{t} by running $\text{UpdKG}(\text{msk}, \text{t}, \text{RL}, \text{st})$.
3. $\text{PrivKG}(\cdot)$: On input an identity id , return sk_{id} by running $\text{PrivKG}(\text{msk}, \text{id})$.
4. $\text{DecKG}(\cdot, \cdot)$: On input an identity id and a time t , return $\text{dk}_{\text{id},\text{t}}$ by running the decryption key $\text{DecKG}(\text{sk}_{\text{id}}, \text{t})$, where sk_{id} is from $\text{PrivKG}(\text{msk}, \text{id})$.
5. $\text{Revoke}(\cdot, \cdot)$: On input an identity id and a time t , update the revocation list RL by running $\text{Revoke}(\text{id}, \text{t}, \text{RL}, \text{st})$.

An SR-IBE $\mathcal{SRIBE}$ scheme is selectively secure against chosen plaintext attack (SR-sID-CPA) if any PPT adversary $\mathcal{A}$ has negligible advantage in the following experiment:

$\text{Exp}_{\mathcal{A}, \mathcal{SRIBE}}^{\text{SR-sID-CPA}}(n)$

$\text{params} \leftarrow \text{Sys}(1^n); \text{id}^*, \text{t}^* \leftarrow \mathcal{A}$
 $(\text{pp}, \text{msk}, \text{st}, \text{RL}) \leftarrow \text{Setup}(\text{params})$
 $M_0, M_1 \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp})$
 $b \xleftarrow{\$} \{0, 1\}$
 $\text{ct}_{\text{id}^*, \text{t}^*} \leftarrow \text{Enc}(\text{id}^*, \text{t}^*, M_r)$
 $b' \leftarrow \mathcal{A}^{\mathcal{O}}(\text{ct}_{\text{id}^*, \text{t}^*})$
 Return 1 if $b' = b$ and 0 otherwise.

Beyond the condition that M_0, M_1 have the same length, the following restrictions must hold:

1. $\text{UpdKG}(\cdot)$ and $\text{Revoke}(\cdot, \cdot)$ can only be queried in non-decreasing order of time.
2. $\text{Revoke}(\cdot, \cdot)$ can not be queried on time $\mathbf{t}$ if $\text{UpdKG}(\cdot)$ has been queried on time $\mathbf{t}$.
3. If $\text{PrivKG}(\cdot)$ was queried on the challenge identity id^* , then $\text{Revoke}(\cdot, \cdot)$ must be queried on $(\text{id}^*, \mathbf{t})$ for any $\mathbf{t} \leq \text{t}^*$.
4. If identity id^* is non-revoked at time t^* , then $\text{DecKG}(\cdot, \cdot)$ can not be queried on $(\text{id}^*, \text{t}^*)$.

The advantage of $\mathcal{A}$ in the above experiment is defined as:

$$\text{Adv}_{\mathcal{A}, \mathcal{SRIBE}}^{\text{SR-sID-CPA}}(n) = \left| \Pr \left[\text{Exp}_{\mathcal{A}, \mathcal{SRIBE}}^{\text{SR-sID-CPA}}(n) = 1 \right] - \frac{1}{2} \right|.$$

Figure 3.1: Our lattice-based SR-IBE

3.3 A Lattice-based SR-IBE Scheme

At a high level, our SR-IBE scheme is a combination of the ABB HIBE (Scheme 2.3.13) and Chen et al.’s RIBE (Scheme 2.4.4). The key authority, who holds master secret keys for both schemes, gives HIBE private keys to recipients, and issues tokens consisting of RIBE private keys to the server. At each time period, the KGC sends an RIBE update key to the server. An illustrated picture is provided in Figure 3.1. The main difficulty is how to design ciphertexts so that they can be transferred to HIBE ciphertexts by the server. To this end, we adopt the double encryption technique. Specifically, the encryption algorithm is a two-step procedure:

1. Encrypt the message M under the HIBE, with respect to an “identity” (id, t) , to obtain an initial ciphertext of the form $(c_2, c_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q$.
2. Encrypt the binary representation $\text{bin}(c_0) \in \{0, 1\}^k$ of c_0 , where $k = \lceil \log q \rceil$, under the RIBE, with respect to id and t , to obtain $(c_1, \hat{c}_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$. The final ciphertext is set as $ct_{id,t} = (c_1, c_2, \hat{c}_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$.

If identity id is not revoked at time t , then the server can partially decrypt ciphertext $ct_{id,t}$, using a transformation key which is essentially the RIBE decryption key. Observe that the “partially decrypted ciphertext” is nothing but the initial ciphertext (c_2, c_0) from the HIBE system. Receiving (c_2, c_0) from the server, the recipient can decrypt it using a decryption key delegated from his long-term private key.

3.3.1 Description of the Scheme

Here, we will formally describe our lattice-based SR-IBE scheme.

Scheme 3.3.1. The algorithms in the scheme are defined as follows.

Sys(1^n): On input security parameter n , the KGC performs the following steps:

1. Set $N = \text{poly}(n)$ as the maximal number of users that the system will support.
2. Let $q = \tilde{O}(n^4)$ be a prime, and set $k = \lceil \log q \rceil, m = 2nk$. Note that parameters n, q, k, m specify vector $\mathbf{g}$, function $\text{bin}(\cdot)$ and primitive matrix $\mathbf{G}$ (see Lemma 2.3.2).
3. Pick a Gaussian parameter $s = \tilde{O}(\sqrt{m})$.
4. Set $B = \tilde{O}(\sqrt{n})$ and let χ be a B -bounded distribution.
5. Choose an FRD map $\mathbf{H} : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^{n \times n}$ (see Scheme 2.3.13).
6. Let the identity space be $\mathcal{I} = \mathbb{Z}_q^n$, the time space be $\mathcal{T} \subset \mathbb{Z}_q^n$ and the message space be $\mathcal{M} = \{0, 1\}$.
7. Output $\text{params} = (n, N, q, k, m, \mathbf{g}, \mathbf{G}, \text{bin}, s, B, \chi, \mathbf{H}, \mathcal{I}, \mathcal{T}, \mathcal{M})$.

Setup(params): On input the system parameters params , the KGC performs the following steps:

1. Generate two independent pairs $(\mathbf{A}, \mathbf{T}_\mathbf{A})$ and $(\mathbf{B}, \mathbf{T}_\mathbf{B})$ using $\text{TrapGen}(n, q, m)$.
2. Choose $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}, \mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{A}_1, \mathbf{A}_2, \mathbf{B}_1, \mathbf{B}_2 \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$.
3. Initialize the revocation list $\text{RL} := \emptyset$. Obtain a binary tree BT with at least N leaf nodes and set the state $\text{st} := \text{BT}$.
4. Set $\text{pp} = (\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{U}, \mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{v})$ and $\text{msk} = (\mathbf{T}_\mathbf{A}, \mathbf{T}_\mathbf{B})$.
5. Output $(\text{pp}, \text{msk}, \text{RL}, \text{st})$.

Token(msk, id, st): On input the master secret key msk , an identity $\text{id} \in \mathcal{I}$ and state st , the KGC performs the following steps:

1. Randomly select an unassigned leaf node ν_{id} in BT and assign it to id .
2. For each $\theta \in \text{Path}(\nu_{\text{id}})$, if $\mathbf{U}_{1,\theta}, \mathbf{U}_{2,\theta}$ are undefined, then choose $\mathbf{U}_{1,\theta} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$, set $\mathbf{U}_{2,\theta} = \mathbf{U} - \mathbf{U}_{1,\theta}$ and store the pair $(\mathbf{U}_{1,\theta}, \mathbf{U}_{2,\theta})$ in node θ . Sample

$$\mathbf{Z}_{1,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_1 + \text{H}(\text{id})\mathbf{G}, \mathbf{T}_{\mathbf{A}}, \mathbf{U}_{1,\theta}, s).$$

Let $\mathbf{A}_{\text{id}} = [\mathbf{A} \mid \mathbf{A}_1 + \text{H}(\text{id})\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$. Remark that $\mathbf{Z}_{1,\theta} \in \mathbb{Z}^{2m \times k}$ and $\mathbf{A}_{\text{id}} \cdot \mathbf{Z}_{1,\theta} = \mathbf{U}_{1,\theta}$.

3. Output the updated state st and $\tau_{\text{id}} = (\theta, \mathbf{Z}_{1,\theta})_{\theta \in \text{Path}(\nu_{\text{id}})}$.

UpdKG(msk, t, st, RL): On input the master secret key msk , a time $t \in \mathcal{T}$, state st and the revocation list RL , the KGC performs the following steps:

1. For each $\theta \in \text{KUNodes}(\text{BT}, \text{RL}, t)$, retrieve $\mathbf{U}_{2,\theta}$ (observe that $\mathbf{U}_{2,\theta}$ is always pre-defined in algorithm **Token**), and sample

$$\mathbf{Z}_{2,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_2 + \text{H}(t)\mathbf{G}, \mathbf{T}_{\mathbf{A}}, \mathbf{U}_{2,\theta}, s).$$

Let $\mathbf{A}_t = [\mathbf{A} \mid \mathbf{A}_2 + \text{H}(t)\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$. Remark that $\mathbf{Z}_{2,\theta} \in \mathbb{Z}^{2m \times k}$ and $\mathbf{A}_t \cdot \mathbf{Z}_{2,\theta} = \mathbf{U}_{2,\theta}$.

2. Output $\text{uk}_t = (\theta, \mathbf{Z}_{2,\theta})_{\theta \in \text{KUNodes}(\text{BT}, \text{RL}, t)}$.

TranKG($\tau_{\text{id}}, \text{uk}_t$): On input token $\tau_{\text{id}} = (\theta, \mathbf{Z}_{1,\theta})_{\theta \in I}$ and update key $\text{uk}_t = (\theta, \mathbf{Z}_{2,\theta})_{\theta \in J}$ for some set of nodes I, J , the server performs the following steps:

1. If $I \cap J = \emptyset$, output $\perp$.
2. Otherwise, pick $\theta \in I \cap J$ and output $\text{tk}_{\text{id},t} = (\mathbf{Z}_{1,\theta}, \mathbf{Z}_{2,\theta})$. Remark that $\mathbf{A}_{\text{id}} \cdot \mathbf{Z}_{1,\theta} + \mathbf{A}_t \cdot \mathbf{Z}_{2,\theta} = \mathbf{U}$.

PrivKG(msk, id): On input the master secret key msk and an identity $\text{id} \in \mathcal{I}$, the KGC performs the following steps:

1. Sample $\mathbf{T}_{\text{id}} \leftarrow \text{SampleBasisLeft}(\mathbf{B}, \mathbf{B}_1 + \mathbf{H}(\text{id})\mathbf{G}, \mathbf{T}_{\mathbf{B}}, s)$. Note that $\mathbf{T}_{\text{id}} \in \mathbb{Z}^{2m \times 2m}$ is a basis of $\Lambda_q^\perp(\mathbf{B}_{\text{id}})$, where $\mathbf{B}_{\text{id}} = [\mathbf{B} \mid \mathbf{B}_1 + \mathbf{H}(\text{id})\mathbf{G}] \in \mathbb{Z}_q^{n \times 2m}$.
2. Output $\text{sk}_{\text{id}} = \mathbf{T}_{\text{id}}$.

DeckG($\text{sk}_{\text{id}}, \mathbf{t}$): On input a private key $\text{sk}_{\text{id}} = \mathbf{T}_{\text{id}}$ and a time $\mathbf{t} \in \mathcal{T}$, the recipient performs the following steps:

1. Sample $\mathbf{r}_{\text{id}, \mathbf{t}} \leftarrow \text{SampleLeft}(\mathbf{B}_{\text{id}}, \mathbf{B}_2 + \mathbf{H}(\mathbf{t})\mathbf{G}, \mathbf{T}_{\text{id}}, \mathbf{v}, s)$. Note that $\mathbf{r}_{\text{id}, \mathbf{t}} \in \mathbb{Z}^{3m}$ and $\mathbf{B}_{\text{id}, \mathbf{t}} \cdot \mathbf{r}_{\text{id}, \mathbf{t}} = \mathbf{v}$, where $\mathbf{B}_{\text{id}, \mathbf{t}} = [\mathbf{B} \mid \mathbf{B}_1 + \mathbf{H}(\text{id})\mathbf{G} \mid \mathbf{B}_2 + \mathbf{H}(\mathbf{t})\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$.
2. Output $\text{dk}_{\text{id}, \mathbf{t}} = \mathbf{r}_{\text{id}, \mathbf{t}}$.

Enc($\text{id}, \mathbf{t}, b$): On input an identity $\text{id} \in \mathcal{I}$, a time $\mathbf{t} \in \mathcal{T}$ and a message $M \in \mathcal{M}$, the sender performs the following steps:

1. Set $\mathbf{A}_{\text{id}, \mathbf{t}} = [\mathbf{A} \mid \mathbf{A}_1 + \mathbf{H}(\text{id})\mathbf{G} \mid \mathbf{A}_2 + \mathbf{H}(\mathbf{t})\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$.
2. Sample $\mathbf{s}, \mathbf{s}' \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}, \mathbf{e}' \leftarrow \chi^m$, $\mathbf{e}'' \leftarrow \chi^k$, and $e \leftarrow \chi$.
3. Pick $\mathbf{R}_1, \mathbf{R}_2, \mathbf{S}_1, \mathbf{S}_2 \xleftarrow{\$} \{-1, 1\}^{m \times m}$.
4. Compute $c_0 = \mathbf{v}^\top \mathbf{s}' + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q$.
5. Output $\text{ct}_{\text{id}, \mathbf{t}} = (\mathbf{c}_1, \mathbf{c}_2, \hat{\mathbf{c}}_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$, where:

$$\begin{cases} \mathbf{c}_1 = \mathbf{A}_{\text{id}, \mathbf{t}}^\top \mathbf{s} + (\mathbf{e} \|\mathbf{R}_1^\top \mathbf{e} \|\mathbf{R}_2^\top \mathbf{e}) \in \mathbb{Z}_q^{3m}, \\ \mathbf{c}_2 = \mathbf{B}_{\text{id}, \mathbf{t}}^\top \mathbf{s}' + (\mathbf{e}' \|\mathbf{S}_1^\top \mathbf{e}' \|\mathbf{S}_2^\top \mathbf{e}') \in \mathbb{Z}_q^{3m}, \\ \hat{\mathbf{c}}_0 = \mathbf{U}^\top \mathbf{s} + \mathbf{e}'' + \text{bin}(c_0) \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q^k. \end{cases}$$

Transform($\text{ct}_{\text{id}, \mathbf{t}}, \text{tk}_{\text{id}, \mathbf{t}}$): On input a ciphertext $\text{ct}_{\text{id}, \mathbf{t}} = (\mathbf{c}_1, \mathbf{c}_2, \hat{\mathbf{c}}_0)$ and a transformation key $\text{tk}_{\text{id}, \mathbf{t}} = (\mathbf{Z}_1, \mathbf{Z}_2)$, the server performs the following steps:

1. Parse $\mathbf{c}_1 = (\mathbf{c}_{1,0} \|\mathbf{c}_{1,1} \|\mathbf{c}_{1,2})$ where $\mathbf{c}_{1,i} \in \mathbb{Z}_q^m$, for $i = 0, 1, 2$. Compute $\mathbf{w} = \hat{\mathbf{c}}_0 - \mathbf{Z}_1^\top (\mathbf{c}_{1,0} \|\mathbf{c}_{1,1}) - \mathbf{Z}_2^\top (\mathbf{c}_{1,0} \|\mathbf{c}_{1,2}) \in \mathbb{Z}_q^k$.
2. Compute $\hat{c}'_0 = \langle \mathbf{g}, \left\lfloor \frac{2}{q} \cdot \mathbf{w} \right\rangle \rangle \in \mathbb{Z}_q$. (Recall that $\mathbf{g} = (1, 2, \dots, 2^{k-1}) \in \mathbb{Z}^k$.)

3. Output $\mathbf{ct}'_{\text{id},\mathbf{t}} = (\mathbf{c}_2, \hat{c}'_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q$.

Dec($\mathbf{ct}'_{\text{id},\mathbf{t}}, \mathbf{dk}_{\text{id},\mathbf{t}}$): On input a partially decrypted ciphertext $\mathbf{ct}'_{\text{id},\mathbf{t}} = (\mathbf{c}_2, \hat{c}'_0)$ and a decryption key $\mathbf{dk}_{\text{id},\mathbf{t}} = \mathbf{r}_{\text{id},\mathbf{t}}$, the recipient performs the following steps:

1. Compute $w' = \hat{c}'_0 - \mathbf{r}_{\text{id},\mathbf{t}}^\top \mathbf{c}_2 \in \mathbb{Z}_q$.
2. Output $\lfloor \frac{2}{q} \cdot w' \rfloor \in \{0, 1\}$.

Revoke($\text{id}, \mathbf{t}, \mathbf{RL}, \text{st}$): On input an identity id , a time $\mathbf{t}$, the revocation list $\mathbf{RL}$ and state $\text{st} = \text{BT}$, the KGC updates $\mathbf{RL}$ by adding $(\nu_{\text{id}}, \mathbf{t})$, where ν_{id} is the node associated with identity id .

3.3.2 Correctness and Efficiency

In this section, we analyze the correctness and efficiency of our SR-IBE scheme.

Correctness. When the scheme is operated as specified, if recipient id is non-revoked at time $\mathbf{t}$, then $\mathbf{tk}_{\text{id},\mathbf{t}} = (\mathbf{Z}_1, \mathbf{Z}_2)$ satisfies:

$$\mathbf{A}_{\text{id}} \cdot \mathbf{Z}_1 + \mathbf{A}_{\mathbf{t}} \cdot \mathbf{Z}_2 = \mathbf{U}.$$

In algorithm **Transform** performed by the server, we have:

$$\begin{aligned} \mathbf{w} &= \hat{\mathbf{c}}_0 - \mathbf{Z}_1^\top (\mathbf{c}_{1,0} \parallel \mathbf{c}_{1,1}) - \mathbf{Z}_2^\top (\mathbf{c}_{1,0} \parallel \mathbf{c}_{1,2}) \\ &= \mathbf{U}^\top \mathbf{s} + \mathbf{e}'' + \text{bin}(c_0) \cdot \left\lfloor \frac{q}{2} \right\rfloor - \mathbf{Z}_1^\top (\mathbf{A}_{\text{id}}^\top \mathbf{s} + (\mathbf{e} \parallel \mathbf{R}_1^\top \mathbf{e})) - \mathbf{Z}_2^\top (\mathbf{A}_{\mathbf{t}}^\top \mathbf{s} + (\mathbf{e} \parallel \mathbf{R}_2^\top \mathbf{e})) \\ &= \text{bin}(c_0) \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{\mathbf{e}'' - \mathbf{Z}_1^\top (\mathbf{e} \parallel \mathbf{R}_1^\top \mathbf{e}) - \mathbf{Z}_2^\top (\mathbf{e} \parallel \mathbf{R}_2^\top \mathbf{e})}_{\text{error}}. \end{aligned}$$

Observe that if the error term above is less than $q/5$, i.e., $\|\text{error}\|_\infty < q/5$, then in Step 2 of algorithm **Transform**, we have that $\lfloor \frac{2}{q} \cdot \mathbf{w} \rfloor = \text{bin}(c_0)$ which implies $\hat{c}'_0 = \langle \mathbf{g}, \lfloor \frac{2}{q} \cdot \mathbf{w} \rfloor \rangle = c_0$.

Then, in the **Dec** algorithm run by the recipient, we have:

$$\begin{aligned}
w' &= \hat{c}'_0 - \mathbf{r}_{\text{id},t}^\top \mathbf{c}_2 \\
&= \mathbf{v}^\top \mathbf{s}' + e + M \cdot \left\lfloor \frac{q}{2} \right\rfloor - \mathbf{r}_{\text{id},t}^\top \left(\mathbf{B}_{\text{id},t}^\top \mathbf{s}' + \left(\mathbf{e}'^\top \mathbf{S}_1^\top \mathbf{e}' \right) \mathbf{S}_2^\top \mathbf{e}' \right) \\
&= M \cdot \left\lfloor \frac{q}{2} \right\rfloor + \underbrace{e - \mathbf{r}_{\text{id},t}^\top \left(\mathbf{e}'^\top \mathbf{S}_1^\top \mathbf{e}' \right) \mathbf{S}_2^\top \mathbf{e}'}_{\text{error}'}.
\end{aligned}$$

Similarly, if the above error term is bounded by $q/5$, i.e., $|\text{error}'| < q/5$, then the recipient should be able to recover the message.

As in [2, 28], the two error terms above are both bounded by $sm^2B \cdot \omega(\log n) = \tilde{O}(n^3)$, which is much smaller than $q/5$, since we set $q = \tilde{O}(n^4)$. This implies the correctness of our scheme.

Efficiency. The efficiency aspect of Scheme 3.3.1 is as follows:

- ◇ The the public parameters $\mathbf{pp}$ has bit-size $(6nm + nk + n) \log q = \tilde{O}(n^2)$.
- ◇ The private key $\mathbf{sk}_{\text{id}}$ is a trapdoor matrix of bit-size $\tilde{O}(n^2)$.
- ◇ The token τ_{id} has bit-size $O(\log N) \cdot \tilde{O}(n)$.
- ◇ The bit-size of the update key $\mathbf{uk}_t$ is $O\left(r \log \frac{N}{r}\right) \cdot \tilde{O}(n)$.
- ◇ The bit-size of the ciphertext $\mathbf{ct}_{\text{id},t}$ is $(6m + k) \log q = \tilde{O}(n)$.
- ◇ The partially decrypted ciphertext $\mathbf{ct}'_{\text{id},t}$ has bit-size $(3m + 1) \log q = \tilde{O}(n)$.

3.3.3 Security

In the following theorem, we show that our SR-IBE scheme is selectively secure in the standard model, assuming hardness of the LWE problem.

Theorem 3.3.2. For the given parameters, the SR-IBE described in Scheme 3.3.1 satisfies the security notion of SR-sID-CPA defined in Definition 3.2.2, provided that the (n, q, χ) -LWE assumption holds.

Proof Outline. We first highlight the main idea of the security proof. The detailed proof is provided later.

Recall that our SR-IBE scheme can be reviewed as a combination of a two-level variant of the ABB HIBE (in Scheme 2.3.13) and Chen et al.'s RIBE (in Scheme 2.4.4). Both of these building blocks are proved to be selectively secure under the **LWE** assumption. We consider two distinct types of adversaries as follows.

Type I Adversary: The adversary is allowed to issue a query to the private key oracle $\text{PrivKG}(\cdot)$ on the challenge identity id^* . In this case, id^* must be revoked before the challenge time t^* .

Type II Adversary: The adversary never issues a query to the private key oracle $\text{PrivKG}(\cdot)$ on the challenge identity id^* . Nevertheless, it may query the decryption key oracle $\text{DecKG}(\cdot, \cdot)$ on (id^*, t) as long as $\text{t} \neq \text{t}^*$.

We can view our proof as a reduction to either the **IND-sRID-CPA** security of Chen et al.'s RIBE scheme (see Theorem 2.4.5) or the **IND-sID-CPA** security of the ABB HIBE scheme (see Theorem 2.3.14), based on which type of adversary the SR-IBE scheme is faced with.

For Type I adversary, it can retrieve the depth-1 private key of the HIBE scheme, and therefore any decryption key which is essentially a depth-2 private key. In this way, we cannot reduce our security to the security of the underlying HIBE, and alternatively, we reduce it to that of the underlying RIBE. Using the RIBE, we can answer all queries to tokens (which are actually RIBE private keys) and update keys (which are RIBE update keys as well).

For Type II adversary, it does not query the private key of id^* , which implies that queries to tokens and update keys for all identities and time periods, even for id^* and t^* , are allowed. In this setting, we consider to reduce our security to the security of the underlying HIBE and it is possible to answer all queries to private keys and decryption keys using the underlying HIBE scheme.

Proof. We will show that if there is a PPT adversary $\mathcal{A}$ succeeding in breaking the SR-sID-CPA security of our SR-IBE scheme, then we can use it to construct a PPT algorithm $\mathcal{S}$ breaking either the IND-sRID-CPA security of Chen et al.'s RIBE scheme or the IND-sID-CPA security of the ABB HIBE scheme. Then the theorem follows from the facts that the two building blocks are both secure under the (n, q, χ) -LWE assumption (see Theorem 2.3.14 and Theorem 2.4.5).

Assume that id^* is the target identity and $\mathbf{t}^*$ is the target time. We consider two distinct types of adversaries as follows.

Type I Adversary: The adversary is allowed to issue a query to the private key oracle $\text{PrivKG}(\cdot)$ on the challenge identity id^* . In this case, id^* must be revoked before the challenge time $\mathbf{t}^*$.

Type II Adversary: The adversary never issues a query to the private key oracle $\text{PrivKG}(\cdot)$ on the challenge identity id^* . Nevertheless, it may query the decryption key oracle $\text{DecKG}(\cdot, \cdot)$ on $(\text{id}^*, \mathbf{t})$ as long as $\mathbf{t} \neq \mathbf{t}^*$.

The simulator $\mathcal{S}$ begins by randomly guessing which type of adversaries is going to be. In the following, we separately describe the $\mathcal{S}$'s progress for the two types of adversaries.

Lemma 3.3.3. If there is a PPT Type I adversary $\mathcal{A}$ breaking the SR-sID-CPA security of the SR-IBE in Scheme 3.3.1 with advantage ϵ , then there is a PPT algorithm $\mathcal{S}$ breaking the IND-sRID-CPA security of Chen et al.'s RIBE in Scheme 2.4.4 with the same advantage.

Proof. Assume that $\mathcal{B}$ is the challenger in the IND-sRID-CPA game for Chen et al.'s RIBE scheme. Algorithm $\mathcal{S}$ will interact with $\mathcal{A}$ and $\mathcal{B}$ as follows.

Initial: $\mathcal{S}$ runs algorithm $\text{Sys}(1^n)$ to output a set of system parameters $\text{params} = (n, N, q, k, m, \mathbf{g}, \mathbf{G}, \text{bin}, s, B, \chi, \mathbf{H}, \mathcal{I}, \mathcal{T}, \mathcal{M})$, as in the real scheme. The adversary

$\mathcal{A}$ announces to $\mathcal{S}$ the challenge id^* and $\mathbf{t}^*$, and $\mathcal{S}$ forwards them to the RIBE challenger $\mathcal{B}$.

Setup: $\mathcal{S}$ initializes the revocation list RL as a empty set and set the state st as a binary tree BT . Then $\mathcal{S}$ prepares the public parameters pp as follows:

1. Receive $\text{pp}_{\text{RIBE}} = (\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{U})$ from $\mathcal{B}$, where $\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2 \in \mathbb{Z}_q^{n \times m}$, $\mathbf{U} \in \mathbb{Z}_q^{n \times k}$.
2. Generate $(\mathbf{B}, \mathbf{T}_{\mathbf{B}})$ by running $\text{TrapGen}(n, q, m)$. Choose $\mathbf{B}_1, \mathbf{B}_2 \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and $\mathbf{v} \xleftarrow{\$} \mathbb{Z}_q^n$.
3. Let $\text{pp} = (\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{U}, \mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{v})$, and send pp to $\mathcal{A}$. Remark that the distribution of pp is exactly the one expected by $\mathcal{A}$.

Token and Update Key Oracles: When $\mathcal{A}$ queries a token for identity id , algorithm $\mathcal{S}$ forwards id to $\mathcal{B}$. The simulator $\mathcal{S}$ receives an RIBE private key from $\mathcal{B}$, and forwards it to $\mathcal{A}$ as the token τ_{id} . Similarly, if $\mathcal{A}$ queries an update key for time $\mathbf{t}$, then $\mathcal{S}$ sets the update key $\text{uk}_{\mathbf{t}}$ as the RIBE update key it obtains by interacting with $\mathcal{B}$. Recall that for a Type I adversary, the target identity id^* must be revoked before the target time $\mathbf{t}^*$, which means that $\mathcal{A}$ is allowed to query token for id^* and also update key for $\mathbf{t}^*$.

Private Key and Decryption Key Oracles: Since $\mathcal{S}$ knows the master secret key part $\mathbf{T}_{\mathbf{B}}$, it can answer all private key and decryption key queries exactly as in Scheme 3.3.1.

Challenge: $\mathcal{A}$ outputs two messages $M_0, M_1 \in \mathcal{M}$ to $\mathcal{S}$ who prepares the challenge ciphertext as follows:

1. Select $\mathbf{s}' \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}' \xleftarrow{\$} \chi^m$ and $e \xleftarrow{\$} \chi$. Pick $\mathbf{S}_1, \mathbf{S}_2 \xleftarrow{\$} \{-1, 1\}^{m \times m}$.
2. Set $\mathbf{B}_{\text{id}^*, \mathbf{t}^*} = [\mathbf{B} \mid \mathbf{B}_1 + \text{H}(\text{id}^*)\mathbf{G} \mid \mathbf{B}_2 + \text{H}(\mathbf{t}^*)\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$.
3. Compute $M'_0 = \mathbf{v}^\top \mathbf{s}' + e + M_0 \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q$ and $M'_1 = \mathbf{v}^\top \mathbf{s}' + e + M_1 \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q$.

4. Pick $d \xleftarrow{\$} \{0, 1\}$ and set

$$M_0'' = \text{bin}(M_d') \in \{0, 1\}^k; \quad M_1'' = \text{bin}(M_{1 \oplus d}') \in \{0, 1\}^k,$$

where $\oplus$ denotes the addition modulo 2.

Then forward M_0'', M_1'' as two challenge messages to the RIBE challenger $\mathcal{B}$, who will return an RIBE ciphertext $(\mathbf{c}'_1, \mathbf{c}'_0) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$ of M_c'' under identity id^* and time $\mathbf{t}^*$. Here $c \in \{0, 1\}$ is uniformly random.

5. $\mathcal{A}$ is then given the challenge ciphertext $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$ where:

$$\begin{cases} \mathbf{c}_1^* = \mathbf{c}'_1 \in \mathbb{Z}_q^{3m}, \\ \mathbf{c}_2^* = \mathbf{B}_{\text{id}^*, \mathbf{t}^*}^\top \mathbf{s}' + (\mathbf{e}' \| \mathbf{S}_1^\top \mathbf{e}' \| \mathbf{S}_2^\top \mathbf{e}') \in \mathbb{Z}_q^{3m}, \\ \hat{\mathbf{c}}_0^* = \mathbf{c}'_0 \in \mathbb{Z}_q^k. \end{cases}$$

Note that, by construction, we have $M_c'' = \text{bin}(M_{d \oplus c}')$, and thus, $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*)$ is an SR-IBE encryption of the message $M_{d \oplus c}'$ under $(\text{id}^*, \mathbf{t}^*)$. Note also that, the bit $d \oplus c$ is uniformly random in $\{0, 1\}$.

Guess: After being allowed to make additional queries, $\mathcal{A}$ outputs $d' \in \{0, 1\}$ as the guess that the challenge ciphertext $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*)$ is an encryption of $M_{d'}'$. Then $\mathcal{S}$ computes $c' = d \oplus d'$ and returns it to $\mathcal{B}$, which is the guess for the bit c chosen by the latter.

On the one hand, recall that it is assumed that $\mathcal{A}$ breaks the SR-sID-CPA security of our SR-IBE scheme with advantage ϵ , which means

$$\text{Adv}_{\mathcal{A}, \text{SRIBE}}^{\text{SR-sID-CPA}}(n) = \left| \Pr[d' = d \oplus c] - \frac{1}{2} \right| = \epsilon.$$

On the other hand, by construction, we have $d' = d \oplus c \Leftrightarrow d' \oplus d = c \Leftrightarrow c' = c$. It then

follows that:

$$\text{Adv}_{S, \mathcal{RIBE}}^{\text{IND-sRID-CPA}}(n) = \left| \Pr[c = c'] - \frac{1}{2} \right| = \epsilon.$$

□

Lemma 3.3.4. If there is a PPT Type II adversary $\mathcal{A}$ breaking the SR-sID-CPA security of our SR-IBE in Scheme 3.3.1 with advantage ϵ , then there is a PPT adversary $\mathcal{S}$ breaking the IND-sID-CPA security of the ABB HIBE in Scheme 2.3.13 with the same advantage.

Proof. We proceed in a similar way as in the Lemma 3.3.3. Let $\mathcal{B}$ be the challenger in the IND-sID-CPA game for the ABB HIBE scheme. Algorithm $\mathcal{S}$ will interact with $\mathcal{A}$ and $\mathcal{B}$ as follows.

Initial: $\mathcal{S}$ runs algorithm $\text{Sys}(1^n)$ to output a set of system parameters $\text{params} = (n, N, q, k, m, \mathbf{g}, \mathbf{G}, \text{bin}, s, B, \chi, \mathbf{H}, \mathcal{I}, \mathcal{T}, \mathcal{M})$. Then $\mathcal{A}$ announces to $\mathcal{S}$ the target identity id^* and time $\mathbf{t}^*$, and $\mathcal{S}$ forwards $(\text{id}^*, \mathbf{t}^*)$ to the HIBE challenger $\mathcal{B}$, as a depth-2 target “identity”.

Setup: $\mathcal{S}$ initializes the revocation list RL as a empty set and set the sate st as a binary tree BT . Then $\mathcal{S}$ prepares the public parameters pp as follows:

1. Receive $\text{pp}_{\text{HIBE}} = (\mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{v})$ from $\mathcal{B}$ where $\mathbf{B}, \mathbf{B}_1, \mathbf{B}_2 \in \mathbb{Z}_q^{n \times m}$, $\mathbf{v} \in \mathbb{Z}_q^n$.
2. Generate $(\mathbf{A}, \mathbf{T}_{\mathbf{A}})$ by running $\text{TrapGen}(n, q, m)$. Choose $\mathbf{A}_1, \mathbf{A}_2 \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times k}$.
3. Let the public parameters be $\text{pp} = (\mathbf{A}, \mathbf{A}_1, \mathbf{A}_2, \mathbf{U}, \mathbf{B}, \mathbf{B}_1, \mathbf{B}_2, \mathbf{v})$ and send it to $\mathcal{A}$.

Token and Update Key Oracles: Since $\mathcal{S}$ knows the master secret key part $\mathbf{T}_{\mathbf{A}}$, it can answer all token and update key queries as in the real scheme.

Private Key and Decryption Key Oracles: If $\mathcal{A}$ issues a private key query for an identity id where $\text{id} \neq \text{id}^*$, then $\mathcal{S}$ forwards id to $\mathcal{B}$. Receiving the private key from the latter, $\mathcal{S}$ sets it as the private key sk_{id} and forwards sk_{id} to $\mathcal{A}$. When $\mathcal{A}$ queries a decryption key for an identity id and a time t , where $(\text{id}, \text{t}) \neq (\text{id}^*, \text{t}^*)$, algorithm $\mathcal{S}$ forwards $(\text{id}, \text{t}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q^n$ to $\mathcal{B}$, and gets a matrix $\mathbf{Z} \in \mathbb{Z}^{3m \times 3m}$, which is a short trapdoor for $\Lambda_q^\perp(\mathbf{B}_{\text{id}, \text{t}})$ where $\mathbf{B}_{\text{id}, \text{t}} = [\mathbf{B} \mid \mathbf{B}_1 + \text{H}(\text{id})\mathbf{G} \mid \mathbf{B}_2 + \text{H}(\text{t})\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$. Then $\mathcal{S}$ samples $\mathbf{r}_{\text{id}, \text{t}} \leftarrow \text{SamplePre}(\mathbf{B}_{\text{id}, \text{t}}, \mathbf{Z}, \mathbf{v}, s)$, sets $\mathbf{dk}_{\text{id}, \text{t}} = \mathbf{r}_{\text{id}, \text{t}}$, and sends it to $\mathcal{A}$. The requirement $(\text{id}, \text{t}) \neq (\text{id}^*, \text{t}^*)$ means that, if $\text{t} \neq \text{t}^*$ then id can be the same as id^* .

Challenge: $\mathcal{A}$ gives two challenge messages $M_0, M_1 \in \{0, 1\}$ to $\mathcal{S}$, who prepares the challenge ciphertext as follows:

1. Pick $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$ and $\mathbf{e}' \leftarrow \chi^k$. Select $\mathbf{R}_1, \mathbf{R}_2 \xleftarrow{\$} \{-1, 1\}^{m \times m}$.
2. Set $\mathbf{A}_{\text{id}^*, \text{t}^*} = [\mathbf{A} \mid \mathbf{A}_1 + \text{H}(\text{id}^*)\mathbf{G} \mid \mathbf{A}_2 + \text{H}(\text{t}^*)\mathbf{G}] \in \mathbb{Z}_q^{n \times 3m}$.
3. Pick $d \xleftarrow{\$} \{0, 1\}$. Set $M'_0 = M_d$ and $M'_1 = M_{1 \oplus d}$. Forward M'_0, M'_1 as two challenge messages to the HIBE challenger $\mathcal{B}$, who will return a ciphertext $(\mathbf{c}_1'', c_0'') \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q$, as an HIBE encryption of message M'_c under $(\text{id}^*, \text{t}^*)$. Here $c \xleftarrow{\$} \{0, 1\}$.
4. $\mathcal{A}$ is given $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*) \in \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^{3m} \times \mathbb{Z}_q^k$ where:

$$\begin{cases} \mathbf{c}_1^* = \mathbf{A}_{\text{id}^*, \text{t}^*}^\top \mathbf{s} + (\mathbf{e} \|\mathbf{R}_1^\top \mathbf{e} \|\mathbf{R}_2^\top \mathbf{e}) \in \mathbb{Z}_q^{3m}, \\ \mathbf{c}_2^* = \mathbf{c}_1'' \in \mathbb{Z}_q^{3m}, \\ \hat{\mathbf{c}}_0^* = \mathbf{U}^\top \mathbf{s} + \vec{y} + \text{bin}(c_0'') \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q^k. \end{cases}$$

Note that $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*)$ is an SR-IBE encryption of the message $M_{c \oplus d} = M'_c$ under $(\text{id}^*, \text{t}^*)$.

Guess: After being allowed to make additional queries, $\mathcal{A}$ outputs $d' \in \{0, 1\}$ as the

guess that the challenge ciphertext $(\mathbf{c}_1^*, \mathbf{c}_2^*, \hat{\mathbf{c}}_0^*)$ is an encryption of $M_{d'}$. Then $\mathcal{S}$ computes $c' = d \oplus d'$ and returns it to $\mathcal{B}$, which is the guess for the bit c chosen by the latter.

On the one hand, recall that it is assumed that $\mathcal{A}$ breaks the SR-sID-CPA security of our SR-IBE scheme with probability ϵ , which means

$$\text{Adv}_{\mathcal{A}, \text{SRIBE}}^{\text{SR-sID-CPA}}(n) = \left| \Pr[d' = d \oplus c] - \frac{1}{2} \right| = \epsilon.$$

On the other hand, by construction, we have $d' = d \oplus c \Leftrightarrow d' \oplus d = c \Leftrightarrow c' = c$. It then follows that:

$$\text{Adv}_{\mathcal{S}, \text{HIBE}}^{\text{IND-sID-CPA}}(n) = \left| \Pr[c = c'] - \frac{1}{2} \right| = \epsilon.$$

□

Finally, note that algorithm $\mathcal{S}$ can guess the type of the adversary correctly with probability $1/2$ and the adversary's behaviour is independent from the simulator's guess. It then follows from the results of Lemma 3.3.3 and Lemma 3.3.4 that

$$\text{Adv}_{\mathcal{A}, \text{SRIBE}}^{\text{SR-sID-CPA}}(n) = \frac{1}{2} \left(\text{Adv}_{\mathcal{S}, \text{RIBE}}^{\text{IND-sRID-CPA}}(n) + \text{Adv}_{\mathcal{S}, \text{HIBE}}^{\text{IND-sID-CPA}}(n) \right).$$

By Theorem 2.3.14 and Theorem 2.4.5, we then have that $\text{Adv}_{\mathcal{A}, \text{SRIBE}}^{\text{SR-sID-CPA}}(n) = \text{negl}(n)$, provided that the (n, q, χ) -LWE assumption holds. This concludes the proof. □

Further Discussions

Our SR-IBE scheme can only be proven secure in the selective manner, while the pairing-based scheme by Qin et al. [89] is adaptively secure. The selective security inherits from that of the underlying RIBE and HIBE schemes. One possible way to achieve the adaptive security is to employ underlying schemes which are also secure in

the adaptive setting.

Besides of IBE schemes, server-aided revocation mechanism can also be used to realize user revocation for multi-receiver encryption schemes. In Chapter 4, we will consider server-aided revocable predicate encryption.

Chapter 4

Server-aided Revocable Predicate Encryption

4.1 Introduction

In Boldyreva et al.'s [15] key-update revocation model, the non-revoked users have to communicate with the KGC regularly to receive the update keys. Although such key updating process can be done through a public channel, it is somewhat inconvenient and bandwidth-consuming. To reduce the users's computational burden, Qin et al. [89] proposed an interesting solution in the context of identity-based encryption (IBE), called server-aided revocable identity-based encryption (SR-IBE). Qin et al.'s model takes advantage of a publicly accessible server with powerful computational capabilities, to which one can outsource most of users' workloads. Moreover, the server can be untrusted in the sense that it does not possess any secret information.

Cui et al. [31] subsequently adapted the server-aided revocation mechanism into the ABE setting and introduced server-aided revocable attribute-based encryption (SR-ABE). Briefly speaking, an SR-ABE scheme works as follows. When a new user joins the system, he generates a public-secret key-pair, and sends the public key to

the KGC¹. The latter then generates a user-specific token that is forwarded to the untrusted server through a public channel. At each time period, the update key is sent only to the server rather than to all users. To perform decryption for a specific user, the server first transforms the ciphertext into a “partially decrypted ciphertext”. The latter is bound to the user’s public key, so that only the intended user can recover the plaintext using his private key. In [31], apart from introducing this new model, Cui et al. also described a pairing-based instantiation of SR-ABE.

In this work, inspired by the potentials of PE and the advantages of the server-aided revocation mechanism, we consider the notion of server-aided revocable predicate encryption, and aim to design the first such scheme from lattice assumptions.

OUR RESULTS AND TECHNIQUES. The contribution of this work is two-fold: We first formalize the concept of server-aided predicate encryption (SR-PE), and then put forward an instantiation of SR-PE from lattices. An overview of these two results is given below.

Our model of SR-PE inherits the main advantage of the server-aided revocation mechanism [89]: most of the users’ workloads are delegated to an untrusted server. The model can be seen as a non-trivial adaptation of Cui et al.’s model of SR-ABE [31] into the PE setting, with two notable distinctions. First, while Cui et al. assume a public-secret key-pair for each user, we do not require users to maintain their own public keys. Recall that Shamir’s [98] motivation to initiate the study of IBE is to eliminate the burden of managing public keys. The more general notions of ABE and PE later inherit this advantage over traditional public key encryption. From this point of view, the re-introduction of users’ public keys seems contradict to the spirit of identity-based/attribute-based/predicate cryptosystems. Thus, by not demanding the existence of users’ public keys, we make our model consistent with ordinary (i.e., non-revocable) predicate encryption. Second, our security definition reflects the

¹Alternatively, as pointed out by Cui et al. [31], the key-pair can be generated by the KGC and then sent to the user. This requires a secure channel - which is typically assumed to be available in the setting of centralized cryptosystems.

attribute-hiding property of PE systems, which guarantees that attributes bound to the ciphertexts are not revealed during decryptions, and which is not considered in the context of ABE.

As an effort to instantiate a scheme satisfying our model under post-quantum assumptions, we design a lattice-based construction that is proven secure (in a selective manner) in the standard model, assuming the hardness of the Learning With Errors (LWE) problem [90]. The efficiency of our scheme is comparable to that of the constructions under the server-aided revocation approach [31, 77, 89], in the following sense. The sizes of private keys and ciphertexts, as well as the complexity of decryption on the user side are all independent of the number of users N and the number of revoked users r . In particular, the ciphertext size in our scheme compares favourably to that of Ling et al.’s revocable PE scheme [66], which follows the direct revocation approach. It is also worth mentioning that, if we do not assume the availability of the server (which does not affect security because the server does not possess any secret key) and let the users perform the server’s work themselves, then our scheme would yield the first (non-server-aided) lattice-based PE with constant-size ciphertexts.

At a high level, our scheme employs two main building blocks: the ordinary PE scheme by Agrawal et al. [4] and the CS method due to Naor et al. [75]. We observe that the same two ingredients were adopted in Ling et al.’s scheme [66], but their direct revocation approach is fundamentally different from ours, and thus, we have to find a new way to make these ingredients work smoothly together.

Our first challenge is to enable a relatively sophisticated mechanism, in which an original PE ciphertext bound to an attribute and a time period (but not bound to any user’s identifying information), after being transformed by the server, would become a partially decrypted ciphertext bound to the identifying information of the non-revoked recipient. We note that, in the setting of lattice-based SR-IBE, Nguyen et al. [77] addressed a somewhat related problem using a double encryption technique, where the original and the partially decrypted ciphertexts are both bound to the recipient’s

identity and time period. However such technique requires the sender to know the recipient's identity when generating the ciphertext, and hence, it is not applicable to the PE setting. We further note that, Cui et al. [31] solved a more closely related problem, in which the partially decrypted ciphertext is constrained to bind to the recipient's public key - with respect to some public-key encryption (PKE) system. We observe that it is possible to adapt the technique from [31], but as our SR-PE model does not work with users' public keys, we will instead make use of an IBE instance. Namely, we additionally employ the IBE from [2] and assign each user an identity id . The challenge now is how to embed id into the user-specific token in a way such that the partially decrypted ciphertext will be bound to id .

To address the above problem, we exploit a special property of some **LWE**-based encryption systems, observed by Boneh et al. [19], which allows to transform an encryption of a message under one key into an encryption of the same message under another key. Then, our scheme works roughly as follows. Each user with identity id is issued a private key for a two-level hierarchical system consisting of one instance of the PE system from [4] as well as an additional IBE level for id , associated with a matrix $\mathbf{D}_{\text{id}}$. Meanwhile, the token for id is generated by embedding $\mathbf{D}_{\text{id}}$ into another instance of the same PE system [4]. At each time period $\mathbf{t}$, the KGC employs the CS method to compute an update key $\mathbf{uk}_{\mathbf{t}}$ and sends it to the server. A ciphertext in our scheme is a combination of two PE ciphertexts and an extra component bound to $\mathbf{t}$. If recipient id is not revoked at time period $\mathbf{t}$, the server can use the token for id and $\mathbf{uk}_{\mathbf{t}}$ to transform the second PE ciphertext into an IBE ciphertext associated with $\mathbf{D}_{\text{id}}$, thanks to the special property mentioned above. Finally, the partially decrypted ciphertext, consisting of the first PE ciphertext and the IBE ciphertext, can be fully decrypted using the private key of id .

The security of our proposed SR-PE scheme relies on that of the two lattice-based components from [4] and [2]. Both of them are selectively secure in the standard model, assuming the hardness of the **LWE** problem - so is our scheme.

Figure 4.1: Architecture of server-aided revocation mechanism

ORGANIZATION. The rest of this paper is organized as follows. We give the rigorous definition and security model of SR-PE in Section 4.2. Our main construction is described and analyzed in Section 4.3.

4.2 The model of SR-PE

In this section, we describe the rigorous definition and security model of SR-PE, based on the server-aided revocation mechanism advocated by Qin et al. [89] and the model of SR-ABE by Cui et al. [31].

The mechanism advocated by Qin et al. [89] is depicted in Figure 4.1. Specifically, when a new recipient joins the system, the KGC issues a private key and a corresponding token both associated with his identity and predicate. The former is given to the recipient and the latter is sent to the server. At each time period, the KGC issues an update key to the server who combines with the stored users' tokens to generate the transformation keys for users. A sender encrypts a message under an attribute and a time period and the ciphertext is sent to the untrusted server. The latter transforms the ciphertext to a partially decrypted ciphertext using a transformation key corresponding to the recipient's identity and the time period bound to the ciphertext. Finally, the recipient recovers the message from the partially decrypted ciphertext using his private key.

In comparison with Cui et al.'s model of SR-ABE [31], our model offers two crucial differences. In [31], it is assumed that each user in the system has to maintain a

public-secret key-pair (which can possibly be a key-pair for an ordinary PKE scheme). Although this setting can eliminate the need for a secure channel between the KGC and the users (as explained by Cui et al.), we find it somewhat unnatural in the context of identity-based/attribute-based/predicate cryptosystems. (After all, one of the main advantages of these systems over PKE systems is the elimination of users' public keys.) In contrast, our model of SR-PE does not require the users to maintain their own public keys. In the same spirit of IBE systems, we get rid of the notion of users' public keys and we assume a secure channel for transmitting users' private keys.

Another notable difference between our model and [31] is due to gap between security notions for ABE and PE systems. Our model preserves the attribute-hiding property of PE systems, which, unlike ABE systems, attributes bound to the ciphertexts are not revealed during decryptions.

Next, we will formally define the notion of SR-PE.

Definition 4.2.1. A sever-aided revocable predicate encryption scheme $\mathcal{SRPE} = (\text{Sys}, \text{Setup}, \text{PrivKG}, \text{Token}, \text{UpdKG}, \text{TranKG}, \text{Enc}, \text{Transform}, \text{Dec}, \text{Revoke})$ involves 4 parties: a KGC, senders, recipients, and an untrusted server. It is assumed that the server stores a list of tuples (identity, predicate, token), i.e., $(\text{id}, f, \tau_{\text{id},f})$. Algorithms among the parties are defined below.

$\text{Sys}(1^n)$ takes as input a security parameter n and outputs the system parameters params . This algorithm is run by the KGC.

$\text{Setup}(\text{params})$ takes as input the system parameters params , and outputs a revocation list RL (initially empty), a state st , a set of public parameters pp , and a master secret key msk . Throughout this chapter, pp is assumed to be an implicit input of all other algorithms.

$\text{PrivKG}(\text{msk}, \text{id}, f)$ takes as input the master secret key msk and an identity id with predicate f . It outputs a private key $\text{sk}_{\text{id},f}$. This algorithm is run by the KGC and sk_{id} is sent to the recipient id via a secure channel.

$\text{Token}(\text{msk}, \text{id}, f, \text{st})$ takes as input the master secret key msk , an identity id with a predicate f , and state st . It outputs the updated state st and a token $\tau_{\text{id},f}$. This algorithm is run by the KGC and $\tau_{\text{id},f}$ is sent to the server via a public channel.

$\text{UpdKG}(\text{msk}, \text{t}, \text{RL}, \text{st})$ takes as input the master secret key msk , a time period t , the list of revoked users RL , and state st . It outputs an update key uk_t . This algorithm is run by the KGC and uk_t is sent to the server via a public channel.

$\text{TranKG}(\text{id}, \tau_{\text{id},f}, \text{uk}_\text{t})$ takes as input an identity with the corresponding token $\tau_{\text{id},f}$ and an update key uk_t , and outputs a transformation key $\text{tk}_{\text{id},\text{t}}$ for user id at the time period t . This algorithm is run by the server.

$\text{Enc}(I, \text{t}, M)$ takes as input an attribute I , a time t , and a message M . It outputs a ciphertext ct_t which is publicly sent to the server. This algorithm is run by each sender.

$\text{Transform}(\text{ct}_\text{t}, \text{id}, \text{tk}_{\text{id},\text{t}})$ takes as input a ciphertext ct_t , and an identity with the corresponding transform key $\text{tk}_{\text{id},\text{t}}$. It outputs a partially decrypted ciphertext ct'_{id} . This algorithm is run by the server and ct'_{id} is sent to the recipient id via a public channel.

$\text{Dec}(\text{ct}'_{\text{id}}, \text{sk}_{\text{id},f})$ takes as input a partially decrypted ciphertext ct'_{id} and a private key $\text{sk}_{\text{id},f}$. It outputs a message M or the symbol $\perp$. This algorithm is run by each recipient.

$\text{Revoke}(\text{id}, \text{t}, \text{RL}, \text{st})$ takes as input an identity id to be revoked, a time t , the current list of revoked users RL , and state st . It updates the revocation list RL . This algorithm is run by the KGC.

Correctness. The correctness condition for an SR-PE scheme requires that: For any $n \in \mathbb{N}$, any revocation list RL , and all possible state st , if all parties follow the prescribed algorithms and if an identity id is not revoked at a time t , then:

1. If $f(I) = 1$ then $\text{Dec}(\text{ct}'_{\text{id}}, \text{sk}_{\text{id},f}) = M$.
2. If $f(I) = 0$ then $\text{Dec}(\text{ct}'_{\text{id}}, \text{sk}_{\text{id},f}) = \perp$, except with negligible probability.

Security. Next, we give the semantic security against selective attributes chosen plaintext attacks for server-aided revocable predicate encryption (shortened as **SR-sA-CPA**). The selective security means that the adversary needs to announce the challenge attributes and time period before seeing public parameters. In addition, it is assumed that the adversary must commit in advance the set of users to be revoked prior to the challenge time, which is similar to the semi-static query model considered in [10, 43].

Definition 4.2.2 (SR-sA-CPA Security). Let $\mathcal{A}$ be an adversary algorithm and let $I_0, I_1, \mathbf{t}^*$, and RL^* be the challenge attributes, time period, and revocation list, respectively. Let $\mathcal{O}$ be a set of oracles defined as follows:

- $\text{PrivKG}(\cdot, \cdot)$: When querying on an identity id and a predicate f , algorithm $\mathcal{A}$ is given a private key $\text{sk}_{\text{id},f}$ by running $\text{PrivKG}(\text{msk}, \text{id}, f)$.
- $\text{Token}(\cdot, \cdot)$: When querying on an identity id and a predicate f , algorithm $\mathcal{A}$ is given a token $\tau_{\text{id},f}$ obtained from the execution of $\text{Token}(\text{msk}, \text{id}, f, \text{st})$.
- $\text{UpdKG}(\cdot)$: When querying on a time period $\mathbf{t}$, algorithm $\mathcal{A}$ is given an update key $\text{uk}_{\mathbf{t}}$ by running algorithm $\text{UpdKG}(\text{msk}, \mathbf{t}, \text{RL}, \text{st})$. If $\mathbf{t} = \mathbf{t}^*$, then RL^* must be a subset of the RL at $\mathbf{t}^*$.
- $\text{Revoke}(\cdot, \cdot)$: When querying on an identity id and a time $\mathbf{t}$, algorithm $\mathcal{A}$ is given the updated RL by running $\text{Revoke}(\text{id}, \mathbf{t}, \text{RL}, \text{st})$. Note that this oracle cannot be queried on time $\mathbf{t}$ if $\text{UpdKG}(\cdot)$ has been queried on time $\mathbf{t}$.

An SR-PE scheme is **SR-sA-CPA** secure if any PPT adversary $\mathcal{A}$ has negligible advantage in the following experiment:

$$\boxed{\text{Exp}_{\mathcal{A}, \text{SRPE}}^{\text{SR-sA-CPA}}(n)}$$

$\text{params} \leftarrow \text{Sys}(1^n); I_0, I_1, \mathbf{t}^*, \text{RL}^* \leftarrow \mathcal{A}$

$(\text{pp}, \text{msk}, \text{st}, \text{RL}) \leftarrow \text{Setup}(\text{params})$

$M_0, M_1 \leftarrow \mathcal{A}^{\mathcal{O}}(\text{pp})$

$b \xleftarrow{\$} \{0, 1\}$

$\text{ct}^* \leftarrow \text{Enc}(I_b, \mathbf{t}^*, M_b)$

$b' \leftarrow \mathcal{A}^{\mathcal{O}}(\text{ct}^*)$

Return 1 if $b' = b$ and 0 otherwise.

Except the requirement that the two challenge messages M_0, M_1 have the same length, the following two restrictions must hold:

1. Case 1: if an identity id^* with predicate f^* satisfying that $f^*(I_0) = 1$ or $f^*(I_1) = 1$ has been queried to $\text{PrivKG}(\cdot, \cdot)$ and $\text{Token}(\cdot, \cdot)$, then id^* must be included in RL^* .
2. Case 2: if an identity id^* with predicate f^* satisfying that $f^*(I_0) = 1$ or $f^*(I_1) = 1$ is not revoked at $\mathbf{t}^*$, then (id^*, f^*) should not be queried to the $\text{PrivKG}(\cdot, \cdot)$ oracle.

The advantage of $\mathcal{A}$ in the experiment is defined as:

$$\text{Adv}_{\mathcal{A}, \text{SRPE}}^{\text{SR-sA-CPA}}(n) = \left| \Pr \left[\text{Exp}_{\mathcal{A}, \text{SRPE}}^{\text{SR-sA-CPA}}(n) = 1 \right] - \frac{1}{2} \right|.$$

Remark 4.2.3. We can also define an adaptive security notion, where the adversary is not required to specify the challenge attributes I_0, I_1 and time period $\mathbf{t}^*$ before seeing the public parameters pp . Such a notion is obviously stronger than the selective notion defined above.

4.3 A Lattices-based SR-PE Scheme

Our lattice-based SR-PE scheme can be seen as a combination of two AFV PE instances (see Scheme 2.3.17), one IBE instance (see Scheme 2.3.13) and the CS method (See Section 2.4.1). Each recipient's identity id corresponds to a matrix $\mathbf{D}_{\text{id}}$ determined by the IBE system. The KGC generates the private key for the first PE instance with a hierarchical level for $\mathbf{D}_{\text{id}}$, and issues the token by embedding $\mathbf{D}_{\text{id}}$ into the second PE scheme as well as using nodes in $\text{Path}(\text{id})$. At each time period $\mathbf{t}$, the KGC computes an update key using nodes in $\text{KUNodes}(\text{BT}, \text{RL}, \mathbf{t})$. Recall that token and update key are both sent to the sever, who makes use of the intersected node in $\text{Path}(\text{id}) \cap \text{KUNodes}(\text{BT}, \text{RL}, \mathbf{t})$ to obtain a transformation key. Then, a ciphertext in our scheme is a combination of two PE ciphertexts and an extra component bound to $\mathbf{t}$, where all components have the same randomness (i.e., vector $\mathbf{s}$). If recipient id is not revoked at time period $\mathbf{t}$, e.g., $\text{id} \notin \text{RL}$, then the server can partially decrypt the ciphertext, via the decryption algorithm of the second PE instance. Finally, the partially decrypted ciphertext contains a proper ciphertext for the first PE system and an additional component bound to matrix $\mathbf{D}_{\text{id}}$ (all with randomness $\mathbf{s}$) so that it can be fully decrypted using the private key of id (obtained from the first PE instance and specified by $\mathbf{D}_{\text{id}}$).

4.3.1 Description of the Scheme

Scheme 4.3.1. We will formally describe the algorithms in the scheme as follows.

Sys(1^n): Taking as input security parameter n , the KGC performs the following steps:

1. Choose $N = \text{poly}(n)$ as the maximal number of users the system will support, and arbitrary ℓ to be the length of predicate and attribute vectors. Choose $\kappa = \omega(\log n)$ as a dimension parameter.
2. Set $q = \tilde{O}(\ell^2 n^4)$ to be a prime, and let $m = 2n \lceil \log q \rceil$. Remark that the

parameters n, q, m specify the primitive matrix $\mathbf{G}$ (see Lemma 2.3.2).

3. Select a Gaussian parameter $s = \tilde{O}(\sqrt{m})$.
4. Let $B = \tilde{O}(\sqrt{n})$ and set χ to be a B -bounded LWE error distribution.
5. Choose a full-rank difference map $\mathbf{H} : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^{n \times n}$ (see Scheme 2.3.13).
6. Let the identity space be $\mathcal{I} \subseteq \mathbb{Z}_q^n$, the time space be $\mathcal{T} \subseteq \mathbb{Z}_q^n$, the message space be $\mathcal{M} = \{0, 1\}$, the predicate space be $\mathbb{P} = \{f_{\vec{x}} \mid \vec{x} \in \mathbb{Z}_q^\ell\}$ and the attribute space be $\mathbb{A} = \mathbb{Z}_q^\ell$.
7. Define the encoding function **encode** (see Section 2.3.4).
8. Return **params** = $(n, N, \ell, \kappa, q, m, s, B, \chi, \mathbf{H}, \mathcal{I}, \mathcal{T}, \mathcal{M}, \mathbb{P}, \mathbb{A}, \text{encode})$.

Setup(params): Taking as input the system parameters **params**, the KGC performs the following steps:

1. Generate two independent pairs $(\mathbf{A}, \mathbf{T}_\mathbf{A})$ and $(\mathbf{B}, \mathbf{T}_\mathbf{B})$ by running algorithm $\text{TrapGen}(n, q, m)$.
2. Choose $\mathbf{V} \xleftarrow{\$} \mathbb{Z}_q^{n \times \kappa}$ and $\mathbf{C}, \mathbf{D}, \mathbf{A}_i, \mathbf{B}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ for each $i \in [\ell]$.
3. Set the revocation list $\text{RL} := \emptyset$. Let the state $\text{st} := \text{BT}$ where BT is a binary tree with at least N leaf nodes.
4. Let $\text{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \{\mathbf{A}_i\}_{i \in [\ell]}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{V})$ and $\text{msk} = (\mathbf{T}_\mathbf{A}, \mathbf{T}_\mathbf{B})$.
5. Return $(\text{pp}, \text{msk}, \text{RL}, \text{st})$.

PrivKG(msk, id, $\vec{x}$): Taking as input the master secret key **msk** and an identity $\text{id} \in \mathcal{I}$ with predicate vector $\vec{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_q^\ell$, the KGC performs the following steps:

1. Set $\mathbf{B}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{B}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \in \mathbb{Z}_q^{n \times m}$ and $\mathbf{D}_{\text{id}} = \mathbf{D} + \mathbf{H}(\text{id})\mathbf{G} \in \mathbb{Z}_q^{n \times m}$.
2. Sample $\mathbf{Z} \leftarrow \text{SampleLeft}(\mathbf{B}, [\mathbf{B}_{\vec{x}} \mid \mathbf{D}_{\text{id}}], \mathbf{T}_\mathbf{B}, \mathbf{V}, s)$. Note that $\mathbf{Z} \in \mathbb{Z}_q^{3m \times \kappa}$ and $[\mathbf{B} \mid \mathbf{B}_{\vec{x}} \mid \mathbf{D}_{\text{id}}] \cdot \mathbf{Z} = \mathbf{V}$.

3. Return $\text{sk}_{\text{id}, \vec{x}} = \mathbf{Z}$.

Token(msk, id, $\vec{x}$, st): Taking as input the master secret key msk , an identity $\text{id} \in \mathcal{I}$ with predicate vector $\vec{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_q^\ell$, and state st , the KGC performs the following steps:

1. Compute $\mathbf{A}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{A}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \in \mathbb{Z}_q^{n \times m}$.
2. For each $\theta \in \text{Path}(\text{id})$, if $\mathbf{U}_\theta$ is undefined, then pick $\mathbf{U}_\theta \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and store it in node θ ; Sample $\mathbf{Z}_{1,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_{\vec{x}}, \mathbf{T}_\mathbf{A}, \mathbf{D}_{\text{id}} - \mathbf{U}_\theta, s)$. Note that $\mathbf{Z}_{1,\theta} \in \mathbb{Z}^{2m \times m}$ and $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z}_{1,\theta} = \mathbf{D}_{\text{id}} - \mathbf{U}_\theta$.
3. Return the updated state st and $\tau_{\text{id}, \vec{x}} = \{\theta, \mathbf{Z}_{1,\theta}\}_{\theta \in \text{Path}(\text{id})}$.

UpdKG(msk, t, st, RL): Taking as input the master secret key msk , a time $t \in \mathcal{T}$, the revocation list RL , and state st , the KGC performs the following steps:

1. Compute $\mathbf{C}_t = \mathbf{C} + \mathbf{H}(t)\mathbf{G} \in \mathbb{Z}_q^{n \times m}$.
2. For each $\theta \in \text{KUNodes}(\text{BT}, \text{RL}, t)$, retrieve $\mathbf{U}_\theta$ (which is always pre-defined in algorithm **Token**), and sample $\mathbf{Z}_{2,\theta} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{C}_t, \mathbf{T}_\mathbf{A}, \mathbf{U}_\theta, s)$. Note that $\mathbf{Z}_{2,\theta} \in \mathbb{Z}^{2m \times m}$ and $[\mathbf{A} \mid \mathbf{C}_t] \cdot \mathbf{Z}_{2,\theta} = \mathbf{U}_\theta$.
3. Return $\text{uk}_t = \{\theta, \mathbf{Z}_{2,\theta}\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL}, t)}$.

TranKG(id, $\tau_{\text{id}, \vec{x}}$, uk_t): Taking as input an identity id with token $\tau_{\text{id}, \vec{x}} = \{\theta, \mathbf{Z}_{1,\theta}\}_{\theta \in I}$ and an update key $\text{uk}_t = \{\theta, \mathbf{Z}_{2,\theta}\}_{\theta \in J}$ for some set of nodes I and J , the server performs the following steps:

1. If $I \cap J = \emptyset$, output $\perp$.
2. Otherwise, choose $\theta \in I \cap J$ and output $\text{tk}_{\text{id}, t} = (\mathbf{Z}_{1,\theta}, \mathbf{Z}_{2,\theta})$. Note that $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z}_{1,\theta} + [\mathbf{A} \mid \mathbf{C}_t] \cdot \mathbf{Z}_{2,\theta} = \mathbf{D}_{\text{id}}$.

Enc($\vec{y}$, t, M): Taking as input an attribute vector $\vec{y} = (y_1, \dots, y_\ell) \in \mathbb{Z}_q^\ell$, a time $t \in \mathcal{T}$, and a message $M \in \mathcal{M}$, the sender performs the following steps:

1. Sample $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_1, \mathbf{e}_2 \leftarrow \chi^m$ and $\mathbf{e} \leftarrow \chi^\kappa$.
2. Choose $\bar{\mathbf{R}}, \mathbf{S}_i, \mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for each $i \in [\ell]$.
3. Output $\mathbf{ct}_t = (\mathbf{c}, \mathbf{c}_1, \{\mathbf{c}_{1,i}\}_{i \in [\ell]}, \mathbf{c}_{1,0}, \mathbf{c}_2, \{\mathbf{c}_{2,i}\}_{i \in [\ell]})$ where:

$$\left\{ \begin{array}{l} \mathbf{c} = \mathbf{V}^\top \mathbf{s} + \mathbf{e} + \text{encode}(M) \cdot \left\lfloor \frac{q}{2} \right\rfloor \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_1 = \mathbf{A}^\top \mathbf{s} + \mathbf{e}_1 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{1,i} = (\mathbf{A}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{R}_i^\top \mathbf{e}_1 \in \mathbb{Z}_q^m, \quad \forall i \in [\ell] \\ \mathbf{c}_{1,0} = \mathbf{C}_t^\top \mathbf{s} + \bar{\mathbf{R}}^\top \mathbf{e}_1 \in \mathbb{Z}_q^m, \\ \mathbf{c}_2 = \mathbf{B}^\top \mathbf{s} + \mathbf{e}_2 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{2,i} = (\mathbf{B}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{S}_i^\top \mathbf{e}_2 \in \mathbb{Z}_q^m, \quad \forall i \in [\ell]. \end{array} \right.$$

Transform($\mathbf{ct}_t, \text{id}, \mathbf{tk}_{\text{id},t}$): On input $\mathbf{ct}_t = (\mathbf{c}, \mathbf{c}_1, \{\mathbf{c}_{1,i}\}_{i \in [\ell]}, \mathbf{c}_{1,0}, \mathbf{c}_2, \{\mathbf{c}_{2,i}\}_{i \in [\ell]})$ and an identity id with transformation key $\mathbf{tk}_{\text{id},t} = (\mathbf{Z}_1, \mathbf{Z}_2)$, the server performs the following steps:

1. Set $\mathbf{c}_{1,\vec{x}} = \sum_{i=1}^{\ell} (\mathbf{G}^{-1}(x_i \cdot \mathbf{G}))^\top \mathbf{c}_{1,i} \in \mathbb{Z}_q^m$.
2. Compute $\bar{\mathbf{c}} = \mathbf{Z}_1^\top (\mathbf{c}_1 \parallel \mathbf{c}_{1,\vec{x}}) + \mathbf{Z}_2^\top (\mathbf{c}_1 \parallel \mathbf{c}_{1,0}) \in \mathbb{Z}_q^\kappa$.
3. Return $\mathbf{ct}'_{\text{id}} = (\mathbf{c}, \mathbf{c}_2, \{\mathbf{c}_{2,i}\}_{i \in [\ell]}, \bar{\mathbf{c}})$.

Dec($\mathbf{ct}'_{\text{id}}, \mathbf{sk}_{\text{id},\vec{x}}$): Taking as input $\mathbf{ct}'_{\text{id}} = (\mathbf{c}, \mathbf{c}_2, \{\mathbf{c}_{2,i}\}_{i \in [\ell]}, \bar{\mathbf{c}})$ and a private key $\mathbf{sk}_{\text{id},\vec{x}} = \mathbf{Z}$, the recipient performs the following steps:

1. Compute $\mathbf{c}_{2,\vec{x}} = \sum_{i=1}^{\ell} (\mathbf{G}^{-1}(x_i \cdot \mathbf{G}))^\top \mathbf{c}_{2,i} \in \mathbb{Z}_q^m$.
2. Compute $\mathbf{d} = \mathbf{c} - \mathbf{Z}^\top (\mathbf{c}_2 \parallel \mathbf{c}_{2,\vec{x}} \parallel \bar{\mathbf{c}}) \in \mathbb{Z}_q^\kappa$.
3. If $\left\lfloor \frac{2}{q} \cdot \mathbf{d} \right\rfloor = \text{encode}(M')$, for some $M' \in \mathcal{M}$, then output M' . Otherwise, output symbol $\perp$.

Revoke($\text{id}, t, \text{RL}, \text{st}$): Taking as input an identity id , a time t , the revocation list RL

and state $\mathbf{st} = \mathbf{BT}$, the KGC returns an update RL by adding $(\text{id}, \mathbf{t})$ for all nodes associated with identity id .

4.3.2 Correctness and Efficiency

Correctness. In the following, we will show that Scheme 4.3.1 satisfies the correctness requirement except with negligible probability. We proceed as in [4, 38, 66, 102].

Suppose that $\mathbf{ct}_{\mathbf{t}} = (\mathbf{c}, \mathbf{c}_1, \{\mathbf{c}_{1,i}\}_{i \in [\ell]}, \mathbf{c}_{1,0}, \mathbf{c}_2, \{\mathbf{c}_{2,i}\}_{i \in [\ell]})$ is a correctly computed ciphertext of message $M \in \mathcal{M}$, bound with some $\vec{y} \in \mathbb{A}$. Let $\mathbf{tk}_{\text{id}, \mathbf{t}} = (\mathbf{Z}_1, \mathbf{Z}_2)$ be an honestly generated transformation key, where id is not revoked at time $\mathbf{t}$. Then we have $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z}_1 + [\mathbf{A} \mid \mathbf{C}_{\mathbf{t}}] \cdot \mathbf{Z}_2 = \mathbf{D}_{\text{id}}$. We also observe that the following two equations hold:

$$\begin{aligned} \mathbf{c}_{1, \vec{x}} &= \sum_{i=1}^{\ell} \left(\mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \right)^{\top} \mathbf{c}_{1,i} = (\mathbf{A}_{\vec{x}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G})^{\top} \mathbf{s} + (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1, \\ \mathbf{c}_{2, \vec{x}} &= \sum_{i=1}^{\ell} \left(\mathbf{G}^{-1}(x_i \cdot \mathbf{G}) \right)^{\top} \mathbf{c}_{2,i} = (\mathbf{B}_{\vec{x}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G})^{\top} \mathbf{s} + (\mathbf{S}_{\vec{x}})^{\top} \mathbf{e}_2. \end{aligned}$$

where $\mathbf{R}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{R}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$ and $\mathbf{S}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{S}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$. We now consider two cases:

1. Case 1: Suppose that $\langle \vec{x}, \vec{y} \rangle = 0$. Then in this case, the following holds:

$\mathbf{c}_{1, \vec{x}} = (\mathbf{A}_{\vec{x}})^{\top} \mathbf{s} + (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1$ and $\mathbf{c}_{2, \vec{x}} = (\mathbf{B}_{\vec{x}})^{\top} \mathbf{s} + (\mathbf{S}_{\vec{x}})^{\top} \mathbf{e}_2$. Then in Transform algorithm, the following holds:

$$\begin{aligned} \bar{\mathbf{c}} &= \mathbf{Z}_1^{\top} (\mathbf{c}_1 \parallel \mathbf{c}_{1, \vec{x}}) + \mathbf{Z}_2^{\top} (\mathbf{c}_1 \parallel \mathbf{c}_{1,0}) \\ &= \mathbf{Z}_1^{\top} \left([\mathbf{A} \mid \mathbf{A}_{\vec{x}}]^{\top} \mathbf{s} + (\mathbf{e}_1 \parallel (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1) \right) + \mathbf{Z}_2^{\top} \left([\mathbf{A} \mid \mathbf{C}_{\mathbf{t}}]^{\top} \mathbf{s} + (\mathbf{e}_1 \parallel \bar{\mathbf{R}}^{\top} \mathbf{e}_1) \right) \\ &= \mathbf{D}_{\text{id}}^{\top} \mathbf{s} + \underbrace{\mathbf{Z}_1^{\top} (\mathbf{e}_1 \parallel (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}_1) + \mathbf{Z}_2^{\top} (\mathbf{e}_1 \parallel \mathbf{S}^{\top} \mathbf{e}_1)}_{\text{error}'} \end{aligned}$$

and in Dec algorithm, the following holds:

$$\begin{aligned}
\mathbf{d} &= \mathbf{c} - \mathbf{Z}^\top [\mathbf{c}_2 \mid \mathbf{c}_{2,\vec{x}} \mid \bar{\mathbf{c}}] \\
&= \mathbf{V}^\top \mathbf{s} + \mathbf{e} + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) - \mathbf{Z}^\top \left([\mathbf{B} \mid \mathbf{B}_{\vec{x}} \mid \mathbf{D}_{\text{id}}]^\top \mathbf{s} + (\mathbf{e}_2 \| (\mathbf{S}_{\vec{x}})^\top \mathbf{e}_2 \| \text{error}') \right) \\
&= \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) + \underbrace{\mathbf{e} - \mathbf{Z}^\top (\mathbf{e}_2 \| (\mathbf{S}_{\vec{x}})^\top \mathbf{e}_2 \| \text{error}')}_{\text{error}}.
\end{aligned}$$

As in [2, 4, 38, 66, 102], it can be shown that the error term **error** is bounded by $s\ell m^2 B \cdot \omega(\log n) = \tilde{O}(\ell^2 n^3)$, except with negligible probability. In the algorithm Dec, to recover $\text{encode}(M)$ and subsequently the message M , it requires that the error term should be bounded by $q/5$, i.e., $\|\text{error}\|_\infty < q/5$, which is ensured by the setting of prime modulus q , i.e., $q = \tilde{O}(\ell^2 n^4)$.

2. Case 2: Suppose that $\langle \vec{x}, \vec{y} \rangle \neq 0$. Then in this case, the following holds:

$$\mathbf{c}_{2,\vec{x}} = \left(\mathbf{A}_{\vec{x}} + \underbrace{\langle \vec{x}, \vec{y} \rangle}_{\neq 0} \cdot \mathbf{G} \right)^\top \mathbf{s} + (\mathbf{S}_{\vec{x}})^\top \mathbf{e}_2.$$

Then in Dec algorithm, $\mathbf{d} = \mathbf{c} - \mathbf{Z}^\top [\mathbf{c}_2 \mid \mathbf{c}_{2,\vec{x}} \mid \bar{\mathbf{c}}]$ contains the following term:

$$\mathbf{Z}^\top [\mathbf{0} \mid \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G} \mid \mathbf{0}]^\top \mathbf{s} \in \mathbb{Z}_q^\kappa.$$

which can be written as $\langle \vec{x}, \vec{y} \rangle \cdot (\mathbf{G}\mathbf{Z}^2)^\top \mathbf{s}$, where $\mathbf{Z}^2 \in \mathbb{Z}^{m \times \kappa}$ is the middle m rows of matrix $\mathbf{Z}$. From Lemma 2.3.4, we have that the distribution of $\mathbf{G}\mathbf{Z}^2 \in \mathbb{Z}_q^{n \times \kappa}$ is indistinguishable from uniform. It then follows that, vector $\mathbf{d} \in \mathbb{Z}_q^\kappa$ in Dec algorithm, is statistically close to uniform. Therefore, the probability that all of the last $\kappa - 1$ coordinates of vector $\lfloor \frac{2}{q} \cdot \mathbf{d} \rfloor$ are zeros is at most $2^{-(\kappa-1)} = 2^{-\omega(\log n)}$, which is a negligible function in n . More specifically, with all but negligible probability, algorithm Dec will output symbol $\perp$ since it cannot retrieve a proper encoding $\text{encode}(M') \in \{0, 1\}^\kappa$, for any $M' \in \{0, 1\}$.

Efficiency. The efficiency aspect of the SR-PE in Scheme 4.3.1 is as follows:

- ◇ The public parameters $\mathbf{pp}$ has bit-size is $((2\ell + 4)nm + n\kappa) \log q = \tilde{O}(\ell) \cdot \tilde{O}(n^2)$.
- ◇ The bit-size of the private key $\mathbf{sk}_{\text{id}, \vec{x}}$ is bit-size $\tilde{O}(n)$.
- ◇ The token $\tau_{\text{id}, \vec{x}}$ has bit-size $O(\log N) \cdot \tilde{O}(n^2)$.
- ◇ The bit-size of the update key $\mathbf{uk}_t$ is $O\left(r \log \frac{N}{r}\right) \cdot \tilde{O}(n^2)$.
- ◇ The bit-size of the ciphertext $\mathbf{ct}_t$ is $\tilde{O}(\ell n)$.
- ◇ The bit-size of the partially decrypted ciphertext $\mathbf{ct}'_{\text{id}}$ is $\tilde{O}(\ell n)$.

4.3.3 Security

We prove that our SR-PE in Scheme 4.3.1 is **SR-sA-CPA** secure in the standard model, based on hardness of the **LWE** problem.

Theorem 4.3.2. Our SR-PE in Scheme 4.3.1 satisfies the security notion of **SR-sA-CPA** (see Definition 4.2.2), provided hardness of the (n, q, χ) -**LWE** problem.

Proof Outline. We first give the high-level idea of the security proof, while the formal one is given later.

Recall that our SR-IBE scheme can be seen as a combination of two AFV PE systems (in Scheme 2.3.17), which are **wAH-sA-CPA** secure (see definition 2.3.16) based on the **LWE** assumption. We consider two distinct types of adversaries as follows.

Type I Adversary: It is assumed that every identity id^* whose predicate vector $\vec{x}^*$ satisfies that $\langle \vec{x}^*, \vec{y}_0 \rangle = 0$ or $\langle \vec{x}^*, \vec{y}_1 \rangle = 0$, must be included in RL^* . In this case, the adversary is allowed to issue a query to oracle $\text{PrivKG}(\cdot, \cdot)$ on such a pair $(\text{id}^*, \vec{x}^*)$.

Type II Adversary: It is assumed that there exists an $\text{id}^* \notin \text{RL}^*$ whose predicate vector $\vec{x}^*$ satisfies that $\langle \vec{x}^*, \vec{y}_0 \rangle = 0$ or $\langle \vec{x}^*, \vec{y}_1 \rangle = 0$. In this case, id^* may

be not revoked at $\mathbf{t}^*$ and the adversary is not allowed to issue a query to oracle $\text{PrivKG}(\cdot, \cdot)$ on $(\text{id}^*, \vec{x}^*)$.

Here $\vec{y}_0, \vec{y}_1$ are the challenge attribute vectors, $\mathbf{t}^*$ is the challenge time and RL^* is the set of revoked users at $\mathbf{t}^*$.

We then separately show that, for both types of adversaries, the security of our SR-PE scheme can be reduced from the security of the AFV PE (see Theorem 2.3.18).

Proof. We will proceed similarly as in the security proof for our SR-IBE in Chapter 3. Namely, we will show that: for our SR-PE, if there exists an efficient algorithm $\mathcal{A}$ successfully winning the security game in Definition 4.2.2, then it is possible to use $\mathcal{A}$ to build an efficient algorithm $\mathcal{S}$ winning the security game defined in Definition 2.3.16 for the underlying AFV PE scheme. It then follows the theorem holds since the building block is wAH-sA-CPA secure based on the hardness of the LWE problems (see Theorem 2.3.18).

Assume that $\mathbf{t}^*$ is the challenge time, vectors $\vec{y}_0, \vec{y}_1$ are the challenge attributes, and RL^* is the set of revoked users at $\mathbf{t}^*$. Without loss of generality, we can suppose that $\mathcal{A}$ will make token or private key queries on identities whose predicates are satisfied by $\vec{y}_0$ or $\vec{y}_1$. We then divide adversaries into two distinct types, stated in the following.

Type I Adversary: It is assumed that, every identity id^* whose predicate vector $\vec{x}^*$ satisfies that $\langle \vec{x}^*, \vec{y}_0 \rangle = 0$ or $\langle \vec{x}^*, \vec{y}_1 \rangle = 0$, must be included in RL^* . In this case, algorithm $\mathcal{A}$ may access to oracle $\text{PrivKG}(\cdot, \cdot)$ on such a pair $(\text{id}^*, \vec{x}^*)$.

Type II Adversary: It is assumed that there exists an $\text{id}^* \notin \text{RL}^*$ whose predicate vector $\vec{x}^*$ satisfies that $\langle \vec{x}^*, \vec{y}_0 \rangle = 0$ or $\langle \vec{x}^*, \vec{y}_1 \rangle = 0$. In this case, id^* may be not revoked at $\mathbf{t}^*$ and $\mathcal{A}$ is forbidden to access to oracle $\text{PrivKG}(\cdot, \cdot)$ on $(\text{id}^*, \vec{x}^*)$.

Next, we separately show how to construct the algorithm $\mathcal{S}$, which acts as not only the adversary in the underlying PE but also the challenger in our SR-PE. $\mathcal{S}$'s process

starts by randomly guessing that it will be faced with Type I adversary or Type II adversary. In both cases, let Q be the number of users in RL^* .

Lemma 4.3.3. Suppose that $\mathcal{A}$ is a PPT Type I adversary. If $\mathcal{A}$ wins the SR-sA-CPA security experiment for our SR-PE scheme with advantage ϵ , then for the AFV PE scheme, there exists an efficient algorithm $\mathcal{S}$ winning the wAH-sA-CPA security experiment with advantage ϵ/Q .

Proof. Recall that if an identity id has the predicate vector $\vec{x}$ satisfied by the challenge attributes $\vec{y}_0$ or $\vec{y}_1$, it must be include in RL^* . The simulator $\mathcal{S}$ randomly choose $j^* \xleftarrow{\$} [Q]$, at which such an identity appears. Let id^* be the j^* -th user in RL^* and $\vec{x}^*$ be the corresponding predicate vector.

Assume that $\mathcal{B}$ is the challenger in the wAH-sA-CPA security experiment for the underlying PE scheme. In the following, we describe $\mathcal{S}$'s process, by interacting with the adversary $\mathcal{A}$ for our SR-PE and algorithm $\mathcal{B}$.

Initial: $\mathcal{S}$ outputs params by running algorithm $\text{Sys}(1^n)$. Then $\mathcal{S}$ is announced by $\mathcal{A}$ the challenge time $\mathbf{t}^*$, attribute vectors $\vec{y}_0, \vec{y}_1$, and revocation list RL^* . Algorithm $\mathcal{S}$ forwards $\vec{y}_0, \vec{y}_1$ to $\mathcal{B}$.

Setup: $\mathcal{S}$ initialize the revocation list as $\text{RL} := \emptyset$ and the sate as $\text{st} := \text{BT}$. It then prepares the public parameters pp by performing the following steps:

1. Get $\text{pp}_{\text{PE}} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{V})$ from $\mathcal{B}$, where $\mathbf{A}, \mathbf{A}_i \in \mathbb{Z}_q^{n \times m}$, $\mathbf{V} \in \mathbb{Z}_q^{n \times \kappa}$.
2. Run algorithm $\text{TrapGen}(n, q, m)$ to generate $(\mathbf{B}, \mathbf{T}_{\mathbf{B}})$. Pick $\mathbf{D}, \mathbf{B}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ for each $i \in [\ell]$.
3. Select $\bar{\mathbf{R}} \xleftarrow{\$} \{-1, 1\}^{m \times m}$ and let $\mathbf{C} = \mathbf{A}\bar{\mathbf{R}} - \text{H}(\mathbf{t}^*)\mathbf{G} \in \mathbb{Z}_q^{n \times m}$.
4. Let $\text{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \{\mathbf{A}_i\}_{i \in [\ell]}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{V})$, which is sent the adversary $\mathcal{A}$.

It is not difficult to see that all the matrices included in pp are uniformly random or statistically close to uniformly random, which implies that in $\mathcal{A}$'s

view, the distribution of the $\mathbf{pp}$ prepared by $\mathcal{S}$ is statistically indistinguishable from the one generated in the real scheme.

Private Key Oracle: When $\mathcal{A}$ issues a private key query, $\mathcal{S}$ can perform the same as in our SR-PE scheme as it knows $\mathbf{T}_{\mathbf{B}}$, which is the master secret key part corresponding to the private key oracle.

Token and Update Key Oracles: The simulator first defines $\mathbf{U}_\theta$ for each $\theta \in \mathbf{BT}$ as follows:

1. If $\theta \in \text{Path}(\text{id}^*)$, pick $\mathbf{Z}_{1,\theta} \leftarrow \mathcal{D}_{\mathbb{Z}^{2m \times m}, s}$ and set $\mathbf{U}_\theta = \mathbf{D}_{\text{id}^*} - [\mathbf{A} | \mathbf{A}_{\vec{x}^*}] \cdot \mathbf{Z}_{1,\theta}$.
2. If $\theta \notin \text{Path}(\text{id}^*)$, pick $\mathbf{Z}_{2,\theta} \leftarrow \mathcal{D}_{\mathbb{Z}^{2m \times m}, s}$ and set $\mathbf{U}_\theta = [\mathbf{A} | \mathbf{C}_{\mathbf{t}^*}] \cdot \mathbf{Z}_{2,\theta}$.

If $\mathcal{A}$ queries a token for $(\text{id}, \vec{x})$ such that $\langle \vec{x}, \vec{y}_0 \rangle \neq 0$ and $\langle \vec{x}, \vec{y}_1 \rangle \neq 0$, algorithm $\mathcal{S}$ forwards $\vec{x}$ to the PE challenger $\mathcal{B}$. Receiving a PE private key $\mathbf{T}_{\vec{x}}$ from $\mathcal{B}$, algorithm $\mathcal{S}$ can perform as in the real scheme except that the sampling algorithm $\text{Sample}([\mathbf{A} | \mathbf{A}_{\vec{x}}], \mathbf{T}_{\vec{x}}, \mathbf{D}_{\text{id}} - \mathbf{U}_\theta, s)$ replaces algorithm SampleLeft .

If $\mathcal{A}$ makes a token query on $(\text{id}, \vec{x}) \neq (\text{id}^*, \vec{x}^*)$ together with $\langle \vec{x}, \vec{y}_0 \rangle = 0$ or $\langle \vec{x}, \vec{y}_1 \rangle = 0$, the simulator returns $\perp$. For the query on $(\text{id}^*, \vec{x}^*)$, it returns $\{\theta, \mathbf{Z}_{1,\theta}\}_{\theta \in \text{Path}(\text{id}^*)}$ as defined above. Since the specific id^* is unknown in $\mathcal{A}$'s view, $\mathcal{S}$ can simulate successfully with probability at least $1/Q$.

For update key of $\mathbf{t} \neq \mathbf{t}^*$, note $\mathbf{C}_{\mathbf{t}} = \mathbf{C} + \mathbf{H}(\mathbf{t})\mathbf{G} = \mathbf{A}\bar{\mathbf{R}} + (\mathbf{H}(\mathbf{t}) - \mathbf{H}(\mathbf{t}^*))\mathbf{G}$. Algorithm $\mathcal{S}$ can compute $\mathbf{uk}_{\mathbf{t}}$ as in the real scheme except that algorithm $\text{SampleRight}(\mathbf{A}, \bar{\mathbf{R}}, (\mathbf{H}(\mathbf{t}) - \mathbf{H}(\mathbf{t}^*))\mathbf{G}), \mathbf{T}_{\mathbf{G}}, \mathbf{U}_\theta, s)$ replaces algorithm SampleLeft . For the target $\mathbf{t}^*$, algorithm $\mathcal{S}$ returns $\{\theta, \mathbf{Z}_{2,\theta}\}_{\theta \in \text{KUNodes}(\mathbf{BT}, \mathbf{RL}, \mathbf{t}^*)}$ as defined above since the set $\text{KUNodes}(\mathbf{BT}, \mathbf{RL}, \mathbf{t}^*)$ for $\mathbf{t}^*$ is disjoint with $\text{Path}(\text{id}^*)$.

Note that, in the real scheme each column of matrices $\mathbf{Z}_{1,\theta}, \mathbf{Z}_{2,\theta}$ are sampled by running algorithm SampleLeft . Meanwhile, $\mathcal{S}$ simulates these matrices either by running algorithms SampleRight , SamplePre or by sampling from distribution

$\mathcal{D}_{\mathbb{Z}^m, s}$. The distribution of them is all statistically close to discrete Gaussians (see Section 2.3.1), which ensures that the distribution of matrices $\mathbf{Z}_{1,\theta}, \mathbf{Z}_{2,\theta}$ in the simulation and the one from the real scheme are statistically close.

Challenge: After receiving two messages $M_0, M_1 \in \mathcal{M}$ from $\mathcal{A}$, algorithm $\mathcal{S}$ prepares the challenge ciphertext $\mathbf{ct}^*$ by performing the following steps:

1. Sample $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{e}_2 \leftarrow \chi^m$. Choose $\mathbf{S}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for each $i \in [\ell]$.
2. Pick $d \xleftarrow{\$} \{0, 1\}$. Set $M'_0 = M_d$, $M'_1 = M_{1 \oplus d}$, where $\oplus$ denotes the addition modulus 2. Forward M'_0, M'_1 as two challenge messages to the PE challenger $\mathcal{B}$. The latter chooses $c \xleftarrow{\$} \{0, 1\}$ and returns a ciphertext $(\mathbf{c}', \mathbf{c}'_0, \{\mathbf{c}'_i\}_{i \in [\ell]})$ as a PE encryption of M'_c under attribute vector $\vec{y}_c$.
3. Output $\mathbf{ct}^* = (\mathbf{c}^*, \mathbf{c}_1^*, \{\mathbf{c}_{1,i}^*\}_{i \in [\ell]}, \mathbf{c}_{1,0}^*, \mathbf{c}_2^*, \{\mathbf{c}_{2,i}^*\}_{i \in [\ell]})$ as an SR-PE encryption of M_d under $\vec{y}_d$ and $\mathbf{t}^*$, where:

$$\left\{ \begin{array}{l} \mathbf{c}^* = \mathbf{c}' \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_1^* = \mathbf{c}'_0 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{1,i}^* = \mathbf{c}'_i \in \mathbb{Z}_q^m, \quad \forall i \in [\ell] \\ \mathbf{c}_{1,0}^* = \bar{\mathbf{R}}^\top \mathbf{c}'_0 \in \mathbb{Z}_q^m, \\ \mathbf{c}_2^* = \mathbf{B}^\top \mathbf{s} + \mathbf{e}_2 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{2,i}^* = (\mathbf{B}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{S}_i^\top \mathbf{e}_2 \in \mathbb{Z}_q^m, \quad \forall i \in [\ell]. \end{array} \right.$$

Guess: $\mathcal{A}$ may continue to make additional queries to oracles PrivKG, Token and UpdKG. After $\mathcal{A}$ stops querying these oracles, it outputs a bit d' which serves as the guess that the challenge ciphertext $\mathbf{ct}^*$ is an encryption of $M_{d'}$ under $\vec{y}_{d'}$ and $\mathbf{t}^*$. Then setting $c' = d \oplus d'$, algorithm $\mathcal{S}$ returns c' to the PE challenger $\mathcal{B}$, which serves as the guess for $c \in \{0, 1\}$. Recall that c is the bit chosen by $\mathbf{B}$.

On the one hand, it is assumed that for our SR-PE scheme, the adversary $\mathcal{A}$ wins

the SR-sA-CPA security experiment with advantage ϵ , which implies that

$$\text{Adv}_{\mathcal{A}}^{\text{SR-sA-CPA}}(n) = \left| \Pr[d' = d \oplus c] - \frac{1}{2} \right| = \epsilon.$$

On the other hand, in the simulation, it holds that: $d' = d \oplus c \Leftrightarrow d' \oplus d = c \Leftrightarrow c' = c$.

We then have

$$\text{Adv}_{\mathcal{S}, \text{PE}}^{\text{wAH-sA-CPA}}(n) = \left| \Pr[c = c'] - \frac{1}{2} \right| = \epsilon/Q.$$

□

Lemma 4.3.4. Suppose that $\mathcal{A}$ is a PPT Type II adversary. If $\mathcal{A}$ wins the SR-sA-CPA security experiment for our SR-PE scheme with advantage ϵ , then for the AFV PE scheme, there exists an efficient algorithm $\mathcal{S}$ winning the wAH-sA-CPA security experiment with the same advantage.

Proof. Recall that there is an identity id^* whose predicate is satisfied by $\vec{y}_0$ or $\vec{y}_1$ and it is not included in RL^* .

Assume that $\mathcal{B}$ is the challenger in the wAH-sA-CPA security experiment for the underlying PE scheme. In the following, we describe $\mathcal{S}$'s process, by interacting with the adversary $\mathcal{A}$ for our SR-PE scheme and algorithm $\mathcal{B}$.

Initial: $\mathcal{S}$ outputs **params** by running algorithm $\text{Sys}(1^n)$. Then $\mathcal{S}$ is announced by $\mathcal{A}$ the challenge time $\mathbf{t}^*$, attribute vectors $\vec{y}_0, \vec{y}_1$, and revocation list RL^* . Algorithm $\mathcal{S}$ forwards $\vec{y}_0, \vec{y}_1$ to $\mathcal{B}$.

Setup: $\mathcal{S}$ initializes the revocation list as $\text{RL} := \emptyset$ and the state as $\text{st} := \text{BT}$. It then prepares the public parameters **pp** by performing the following steps:

1. Receive $\text{pp}_{\text{PE}} = (\mathbf{B}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{V})$ from $\mathcal{B}$, where $\mathbf{B}, \mathbf{B}_i \in \mathbb{Z}_q^{n \times m}$, $\mathbf{V} \in \mathbb{Z}_q^{n \times \kappa}$.
2. Run algorithm $\text{TrapGen}(n, q, m)$ to generate $(\mathbf{A}, \mathbf{T}_{\mathbf{A}})$ by. Select $\mathbf{C}, \mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ for each $i \in [\ell]$.

3. Select $\bar{\mathbf{S}} \xleftarrow{\$} \{-1, 1\}^{m \times m}$ and let $\mathbf{D} = \mathbf{B}\bar{\mathbf{S}} - \mathbf{H}(\text{id}^*)\mathbf{G}$.
4. Let the public parameters be $\mathbf{pp} = (\mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{D}, \{\mathbf{A}_i\}_{i \in [\ell]}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{V})$ and send $\mathbf{pp}$ to the adversary $\mathcal{A}$. Similar as in previous case, all the matrices included in $\mathbf{pp}$ are uniformly random or statistically close to uniformly random, such that in $\mathcal{A}$'s view, the distribution of the $\mathbf{pp}$ prepared by $\mathcal{S}$ is statistically indistinguishable from the one generated in the real scheme.

Private Key Oracle: $\mathcal{A}$ is not allowed to issue a private key query for id^* . When $\mathcal{A}$ makes a query to $\text{PrivKG}(\cdot, \cdot)$ oracle on $(\text{id}, \vec{x})$ such that $\text{id} \neq \text{id}^*$, $\mathcal{S}$ returns $\mathbf{Z}$ by running $\text{SampleRight}([\mathbf{B} \mid \mathbf{B}_{\vec{x}}], \bar{\mathbf{S}}, (\mathbf{H}(\text{id}) - \mathbf{H}(\text{id}^*))\mathbf{G}, \mathbf{V}, s)$.

Token and Update Key Oracles: When $\mathcal{A}$ issues a private key or update query, $\mathcal{S}$ can perform the same as in our SR-PE scheme as it knows $\mathbf{T}_{\mathbf{B}}$, which is the master secret key part corresponding to these two oracles.

Challenge: After receiving two messages $M_0, M_1 \in \mathcal{M}$ from $\mathcal{A}$, algorithm $\mathcal{S}$ prepares the challenge ciphertext ct^* by performing the following steps:

1. Sample $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}_1 \leftarrow \chi^m$. Choose $\bar{\mathbf{R}}, \mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for each $i \in [\ell]$.
2. Pick $d \xleftarrow{\$} \{0, 1\}$ and set $M'_0 = M_d, M'_1 = M_{1 \oplus d}$. Forward M'_0, M'_1 as two challenge messages to the PE challenger $\mathcal{B}$. The latter chooses $c \xleftarrow{\$} \{0, 1\}$ and returns $(\mathbf{c}', \mathbf{c}'_0, \{\mathbf{c}'_i\}_{i \in [\ell]})$ as a PE encryption of M'_c under $\vec{y}_c$.
3. Output $\text{ct}^* = (\mathbf{c}^*, \mathbf{c}_1^*, \{\mathbf{c}_{1,i}^*\}_{i \in [\ell]}, \mathbf{c}_{1,0}^*, \mathbf{c}_2^*, \{\mathbf{c}_{2,i}^*\}_{i \in [\ell]})$ as an SR-PE encryption of

M_d under $\vec{y}_d, \mathbf{t}^*$, where:

$$\begin{cases} \mathbf{c}^* = \mathbf{c}' \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_1^* = \mathbf{A}^\top \mathbf{s} + \mathbf{e}_1 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{1,i}^* = (\mathbf{A}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{R}_i^\top \mathbf{e}_1 \in \mathbb{Z}_q^m, \quad \forall i \in [\ell] \\ \mathbf{c}_{1,0}^* = \mathbf{C}_t^\top \mathbf{s} + \bar{\mathbf{R}}^\top \mathbf{e}_1 \in \mathbb{Z}_q^m, \\ \mathbf{c}_2^* = \mathbf{c}'_0 \in \mathbb{Z}_q^m, \\ \mathbf{c}_{2,i}^* = \mathbf{c}'_i \in \mathbb{Z}_q^m, \quad \forall i \in [\ell]. \end{cases}$$

Guess: $\mathcal{A}$ may continue to make additional queries to oracles **PrivKG**, **Token** and **UpdKG**. After $\mathcal{A}$ stops queries these oracles, it outputs a bit d' which serves as the guess that the challenge ciphertext $\mathbf{ct}^*$ is an encryption of $M_{d'}$ under $\vec{y}_{d'}$ and $\mathbf{t}^*$. Then setting $c' = d \oplus d'$, algorithm $\mathcal{S}$ returns c' to the PE challenger $\mathcal{B}$, which serves as the guess for $c \in \{0, 1\}$. Recall that c is the bit chosen by $\mathbf{B}$.

On the one hand, it is assumed that for our SR-PE scheme, the adversary $\mathcal{A}$ wins the SR-sA-CPA security experiment with advantage ϵ , which implies that

$$\text{Adv}_{\mathcal{A}}^{\text{SR-sA-CPA}}(n) = \left| \Pr[d' = d \oplus c] - \frac{1}{2} \right| = \epsilon.$$

On the other hand, in the simulation, it holds that: $d' = d \oplus c \Leftrightarrow d' \oplus d = c \Leftrightarrow c' = c$.

We then have

$$\text{Adv}_{\mathcal{S}, \text{PE}}^{\text{wAH-sA-CPA}}(n) = \left| \Pr[c = c'] - \frac{1}{2} \right| = \epsilon.$$

□

Lastly, by construction, the probability that the simulator $\mathcal{S}$ successfully guesses the type of the adversary is $1/2$, which is independent from the adversary $\mathcal{A}$'s behaviour.

By Lemma 4.3.3 and Lemma 4.3.4, we then have:

$$\text{Adv}_{\mathcal{A}}^{\text{SR-sA-CPA}}(n) = \frac{1}{2} \left(\frac{1}{Q} \text{Adv}_{S, \text{PE}}^{\text{wAH-sA-CPA}}(n) + \text{Adv}_{S, \text{PE}}^{\text{wAH-sA-CPA}}(n) \right).$$

By Theorem 2.3.18, we then have that $\text{Adv}_{\mathcal{A}}^{\text{SR-sA-CPA}}(n) = \text{negl}(n)$ under the LWE assumption, which concludes the proof. $\square$

Further Discussions

Recall that the motivation of server-aided revocation mechanism is to reduce users' workloads. However, the KGC's workload is quite high because of the necessity of issuing update keys to the server. To avoid this, the direct revocation model can be considered where there is no need of the key-update phase.

In Chapter 5, we will construct a lattice-based predicate encryption supporting direction revocation.

Chapter 5

Revocable Predicate Encryption from Lattices

5.1 Introduction

Predicate encryption (PE), proposed by Katz, Sahai, and Waters [51] is an important paradigm in the area of public key-encryption. PE system can be regarded as a generalization of attribute-based encryption (ABE) [49, 93], which keeps privacy of not only the plaintext but also the attribute bound to each ciphertext.

Predicate encryption for inner-product predicates was introduced by Katz, Sahai, and Waters [51]. In such a scheme, attribute a and predicate f are expressed as vectors $\vec{x}$ and $\vec{y}$ respectively, and we say $f(a) = 1$ if and only if $\langle \vec{x}, \vec{y} \rangle = 0$ (hereafter, $\langle \vec{x}, \vec{y} \rangle$ denotes the inner product of vector $\vec{x}$ and vector $\vec{y}$). Katz, Sahai, and Waters also demonstrated the expressiveness of inner-product predicates: they can be used to encode several other predicates, such as equalities, hidden vector predicate, polynomial evaluation and CNF/DNF formulae. Following the work of [51], a number of pairing-based PE schemes [12, 56, 81–84] for inner products have been proposed. In the lattice-based world, Agrawal et al. [4] proposed the first such scheme, and Xagawa [102] suggested an improved variant.

The key-update revocation mechanism [15] is one of the most efficient and well-studied models for revocable ABE schemes. However, it admits several limitations, since it requires the key authority to stay online regularly, and the non-revoked users to download updated system information periodically. To eliminate the burden caused by the key update phase, Attrapadung and Imai [10] suggested the direct revocation mechanism for ABE, in which the revocation information can be controlled by the message sender. Each ciphertext is now bound to an attribute a as well as the current revocation list RL . Meanwhile, each private key associated with a predicate f is assigned a unique index I . The decryption procedure is successful if and only if $f(a) = 1$ and $I \notin \text{RL}$. In this direct revocation model, the authority only can stay off-line after issuing private keys for users, and non-revoked users do not have to update their keys. Despite of the clear efficiency advantages for both the key authority and users, this approach requires that senders possess the current revocation list and perform encryptions based on it. The setting that the message sender should possess the revocation information might be inconvenient in certain scenarios, but it is well-suited in cases such information is naturally known to the sender. For instance, in Pay-TV systems [49], the TV program distributor should own the list of revoked users.

In [78, 79], Nieto, Manulis and Sun (NMS) adapted the Attrapadung-Imai direct revocation mechanism into the context of PE, and formalized the notion of revocable predicated encryption (RPE). As discussed in [78, 79], involved privacy challenges may rise when one plugs the revocation problem into the PE setting. In particular, Nieto, Manulis and Sun consider two security notions: attribute-hiding and full-hiding. The former means that the ciphertext only preserves privacy of attribute (and of the encrypted data) as in ordinary PE. The latter is a very strong notion which additionally guarantees that the revocation information is not leaked by the ciphertext. This requirement is suitable for applications where it is necessary for the sender to hide the list of revoked users. Nieto, Manulis and Sun pointed out that a generic construction of full-hiding RPE can be obtained by a combination of a PE scheme and

an anonymous broadcasting scheme, but it is inefficient since the size of the ciphertexts is linearly dependent on the maximal number of users N . Then they proposed a more efficient pairing-based instantiation of full-hiding RPE for inner-product predicates, which relies on the PE schemes by Okamoto and Takashima [82] and Lewko et al. [56], as well as the subset-cover framework [75].

In this work, inspired by the potentials of PE and the advantages of the direct revocation mechanism, we consider full-hiding RPE in the context of lattice-based cryptography, and aim to design the first such scheme from lattice assumptions. Lattice-based cryptography, pioneered by the seminal works by Regev [90] and Gentry et al. [41], has been one of the most exciting research areas in the last decade. Lattices provide several advantages over conventional number-theoretic cryptography, such as conjectured resistance against quantum adversaries and faster arithmetic operations. In the scope of lattice-based revocation schemes, there have been several proposals [28, 29, 77, 100], but they only consider the setting of identity-based encryption (IBE). To the best of our knowledge, the problem of constructing lattice-based RPE schemes has not been addressed so far.

OUR RESULTS AND TECHNIQUES. We introduce the first construction of RPE from lattices. Our scheme satisfies the full-hiding security notion [78, 79] (in a selective manner) in the standard model, based on the hardness of the Learning With Errors (LWE) problem [90]. The scheme inherits the main advantage of the direct revocation mechanism: the authority does not have to be online after the key generation phase, and key updating is not needed. Let N be the maximum expected number of private keys in the system and let r be the number of revoked keys. Then, the efficiency of our scheme is comparable to that of the pairing-based RPE scheme from [78, 79], in the following sense: the size of public parameters is $O(N)$; the size of the private key is $O(\log N)$, and the ciphertext has size $O\left(r \log \frac{N}{r}\right)$ which is ranged between $O(1)$ (when no key is revoked) and $O\left(\frac{N}{2}\right)$ (in the worst case when every second key is revoked).

At a high level, we adopt the approach suggested by Nieto, Manulis and Sun in

their pairing-based instantiation [78, 79], for which we introduce several modifications. Recall that, in [78, 79], to obtain a full-hiding RPE, the authors apply the tree-based revocation technique from [75] to two layers of PE [56, 82], in the following manner: the main PE layer deals with predicate vector $\vec{x}$ and attribute vector $\vec{y}$, while an additional layer is introduced to handle the index I of the private key (encoded as a “predicate”) and the revocation list RL (encoded as an “attribute”). Thanks to the attribute-hiding property of the second PE layer, RL is kept hidden. It is worth noting that Nieto, Manulis and Sun managed to prove the full-hiding security by exploiting the dual system encryption techniques [101] underlying the PE blocks. Their security proof fairly relies on the fact that the simulator is able to compute at least one private key for all predicates, including those for which the challenge attributes satisfy.

To adapt the approach from [78, 79] into the lattice setting, we employ as the main PE layer the scheme for inner-product predicates put forward by Agrawal, Freeman and Vaikuntanathan [4] and subsequently improved by Xagawa [102]. However, we were not able to find a suitable lattice-based ingredient to be used as the second PE layer, so that it interacts smoothly and securely with the main layer (which might due to the fact that there has not been a lattice analogue of the dual system encryption techniques). Instead, we use a variant of Agrawal et al.’s anonymous IBE [2] to realize the second layer as follows. We first consider a binary tree with N leaves, where N is the maximum expected number of private keys. We then associate each node θ in the binary tree with an “identifier” $\mathbf{D}_\theta$. Then, for each $I \in [N]$, we equip the private key indexed by I with “decryption keys” corresponding to all identifiers in the tree path from I to the root. When generating a ciphertext with respect to revocation list RL , the sender aims to the identifiers $\mathbf{D}_{\theta'}$ ’s, for all θ' belonging to the cover set determined by RL . Thanks to the anonymity of the scheme, RL is kept hidden. Furthermore, the correctness of the tree-based revocation technique from [75] ensures that the ciphertext is decryptable using the private key indexed by I if and only if $I \notin \text{RL}$.

To combine the AFV PE layer with the above anonymous IBE layer, we rely on

a splitting technique that can be seen as a secret sharing mechanism and that was used in previous lattice-based revocation schemes [28, 77, 100]. To this end, for each $I \in [N]$, we split a public matrix $\mathbf{U}$ into two random parts: (i) $\mathbf{U}_I$ which is associated with the main PE layer; (ii) $\mathbf{U} - \mathbf{U}_I$ that is linked with the second layer.

The efficiency of our RPE can be improved in the random oracle model, where instead of storing all the matrices $\mathbf{D}_\theta$'s in the public parameters, we simply obtain them as outputs of a random oracle.

Organization. The rest of this paper is organized as follows. We recall definitions and security model of SR-PE in Section 5.2. Our main construction is described and analyzed in Section 5.3

5.2 Definitions

We first recall the definitions of revocable predicate encryption (RPE) from [10, 78, 79], and its full-hiding security notion suggested by Nieto, Manulis and Sun [78, 79].

Definition 5.2.1. A RPE scheme $\mathcal{RPE} = (\text{Setup}, \text{KeyGen}, \text{Enc}, \text{Dec})$, associated with a predicate space $\mathbb{P}$, an attribute space $\mathbb{A}$, an index space $\mathcal{I}$ and a message space $\mathcal{M}$, consists of four algorithms.

Setup (1^n) takes as input a security parameter n . It outputs a state information $\mathbf{st}$, a set of public parameters $\mathbf{pp}$ and a master secret key $\mathbf{msk}$. We assume that $\mathbf{pp}$ is an implicit input of all other algorithms.

KeyGen ($\mathbf{msk}, \mathbf{st}, \vec{x}, I$) takes as input the master secret key $\mathbf{msk}$, the state $\mathbf{st}$, a predicate vector $\vec{x}$ corresponding to a predicate $f_{\vec{x}} \in \mathbb{P}$ and an index $I \in \mathcal{I}$. It outputs an updated state $\mathbf{st}$ and a private key $\mathbf{sk}_{\vec{x}, I}$.

Enc ($\vec{y}, \text{RL}, M$) takes as input an attribute vector $\vec{y} \in \mathbb{A}$, a revocation list $\text{RL} \subseteq \mathcal{I}$, and a message $M \in \mathcal{M}$. It outputs a ciphertext ct .

$\text{Dec}(\text{ct}, \text{sk}_{\vec{x}, I})$ takes as input a ciphertext ct and a private key $\text{sk}_{\vec{x}, I}$. It outputs a message M or the distinguished symbol $\perp$.

Correctness. The correctness requirement demands that, for all pp and msk outputted by $\text{Setup}(1^n)$, all $f_{\vec{x}} \in \mathbb{P}$, $\vec{y} \in \mathbb{A}$, $I \in \mathcal{I}$, all possible valid state information st , all $\text{sk}_{\vec{x}, I} \leftarrow \text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$ and $\text{ct} \leftarrow \text{Enc}(\vec{y}, \text{RL}, M)$, if $I \notin \text{RL}$, then:

1. If $f_{\vec{x}}(\vec{y}) = 1$ then $\text{Dec}(\text{ct}, \text{sk}_{\vec{x}, I}) = M$.
2. If $f_{\vec{x}}(\vec{y}) = 0$ then $\text{Dec}(\text{ct}, \text{sk}_{\vec{x}, I}) = \perp$ with all but negligible probability.

Full-Hiding Security. Nieto, Manulis and Sun [78, 79] introduced the notion of full-hiding security against chosen plaintext attacks for RPE, which guarantees that ciphertexts do not leak any information about the plaintexts, the attributes, nor the revoked indexes. The notion can be defined in the strong, adaptive setting, or in the relaxed, selective manner where the adversary must commit to the challenge attribute vectors $\vec{y}^{(0)}, \vec{y}^{(1)}$ and revocation lists $\text{RL}^{(0)}, \text{RL}^{(1)}$ before seeing public parameters. In this work, we consider the latter.

Definition 5.2.2 (FH-sA-CPA Security). An RPE scheme is selectively full hiding against chosen plaintext attacks (FH-sA-CPA) if any PPT adversary $\mathcal{A}$ has negligible advantage in the following game:

1. $\mathcal{A}$ announces the target attribute vectors $\vec{y}^{(0)}, \vec{y}^{(1)}$, revocation lists $\text{RL}^{(0)}, \text{RL}^{(1)}$.
2. $\text{Setup}(1^n)$ is run to output a state information st , a set of public parameters pp and a master secret key msk . Then $\mathcal{A}$ is given pp .
3. $\mathcal{A}$ may make queries for private keys. For a query of a predicate vector and an index in the form $(\vec{x}, I)$, the adversary $\mathcal{A}$ is given $\text{sk}_{\vec{x}, I} \leftarrow \text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$, subject to one of the following restrictions:

- $f_{\vec{x}}(\vec{y}^{(0)}) = f_{\vec{x}}(\vec{y}^{(1)}) = 0$;

- $f_{\vec{x}}(\vec{y}^{(0)}) = f_{\vec{x}}(\vec{y}^{(1)}) = 1$ and $I \in \text{RL}^{(0)} \cap \text{RL}^{(1)}$;
 - $f_{\vec{x}}(\vec{y}^{(0)}) = 1 \wedge f_{\vec{x}}(\vec{y}^{(1)}) = 0$ and $I \in \text{RL}^{(0)}$;
 - $f_{\vec{x}}(\vec{y}^{(0)}) = 0 \wedge f_{\vec{x}}(\vec{y}^{(1)}) = 1$ and $I \in \text{RL}^{(1)}$.
4. $\mathcal{A}$ outputs two challenge messages $M^{(0)}, M^{(1)}$. A uniformly random bit b is chosen, and $\mathcal{A}$ is given the challenge ciphertext $\text{ct}^* \leftarrow \text{Enc}(\vec{y}^{(b)}, \text{RL}^{(b)}, M^{(b)})$.
 5. $\mathcal{A}$ may continue to make additional queries for private keys, subject to the same restrictions as before.
 6. $\mathcal{A}$ outputs a bit b' and succeeds if $b' = b$. The advantage of $\mathcal{A}$ in the game is defined as:

$$\text{Adv}_{\mathcal{A}, \mathcal{RPE}}^{\text{FH-sA-CPA}}(n) = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

Remark 5.2.3. In the above game, the restrictions for private-key queries are to prevent the adversary to trivially win the game by successfully decrypting the challenge ciphertext ct^* . For the same reason, it is necessary to assume that the two ciphertexts $\text{Enc}(\vec{y}^{(0)}, \text{RL}^{(0)}, M^{(0)})$ and $\text{Enc}(\vec{y}^{(1)}, \text{RL}^{(1)}, M^{(1)})$ have the same size.

5.3 A Lattice-based RPE Scheme

In this section, we present the construction of lattice-based RPE scheme for inner-product predicates.

Our scheme employs two encryption layers: the AFV PE scheme (see Scheme 2.3.17) and a variant of the ABB IBE scheme (see Scheme 2.3.13). Revocation is realized by the CS method (see Section 2.4.1) and a splitting technique that can be seen as a secret sharing mechanism.

5.3.1 Description of the Scheme

We now provide the detailed description of our lattice-based RPE scheme.

Scheme 5.3.1. Our scheme works with (security parameter n and) global parameters $N, \ell, n, q, m, \mathbf{G}, s, B, \chi, \kappa, \text{encode}$, specified below.

- ◇ $N = \text{poly}(n)$: the maximum expected number of users;
- ◇ $\ell = \text{poly}(n)$: the length of predicate and attribute vectors;
- ◇ Prime modulus $q = \tilde{O}(\ell^2 n^4)$, dimension $m = \lceil 2n \log q \rceil$;
- ◇ The primitive matrix $\mathbf{G}$ with public trapdoor $\mathbf{T}_{\mathbf{G}}$ (see Lemma 2.3.2);
- ◇ Gaussian parameter $s = \tilde{O}(\ell\sqrt{m})$; Norm bound $B = \tilde{O}(\sqrt{m})$ and B -bounded distribution χ .
- ◇ The encode function $\text{encode} : \mathcal{M} \rightarrow \{0, 1\}^\kappa$ for $\kappa = \omega(\log n)$, such that we have $\text{encode}(b) = (b, 0, \dots, 0) \in \{0, 1\}^\kappa$ for each $b \in \{0, 1\}$ (see Section 2.3.4).

The attribute space is set as $\mathbb{A} = \mathbb{Z}_q^\ell$. Then each vector $\vec{x} \in \mathbb{A}$ is associated with predicate $f_{\vec{x}} : \mathbb{A} \rightarrow \{0, 1\}$, where for all $\vec{y} \in \mathbb{A}$, we have: $f_{\vec{x}}(\vec{y}) = 1$ if and only if $\langle \vec{x}, \vec{y} \rangle = 0$. The predicate space is defined as $\mathbb{P} = \{f_{\vec{x}} \mid \vec{x} \in \mathbb{A}\}$. The scheme works with index space $\mathcal{I} = [N]$ and message space $\mathcal{M} = \{0, 1\}$.

Setup(1^n): On input security parameter n , this algorithm performs the following steps:

1. Run the algorithm $\text{TrapGen}(n, q, m)$ to generate a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a basis $\mathbf{T}_{\mathbf{A}}$ for $\Lambda_q^\perp(\mathbf{A})$ such that $\|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \leq O(\sqrt{n \log q})$.
2. Choose $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{n \times \kappa}$.
3. Pick $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, for each $i \in [\ell]$.
4. Build a binary tree BT with N leaf nodes. For each node $\theta \in \text{BT}$, pick $\mathbf{D}_\theta \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, which will be viewed as the “identifier” of the node and is stored on BT .
5. Initialize the state $\text{st} = \emptyset$, to record the assigned indexes so far.

6. Output $\text{st}, \text{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \text{BT})$ and $\text{msk} = \mathbf{T}_\mathbf{A}$.

$\text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$: On input the master key msk , state st , a predicate vector $\vec{x} = (x_1, \dots, x_\ell) \in \mathbb{Z}_q^\ell$ and an index $I \in [N]$, this algorithm performs the following steps:

1. If $I \in \text{st}$, then return $\perp$. Else, add I to st .
2. Choose $\mathbf{U}_I \xleftarrow{\$} \mathbb{Z}_q^{n \times \kappa}$.
3. Set $\mathbf{A}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{A}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$ and sample $\mathbf{Z} \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{A}_{\vec{x}}, \mathbf{T}_\mathbf{A}, \mathbf{U}_I, s)$.
We remark that $\mathbf{Z}$ is a matrix in $\mathbb{Z}^{2m \times \kappa}$ satisfying $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} = \mathbf{U}_I$.
4. For each $\theta \in \text{Path}(I)$, sample $\mathbf{Z}_\theta \leftarrow \text{SampleLeft}(\mathbf{A}, \mathbf{D}_\theta, \mathbf{T}_\mathbf{A}, \mathbf{U} - \mathbf{U}_I, s)$. We note that each $\mathbf{Z}_\theta$ is a matrix in $\mathbb{Z}^{2m \times \kappa}$ satisfying $[\mathbf{A} \mid \mathbf{D}_\theta] \cdot \mathbf{Z}_\theta = \mathbf{U} - \mathbf{U}_I$.
5. Output the updated state st and $\text{sk}_{\vec{x}, I} = (I, \mathbf{Z}, \{\mathbf{Z}_\theta\}_{\theta \in \text{Path}(I)})$.

$\text{Enc}(\vec{y}, \text{RL}, M)$: On input an attribute vector $\vec{y} = (y_1, \dots, y_\ell) \in \mathbb{Z}_q^\ell$, a revocation list $\text{RL} \subseteq [N]$ and a message $M \in \{0, 1\}$, this algorithm performs the following steps:

1. Choose $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}' \leftarrow \chi^\kappa$ and $\mathbf{e} \leftarrow \chi^m$.
2. Pick $\mathbf{R}_i, \mathbf{S}_\theta \xleftarrow{\$} \{-1, 1\}^{m \times m}$, for each $i \in [\ell]$ and each $\theta \in \text{KUNodes}(\text{BT}, \text{RL})$.
3. Output $\text{ct} = (\mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\hat{\mathbf{c}}_\theta\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})})$, where:

$$\begin{cases} \mathbf{c}' = \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_0 = \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m, \\ \mathbf{c}_i = (\mathbf{A}_i + y_i \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{R}_i^\top \mathbf{e} \in \mathbb{Z}_q^m, \quad \forall i \in [\ell], \\ \hat{\mathbf{c}}_\theta = \mathbf{D}_\theta^\top \mathbf{s} + \mathbf{S}_\theta^\top \mathbf{e} \in \mathbb{Z}_q^m, \quad \forall \theta \in \text{KUNodes}(\text{BT}, \text{RL}). \end{cases}$$

$\text{Dec}(\text{ct}, \text{sk}_{\vec{x}, I})$: On input a ciphertext $\text{ct} = (\mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\hat{\mathbf{c}}_{\theta'}\}_{\theta'})$, where $\{\hat{\mathbf{c}}_{\theta'}\}_{\theta'}$ is a collection of vectors in $\mathbb{Z}_q^m$, and a private key $\text{sk}_{\vec{x}, I} = (I, \mathbf{Z}, \{\mathbf{Z}_\theta\}_{\theta \in \text{Path}(I)})$, this algorithm proceeds as follows:

1. Set $\mathbf{c}_{\vec{x}} = \sum_{i=1}^{\ell} (\mathbf{G}^{-1}(x_i \cdot \mathbf{G}))^{\top} \mathbf{c}_i \in \mathbb{Z}_q^m$.
2. For all pairs (θ, θ') , compute $\mathbf{d}_{\theta, \theta'} = \mathbf{c}' - \mathbf{Z}^{\top}(\mathbf{c}_0 \| \mathbf{c}_{\vec{x}}) - \mathbf{Z}_{\theta}^{\top}(\mathbf{c}_0 \| \hat{\mathbf{c}}_{\theta'}) \in \mathbb{Z}_q^{\kappa}$.
3. If there exists a pair (θ, θ') such that $\lfloor \frac{2}{q} \cdot \mathbf{d}_{\theta, \theta'} \rfloor = \text{encode}(M')$, for some $M' \in \{0, 1\}$, then output M' . Otherwise, output $\perp$.

5.3.2 Correctness, Efficiency and Potential Implementation

Correctness. We will show that the scheme satisfies the correctness requirement with all but negligible probability. We proceed as in [4, 38, 102].

Suppose that $\mathbf{ct} = (\mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\hat{\mathbf{c}}_{\theta'}\}_{\theta' \in \text{KUNodes}(\text{BT}, \text{RL})})$ is a correctly computed ciphertext of message $M \in \{0, 1\}$, with respect to some $\vec{y} \in \mathbb{Z}_q^{\ell}$ and some $\text{RL} \subseteq [N]$. Let $\mathbf{sk}_{\vec{x}, I} = (I, \mathbf{Z}, \{\mathbf{Z}_{\theta}\}_{\theta \in \text{Path}(I)})$ be an honestly generated private key, where $I \notin \text{RL}$. We first observe that the following holds:

$$\mathbf{c}_{\vec{x}} = \sum_{i=1}^{\ell} (\mathbf{G}^{-1}(x_i \cdot \mathbf{G}))^{\top} \mathbf{c}_i = (\mathbf{A}_{\vec{x}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G})^{\top} \mathbf{s} + (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e} \quad (5.1)$$

where $\mathbf{R}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{R}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$. By construction, since $I \notin \text{RL}$, there exists a pair (θ, θ') corresponding to the same node in BT satisfying

$$[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} + [\mathbf{A} \mid \mathbf{D}_{\theta'}] \cdot \mathbf{Z}_{\theta} = \mathbf{U}.$$

Now we consider two cases:

1. Case 1: Suppose that $\langle \vec{x}, \vec{y} \rangle = 0$. We then have:

$$\mathbf{c}_{\vec{x}} = (\mathbf{A}_{\vec{x}})^{\top} \mathbf{s} + (\mathbf{R}_{\vec{x}})^{\top} \mathbf{e}.$$

Then for the pair (θ, θ') specified above, the following holds:

$$\begin{aligned}
\mathbf{d}_{\theta, \theta'} &= \mathbf{c}' - \mathbf{Z}^\top (\mathbf{c}_0 \| \mathbf{c}_{\vec{x}}) - \mathbf{Z}_\theta^\top (\mathbf{c}_0 \| \hat{\mathbf{c}}_{\theta'}) \\
&= \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) - \mathbf{Z}^\top \left([\mathbf{A} \mid \mathbf{A}_{\vec{x}}]^\top \mathbf{s} + (\mathbf{e} \| (\mathbf{R}_{\vec{x}})^\top \mathbf{e}) \right) \\
&\quad - \mathbf{Z}_\theta^\top \left([\mathbf{A} \mid \mathbf{D}_{\theta'}]^\top \mathbf{s} + (\mathbf{e} \| \mathbf{S}_{\theta'}^\top \mathbf{e}) \right) \\
&= \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) + \underbrace{\mathbf{e}' - \mathbf{Z}^\top (\mathbf{e} \| (\mathbf{R}_{\vec{x}})^\top \mathbf{e}) - \mathbf{Z}_\theta^\top (\mathbf{e} \| \mathbf{S}_{\theta'}^\top \mathbf{e})}_{\text{error}}.
\end{aligned}$$

As in [2, 4, 38, 102], the above error term can be showed to be bounded by $s\ell m^2 B \cdot \omega(\log n) = \tilde{O}(\ell^2 n^3)$, with all but negligible probability. In order for the decryption algorithm to correctly recover $\text{encode}(M)$, and subsequently the message M , it is required that the error term is bounded by $q/5$, i.e., $\|\text{error}\|_\infty < q/5$. This is guaranteed by our setting of modulus q , i.e., $q = \tilde{O}(\ell^2 n^4)$.

2. Case 2: Suppose that $\langle \vec{x}, \vec{y} \rangle \neq 0$. We then have:

$$\mathbf{c}_{\vec{x}} = \left(\mathbf{A}_{\vec{x}} + \underbrace{\langle \vec{x}, \vec{y} \rangle}_{\neq 0} \cdot \mathbf{G} \right)^\top \mathbf{s} + (\mathbf{R}_{\vec{x}})^\top \mathbf{e}. \quad (5.2)$$

Then for each pair (θ, θ') , the following holds:

$$\begin{aligned}
\mathbf{d}_{\theta, \theta'} &= \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) \\
&\quad - \mathbf{Z}^\top \left([\mathbf{A} \mid \mathbf{A}_{\vec{x}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}]^\top \mathbf{s} + (\mathbf{e} \| (\mathbf{R}_{\vec{x}})^\top \mathbf{e}) \right) \\
&\quad - \mathbf{Z}_\theta^\top \left([\mathbf{A} \mid \mathbf{D}_{\theta'}]^\top \mathbf{s} + (\mathbf{e} \| \mathbf{S}_{\theta'}^\top \mathbf{e}) \right) \\
&= \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) + \left(\mathbf{U} - [\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} - [\mathbf{A} \mid \mathbf{D}_{\theta'}] \cdot \mathbf{Z}_\theta \right)^\top \mathbf{s} \\
&\quad - \mathbf{Z}^\top [\mathbf{0} \mid \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}]^\top \mathbf{s} + \text{error}
\end{aligned}$$

Notice that the term $\mathbf{Z}^\top [\mathbf{0} \mid \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}]^\top \mathbf{s}$ can be written as $\langle \vec{x}, \vec{y} \rangle \cdot (\mathbf{G}\mathbf{Z}_2)^\top \mathbf{s} \in$

$\mathbb{Z}_q^\kappa$, where $\mathbf{Z}_2 \in \mathbb{Z}^{m \times \kappa}$ is the bottom part of matrix $\mathbf{Z}$. By Lemma 2.3.4, we have that the distribution of $\mathbf{G}\mathbf{Z}_2$ is statistically close to uniform over $\mathbb{Z}_q^{n \times \kappa}$. It follows that, for each pair (θ, θ') , vector $\mathbf{d}_{\theta, \theta'} \in \mathbb{Z}_q^\kappa$ is indistinguishable from uniform. As a result, the probability that the last $\kappa - 1$ coordinates of vector $\lfloor \frac{2}{q} \cdot \mathbf{d}_{\theta, \theta'} \rfloor$ are all 0 is at most $2^{-(\kappa-1)} = 2^{-\omega(\log n)}$, which is negligible in n . In other words, with all but negligible probability, the decryption algorithm outputs $\perp$ since it does not obtain a proper encoding $\text{encode}(M) \in \{0, 1\}^\kappa$, for $M \in \{0, 1\}$.

Efficiency. The efficiency aspect of our RPE scheme is as follows:

- ◇ The public parameters $\mathbf{pp}$ has bit-size $((\ell + 2N)nm + n\kappa) \log q = \left(\tilde{O}(\ell) + O(N) \right) \cdot \tilde{O}(n^2)$.
- ◇ The bit-size of private key $\mathbf{sk}_{\vec{x}, I}$ is bit-size $O(\log N) \cdot \tilde{O}(n)$.
- ◇ The ciphertext $\mathbf{ct}$ has bit-size $\left(\tilde{O}(\ell) + O(r \log \frac{N}{r}) \right) \cdot \tilde{O}(n)$.

The efficiency of our scheme then is comparable to that of the pairing-based RPE instantiation [78] in the following sense: the set of public parameters has size $O(N)$; the private key has size $O(\log N)$, and the size of the ciphertext is $O\left(r \log \frac{N}{r}\right)$ which is ranged between $O(1)$ (when no key is revoked) and $O\left(\frac{N}{2}\right)$ (in the worst case when every second key is revoked).

In Section 5.3.4, we will discuss a variant of our RPE scheme in the random oracle model, with smaller size of public parameters.

Potential Implementation. While the focus of this work is to provide the first provably secure construction of RPE from lattices, it would be desirable to back it up with practical implementations and to compare the implementation details with those of pairing-based counterparts. However, this would be a highly challenging task, due to two main reasons:

1. We are not aware of any concrete implementation of the two building blocks of our scheme, i.e., the AFV PE scheme [4, 102] and the ABB IBE scheme [2].

2. In [78, 79], Nieto, Manulis and Sun did not provide implementation details of their pairing-based instantiation of RPE.

Nevertheless, in the following, we will discuss the potential of such implementation, by analyzing the main cryptographic operations needed to implement the scheme. Apart from simple operations such as samplings of uniformly random matrices and vectors whose entries are in $\mathbb{Z}_q$ or $\{-1, 1\}$, as well as multiplication and addition operations over $\mathbb{Z}_q$, the algorithms in the scheme require the following time-consuming tasks:

- ◇ Generation of a lattice trapdoor;
- ◇ Samplings of discrete Gaussian vectors over lattices;
- ◇ Samplings of **LWE** error vectors.

We remark that it is feasible to implement the listed above cryptographic tasks using the algorithms provided in [41, 72], which were recently improved in [39, 74]. Some implementation results of the above cryptographic tasks were recently reported in [50], which may serve as a stepping stone of the potential implementation of our RPE scheme.

Given these circumstances and potentials of implementation, we leave the detailed implementation aspect of our scheme as a future investigation.

5.3.3 Security

In the following theorem, we state that our RPE scheme is selectively full hiding in the standard model, under the **LWE** assumption.

Theorem 5.3.2. For the given parameters, the RPE system in Scheme 5.3.1 satisfies the security notion of **FH-sA-CPA** defined in Definition 5.2.2, assuming hardness of the (n, q, χ) -**LWE** problem.

Proof Outline. We first highlight the high-level idea of the security proof and then provide the formal proof later.

We will demonstrate the proof via 4 games, which can be prove are either statistically indistinguishable or computationally indistinguishable based on the **LWE** assumption. Specially, Game 1 is exactly the one from the security definition. In Game 2 and Game 3, we change the way to answer queries to private keys and to generate the challenge ciphertext, respectively. Finally, in Game 4, the challenge ciphertext is replaced with uniformly random strings. Therefore, the advantage of the adversary in Game 4 is zero. Hence, the advantage of the adversary in Game 1 is negligible by the indistinguishability of these four games.

Proof. We proceed through a series of games, similar to those in [4, 38, 44]. Before describing the games, we first define the auxiliary algorithms for generating simulated public parameters, private keys and ciphertexts.

Auxiliary algorithms. For the same parameters defined in Scheme 5.3.1, we consider the following auxiliary algorithms.

Sim.Setup $(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^*, \text{RL}^*)$: On input a security parameter n , a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, a matrix $\mathbf{U} \in \mathbb{Z}_q^{n \times \kappa}$, the challenge attribute vector $\vec{y}^* = (y_1^*, \dots, y_\ell^*) \in \mathbb{Z}_q^\ell$ and revocation list $\text{RL}^* \subseteq [N]$, this algorithm performs the following steps:

1. For each $i \in [\ell]$, pick $\mathbf{R}_i \xleftarrow{\$} \{-1, 1\}^{m \times m}$ and set $\mathbf{A}_i = \mathbf{A}\mathbf{R}_i - y_i^* \cdot \mathbf{G}$.
2. Build a binary tree **BT** and pick $\mathbf{S}_\theta \xleftarrow{\$} \{-1, 1\}^{m \times m}$ for each $\theta \in \text{BT}$. Set the identifier:

$$\mathbf{D}_\theta = \begin{cases} \mathbf{A}\mathbf{S}_\theta, & \text{if } \theta \in \text{KUNodes}(\text{BT}, \text{RL}^*), \\ \mathbf{A}\mathbf{S}_\theta + \mathbf{G}, & \text{otherwise.} \end{cases}$$

3. Initialize the state $\text{st} := \emptyset$.
4. Output $\text{st}, \text{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \text{BT})$ and $\text{msk}^* = (\{\mathbf{R}_i\}_{i \in [\ell]}, \{\mathbf{S}_\theta\}_{\theta \in \text{BT}})$.

$\text{Sim.KeyGen}(\text{msk}^*, \text{st}, \vec{x}, I, \vec{y}^*, \text{RL}^*)$: This algorithm takes as input msk^* , state st , a predicate vector $\vec{x} \in \mathbb{Z}_q^\ell$, an index $I \in [N]$, the challenge attribute vector $\vec{y}^* \in \mathbb{Z}_q^\ell$ and revocation list $\text{RL}^* \subseteq [N]$, such that the following condition holds: if $\langle \vec{x}, \vec{y}^* \rangle = 0$ then $I \in \text{RL}^*$ (see Definition 5.2.2).

The algorithm returns $\perp$ if $I \in \text{st}$. Otherwise, it outputs the updated state $\text{st} \leftarrow \text{st} \cup \{I\}$ and a private key $\text{sk}_{\vec{x}, I} = (I, \mathbf{Z}, \{\mathbf{Z}_\theta\}_{\theta \in \text{Path}(I)})$ computed based on the value $\langle \vec{x}, \vec{y}^* \rangle$ as follows.

1. Case 1: $\langle \vec{x}, \vec{y}^* \rangle \neq 0$.

(a) If $I \notin \text{RL}^*$, then there is exactly one node θ^* in the intersection $\text{Path}(I) \cap \text{KUNodes}(\text{BT}, \text{RL}^*)$.

Using Lemma 2.2.6, sample $\mathbf{Z}_{\theta^*} \leftarrow (\mathcal{D}_{\mathbb{Z}^{2m}, s})^\kappa$ and set $\mathbf{U}_I = \mathbf{U} - [\mathbf{A} \mid \mathbf{D}_{\theta^*}] \cdot \mathbf{Z}_{\theta^*}$. For each node $\theta \in \text{Path}(I) \setminus \{\theta^*\}$, sample

$$\mathbf{Z}_\theta \leftarrow \text{SampleRight}(\mathbf{A}, \mathbf{S}_\theta, 1, \mathbf{G}, \mathbf{T}_\mathbf{G}, \mathbf{U} - \mathbf{U}_I, s).$$

(b) If $I \in \text{RL}^*$, choose $\mathbf{U}_I \xleftarrow{\$} \mathbb{Z}_q^{n \times \kappa}$. Then for each $\theta \in \text{Path}(I)$, sample

$$\mathbf{Z}_\theta \leftarrow \text{SampleRight}(\mathbf{A}, \mathbf{S}_\theta, 1, \mathbf{G}, \mathbf{T}_\mathbf{G}, \mathbf{U} - \mathbf{U}_I, s).$$

After determining $\mathbf{U}_I$, in this case, as we have:

$$\mathbf{A}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{A}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G}) = \mathbf{A} \mathbf{R}_{\vec{x}} - \underbrace{\langle \vec{x}, \vec{y}^* \rangle \cdot \mathbf{G}}_{\neq 0}$$

where $\mathbf{R}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{R}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G})$, we can sample matrix

$$\mathbf{Z} \leftarrow \text{SampleRight}(\mathbf{A}, \mathbf{R}_{\vec{x}}, -\langle \vec{x}, \vec{y}^* \rangle, \mathbf{G}, \mathbf{T}_\mathbf{G}, \mathbf{U}_I, s)$$

satisfying $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} = \mathbf{U}_I$.

2. Case 2: $\langle \vec{x}, \vec{y}^* \rangle = 0$.

In this case, we always have that $I \in \text{RL}^*$, which implies that $\text{Path}(I) \cap \text{KUNodes}(\text{BT}, \text{RL}^*) = \emptyset$. Note that, here we do not have a trapdoor for the matrix $[\mathbf{A} \mid \mathbf{A}_{\vec{x}}]$, since in this case:

$$\mathbf{A}_{\vec{x}} = \sum_{i=1}^{\ell} \mathbf{A}_i \mathbf{G}^{-1}(x_i \cdot \mathbf{G}) = \mathbf{A} \mathbf{R}_{\vec{x}}.$$

However, we can instead compute $\mathbf{Z}$ and $\{\mathbf{Z}_{\theta}\}_{\theta \in \text{Path}(I)}$ as follows. First, we sample $\mathbf{Z} \leftarrow (\mathcal{D}_{\mathbb{Z}^{2m}, s})^{\kappa}$ and set $\mathbf{U}_I = [\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z}$. Then, for each $\theta \in \text{Path}(I)$, we sample

$$\mathbf{Z}_{\theta} \leftarrow \text{SampleRight}(\mathbf{A}, \mathbf{S}_{\theta}, 1, \mathbf{G}, \mathbf{T}_{\mathbf{G}}, \mathbf{U} - \mathbf{U}_I, s).$$

Sim.Enc($\text{msk}^*, M, \mathbf{d}_0, \mathbf{d}'$): On input msk^* , a message $M \in \{0, 1\}$, and the extra inputs $\mathbf{d}_0 \in \mathbb{Z}_q^m$, $\mathbf{d}' \in \mathbb{Z}_q^{\kappa}$, it outputs $\text{ct} = (\mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\mathbf{c}_{\theta}\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})})$, where:

$$\begin{cases} \mathbf{c}' = \mathbf{d}' + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M) \in \mathbb{Z}_q^{\kappa}, \\ \mathbf{c}_0 = \mathbf{d}_0 \in \mathbb{Z}_q^m, \\ \mathbf{c}_i = \mathbf{R}_i^{\top} \mathbf{d}_0 \in \mathbb{Z}_q^m, \quad \forall i \in [\ell], \\ \hat{\mathbf{c}}_{\theta} = \mathbf{S}_{\theta}^{\top} \mathbf{d}_0 \in \mathbb{Z}_q^m, \quad \forall \theta \in \text{KUNodes}(\text{BT}, \text{RL}^*). \end{cases}$$

The series of games. Let $\mathcal{A}$ be the adversary in the selective full-hiding game of Definition 5.2.2. We define the following series of games.

- $\text{Game}_0^{(b)}$: This game is the real security game in Definition 5.2.2, where the chosen bit is $b \in \{0, 1\}$.
- $\text{Game}_1^{(b)}$: This game is the same as $\text{Game}_0^{(b)}$, except that algorithms $\text{Setup}(1^n)$,

$\text{Enc}(\vec{y}^{(b)}, \text{RL}^{(b)}, M^{(b)})$ are replaced by

$$\text{Sim.Setup}(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^{(b)}, \text{RL}^{(b)}), \quad \text{Sim.Enc}(\text{msk}^*, M^{(b)}, \mathbf{A}^\top \mathbf{s} + \mathbf{e}, \mathbf{U}^\top \mathbf{s} + \mathbf{e}'),$$

respectively, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$, $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow \chi^m$, and $\mathbf{e}' \leftarrow \chi^\kappa$.

- $\text{Game}_2^{(b)}$: This game is the same as $\text{Game}_1^{(b)}$, except that $\text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$ is replaced by $\text{Sim.KeyGen}(\text{msk}^*, \text{st}, \vec{x}, I, \vec{y}^{(b)}, \text{RL}^{(b)})$.
- $\text{Game}_3^{(b)}$: This game is the same as $\text{Game}_2^{(b)}$, except that the inputs of algorithm $\text{Sim.Enc}(\text{msk}^*, M^{(b)}, \mathbf{d}_0, \mathbf{d}')$ are changed to $\mathbf{d}_0 \xleftarrow{\$} \mathbb{Z}_q^m$ and $\mathbf{d}' \xleftarrow{\$} \mathbb{Z}_q^\kappa$.
- Game_4 : In this final game, we make the following modifications:
 - $\text{Sim.Setup}(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^{(b)}, \text{RL}^{(b)})$ is replaced by $\text{Setup}(1^n)$.
 - $\text{Sim.KeyGen}(\text{msk}^*, \text{st}, \vec{x}, I, \vec{y}^{(b)}, \text{RL}^{(b)})$ is replaced by $\text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$.
 - Instead of computing $\mathbf{c}' = \mathbf{d}' + \lfloor \frac{q}{2} \rfloor \cdot \text{encode}(M^{(b)}) \in \mathbb{Z}_q^\kappa$, we sample $\mathbf{c}' \xleftarrow{\$} \mathbb{Z}_q^\kappa$.

To prove Theorem 5.3.2, we will demonstrate in the following 4 lemmas that any two consecutive games in the above series are either statistically indistinguishable or computationally indistinguishable under the LWE assumption.

Lemma 5.3.3. The adversary $\mathcal{A}$'s view in $\text{Game}_0^{(b)}$ is statistically close to the view in $\text{Game}_1^{(b)}$.

Proof. We need to show that the public parameters $\text{pp} = (\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \text{BT})$ and the ciphertext $\text{ct} = (\mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\widehat{\mathbf{c}}_\theta\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})})$ produced by Setup and Enc in $\text{Game}_0^{(b)}$ are statistically close to those produced by $\text{Sim.Setup}(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^{(b)}, \text{RL}^{(b)})$ and $\text{Sim.Enc}(\text{msk}^*, M^{(b)}, \mathbf{A}^\top \mathbf{s} + \mathbf{e}, \mathbf{U}^\top \mathbf{s} + \mathbf{e}')$ in $\text{Game}_1^{(b)}$, respectively.

We first note that matrix $\mathbf{A}$ is truly uniform in $\text{Game}_1^{(b)}$. In $\text{Game}_0^{(b)}$, it is generated via algorithm TrapGen , and hence, is statistically close to uniform over $\mathbb{Z}_q^{n \times m}$ by

Lemma 2.3.1. Furthermore, we observe that matrix $\mathbf{U} \in \mathbb{Z}_q^{n \times \kappa}$ is truly uniform in both games.

Let $\vec{y}^{(b)} = (y_1^{(b)}, \dots, y_\ell^{(b)})$. For each $i \in [\ell]$ and each $\theta \in \text{BT}$, matrices $\mathbf{A}_i, \mathbf{D}_\theta \in \mathbb{Z}_q^{n \times m}$ are truly uniform in $\text{Game}_0^{(b)}$, while in $\text{Game}_1^{(b)}$, they are generated as:

$$\mathbf{A}_i = \mathbf{A}\mathbf{R}_i - y_i^{(b)} \cdot \mathbf{G}; \quad \mathbf{D}_\theta = \mathbf{A}\mathbf{S}_\theta + \rho_\theta \cdot \mathbf{G},$$

where $\mathbf{R}_i, \mathbf{S}_\theta \xleftarrow{\$} \{-1, 1\}^{m \times m}$ and $\rho_\theta \in \{0, 1\}$. Then, the ciphertext components $\mathbf{c}'$, $\mathbf{c}_0$, $\{\mathbf{c}_i\}_{i \in [\ell]}$ and $\{\widehat{\mathbf{c}}_\theta\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})}$ in both games can be expressed as:

$$\begin{cases} \mathbf{c}' = \mathbf{U}^\top \mathbf{s} + \mathbf{e}' + \lfloor \frac{q}{2} \rfloor \cdot \text{encode}(M^{(b)}) \in \mathbb{Z}_q^\kappa, \\ \mathbf{c}_0 = \mathbf{A}^\top \mathbf{s} + \mathbf{e} \in \mathbb{Z}_q^m, \\ \mathbf{c}_i = (\mathbf{A}_i + y_i^{(b)} \cdot \mathbf{G})^\top \mathbf{s} + \mathbf{R}_i^\top \mathbf{e} = \mathbf{R}_i^\top (\mathbf{A}^\top \mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^m, \quad \forall i \in [\ell], \\ \widehat{\mathbf{c}}_\theta = \mathbf{D}_\theta^\top \mathbf{s} + \mathbf{S}_\theta^\top \mathbf{e} = \mathbf{S}_\theta^\top (\mathbf{A}^\top \mathbf{s} + \mathbf{e}) \in \mathbb{Z}_q^m, \quad \forall \theta \in \text{KUNodes}(\text{BT}, \text{RL}^{(b)}), \end{cases}$$

where $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$, $\mathbf{e}' \xleftarrow{\$} \chi^\kappa$ and $\mathbf{e} \xleftarrow{\$} \chi^m$. By Lemma 2.3.3, the joint distributions of:

$$\left(\mathbf{A}, \mathbf{A}\mathbf{R}_i - \vec{y}_i^{(b)} \cdot \mathbf{G}, \mathbf{R}_i^\top \mathbf{e} \right) \quad \text{and} \quad \left(\mathbf{A}, \mathbf{A}_i, \mathbf{R}_i^\top \mathbf{e} \right),$$

as well as

$$\left(\mathbf{A}, \mathbf{A}\mathbf{S}_\theta + \rho_\theta \cdot \mathbf{G}, \mathbf{S}_\theta^\top \mathbf{e} \right) \quad \text{and} \quad \left(\mathbf{A}, \mathbf{D}_\theta, \mathbf{S}_\theta^\top \mathbf{e} \right)$$

as statistically indistinguishable.

The above discussions imply that the distributions of

$$\left(\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \{\mathbf{D}_\theta\}_{\theta \in \text{BT}}, \mathbf{c}', \mathbf{c}_0, \{\mathbf{c}_i\}_{i \in [\ell]}, \{\widehat{\mathbf{c}}_\theta\}_{\theta \in \text{KUNodes}(\text{BT}, \text{RL})} \right)$$

in $\text{Game}_0^{(b)}$ and $\text{Game}_1^{(b)}$ are statistically indistinguishable. This concludes the lemma. $\square$

Lemma 5.3.4. The adversary $\mathcal{A}$'s view in $\text{Game}_1^{(b)}$ is statistically close to the view in $\text{Game}_2^{(b)}$.

Proof. Note that, from $\text{Game}_1^{(b)}$ to $\text{Game}_2^{(b)}$, we replace the real key generation algorithm **KeyGen** by algorithm **Sim.KeyGen**. Thus, we need to demonstrate that for all queries of the form $(\vec{x}, I)$ from $\mathcal{A}$, the private keys $\text{sk}_{\vec{x}, I} = (I, \mathbf{Z}, \{\mathbf{Z}_\theta\}_{\theta \in \text{Path}(I)})$ outputted by **KeyGen** and **Sim.KeyGen** are statistically indistinguishable.

First of all, we note that, in both cases, matrices $\mathbf{Z} \in \mathbb{Z}^{2m \times \kappa}, \{\mathbf{Z}_\theta \in \mathbb{Z}^{2m \times \kappa}\}_{\theta \in \text{Path}(I)}$ satisfy the condition:

$$[\mathbf{A} \mid \mathbf{A}_{\vec{x}}] \cdot \mathbf{Z} + [\mathbf{A} \mid \mathbf{D}_\theta] \cdot \mathbf{Z}_\theta = \mathbf{U}, \quad \forall \theta \in \text{Path}(I).$$

Next, we note that, in **KeyGen**, the columns of these matrices are sampled via algorithm **SampleLeft**, while in **Sim.KeyGen**, they are either sampled by algorithm **SampleRight** or sampled from $\mathcal{D}_{\mathbb{Z}^m, s}$. The properties of these sampling algorithms (see Section 2.3.1) then ensure that the two distributions are statistically indistinguishable. $\square$

Lemma 5.3.5. Under the (n, q, χ) -LWE assumption, the adversary $\mathcal{A}$'s view in $\text{Game}_2^{(b)}$ is computationally indistinguishable from the view in $\text{Game}_3^{(b)}$.

Proof. Recall that, from $\text{Game}_2^{(b)}$ to $\text{Game}_3^{(b)}$, we change the inputs $\mathbf{d}_0, \mathbf{d}'$ to algorithm **Sim.Enc** from LWE instances to uniformly random vectors in $\mathbb{Z}_q^m$ and $\mathbb{Z}_q^\kappa$, respectively. Suppose that the adversary $\mathcal{A}$ has non-negligible advantage in distinguishing $\text{Game}_3^{(b)}$ from $\text{Game}_2^{(b)}$. We use $\mathcal{A}$ to construct an LWE solver $\mathcal{B}$ that proceeds as follows:

- $\mathcal{B}$ requests for $m + \kappa$ instances $\{(\mathbf{a}_j, v_j) \in \mathbb{Z}_q^n \times \mathbb{Z}_q\}_{j \in [m+\kappa]}$ from the LWE challenger.
- $\mathcal{B}$ computes the following matrices and vectors

$$\begin{aligned} \mathbf{A} &:= [\mathbf{a}_1, \dots, \mathbf{a}_m] \in \mathbb{Z}_q^{n \times m}, & \mathbf{U} &:= [\mathbf{a}_{m+1}, \dots, \mathbf{a}_{m+\kappa}] \in \mathbb{Z}_q^{n \times \kappa}, \\ \mathbf{d}_0 &:= [v_1, \dots, v_m]^\top \in \mathbb{Z}_q^m, & \mathbf{d}' &:= [v_{m+1}, \dots, v_{m+\kappa}]^\top \in \mathbb{Z}_q^\kappa, \end{aligned}$$

and runs $\text{Sim.Setup}(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^{(b)}, \text{RL}^{(b)})$ as in $\text{Game}_2^{(b)}$.

- $\mathcal{B}$ answers the private key queries by running $\text{Sim.KeyGen}(\text{msk}^*, \text{st}, \vec{x}, I, \vec{y}^{(b)}, \text{RL}^{(b)})$ as in $\text{Game}_2^{(b)}$.
- After receiving from $\mathcal{A}$ two messages $M^{(0)}, M^{(1)} \in \{0, 1\}$, algorithm $\mathcal{B}$ prepares a challenge ciphertext ct^* by running $\text{Sim.Enc}(\text{msk}^*, M^{(b)}, \mathbf{d}_0, \mathbf{d}')$.
- Finally, after being allowed to make additional queries, $\mathcal{A}$ guesses whether it is interacting with $\text{Game}_3^{(b)}$ or $\text{Game}_2^{(b)}$. Then, $\mathcal{B}$ outputs $\mathcal{A}$'s guess as the answer to the LWE challenger.

Recall that by Definition 2.2.8, for each $j \in [m + \kappa]$, either $v_j = \langle \mathbf{a}_j, \mathbf{s} \rangle + e_j$ for secret $\mathbf{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and noise $e_j \leftarrow \chi$; or v_j is uniformly random in $\mathbb{Z}_q$. On the one hand, if v_j is uniformly random in $\mathbb{Z}_q$, then $\mathcal{A}$'s view is as in $\text{Game}_3^{(b)}$. On the other hand, if $v_j = \langle \mathbf{a}_j, \mathbf{s} \rangle + e_j$, then the adversary $\mathcal{A}$'s view is as in $\text{Game}_2^{(b)}$. Therefore, algorithm $\mathcal{B}$ can solve the (n, q, χ) -LWE problem with non-negligible probability, assuming that the adversary $\mathcal{A}$ can distinguish $\text{Game}_3^{(b)}$ from $\text{Game}_2^{(b)}$ with non-negligible advantage. This concludes the lemma. $\square$

Lemma 5.3.6. The adversary $\mathcal{A}$'s view in $\text{Game}_3^{(b)}$ is statistically close to the view in Game_4 .

Proof. First of all, based on the same argument as in Lemma 5.3.3, we can deduce that the public parameters pp outputted by algorithm $\text{Sim.Setup}(1^n, \mathbf{A}, \mathbf{U}, \vec{y}^{(b)}, \text{RL}^{(b)})$ in $\text{Game}_3^{(b)}$ is statistically close that generated by algorithm $\text{Setup}(1^n)$ in Game_4 .

Secondly, based on the same argument as in Lemma 5.3.4, we can deduce that the output of algorithm $\text{Sim.KeyGen}(\text{msk}^*, \text{st}, \vec{x}, I, \vec{y}^{(b)}, \text{RL}^{(b)})$ in $\text{Game}_3^{(b)}$ is statistically close to that of algorithm $\text{KeyGen}(\text{msk}, \text{st}, \vec{x}, I)$ in Game_4 .

Thirdly, the shift from $\mathbf{c}' = \mathbf{d}' + \left\lfloor \frac{q}{2} \right\rfloor \cdot \text{encode}(M^{(b)}) \in \mathbb{Z}_q^\kappa$ to a uniformly random $\mathbf{c}' \in \mathbb{Z}_q^\kappa$ is only a conceptual change, because vector $\mathbf{d}'$ in $\text{Game}_3^{(b)}$ is a uniformly random element in $\mathbb{Z}_q^\kappa$. $\square$

Finally, Theorem 5.3.2 follows from the fact that the advantage of $\mathcal{A}$ in Game_4 is zero, since Game_4 no longer depends on the bit b . $\square$

5.3.4 Extensions

In this section, we discuss several possible extensions of our RPE scheme.

Multi-bit Version. The RPE described in Scheme 5.3.1 requires that the message space contains only one bit. Using standard techniques for multi-bit **LWE**-based encryption, e.g., [2, 41, 87], with small overhead, we can achieve a τ -bit variant for any $\tau = \text{poly}(n)$. A notable modification in this case is that we will employ a revised encoding function $\text{encode}' : \{0, 1\}^\tau \rightarrow \{0, 1\}^{\tau+\kappa}$, where for any message $\mu \in \{0, 1\}^\tau$, vector $\text{encode}'(\mu)$ is obtained by appending $\kappa = \omega(\log n)$ zeros to μ .

Better Efficiency in the Random Oracle Model. The RPE from Scheme 5.3.1 has relatively large public parameters pp , i.e., of bit-size $(\tilde{O}(\ell) + O(N)) \cdot \tilde{O}(\lambda^2)$, for which the dependence on N is due to the fact that we have to associate each node θ in the binary tree with a uniformly random matrix in $\mathbf{D}_\theta \in \mathbb{Z}_q^{n \times m}$, in order to obtain full-hiding security in the standard model. Fortunately, the size of pp can be reduced to $\tilde{O}(\ell) \cdot \tilde{O}(\lambda^2)$ (which is comparable to that of the underlying PE scheme [4, 102]), if we work in the random oracle model [14]. The idea is as follows.

Let $\mathcal{H} : \{0, 1\}^* \rightarrow \mathbb{Z}_q^{n \times m}$ be a random oracle. Then, in the scheme, for each node θ , we obtain uniformly random matrix $\mathbf{D}_\theta$ as $\mathbf{D}_\theta := \mathcal{H}(\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \theta)$. The rest of the scheme remains the same. In the security proof, we first simulate the generation of $\mathbf{D}_\theta$ as in the proof of Theorem 5.3.2. Then, it remains to program the random oracle such that $\mathcal{H}(\mathbf{A}, \{\mathbf{A}_i\}_{i \in [\ell]}, \mathbf{U}, \theta) := \mathbf{D}_\theta$. This modification allows us to make the size of pp independent of N .

Exponentially Larger Ring. Finally, as shown in [51, 102], some applications of PE for inner-product predicates over R^ℓ (in our scheme, $R = \mathbb{Z}_q$) require that R has exponentially large cardinality, such as implementations of PE for CNF formulae [51]

and hidden vector encryption [20]. Under this circumstances, for our scheme, it requires to set the modulus q to be exponential in n . Hence, it would be desirable to achieve a lattice-based PE scheme supporting both revocation and exponentially large R , that requires only polynomial moduli. One possible solution towards tackling this question is to adapt the techniques introduced by Xagawa [102], where one works with $R = \text{GF}(q^n)$ instead of $\mathbb{Z}_q$.

Chapter 6

Conclusions and Open Problems

Throughout this thesis, we focused on how to equip lattice-based cryptosystems with efficient revocation mechanisms, possibly under the new revocation model we designed. In terms of cryptographic protocols, identity-based encryption (IBE) and predicate encryption (PE) schemes were considered. There were lattice-based revocable IBE systems in previous works and we improved the efficiency. Revocation has not been addressed for lattice-based PE schemes before and we gave first trials. For revocation methods, we considered the server-aided revocation mechanism and the direct revocation model. The former significantly improves efficiency of the key-update revocation model by outsourcing the key-update phase to a computationally powerful sever, while the latter further eliminates key updates by forcing ciphertexts to carry on the revocation information.

Adaptations of new revocation models for lattice-based cryptosystems emerge the need of new techniques. For the server-aided revocation mechanism, we observed that the key-splitting technique employed in existing pairing-based proposals seemed not easy to find lattice category. Instead, the double encryption and syndrome-splitting (which can be seen as a variant of secret sharing) techniques came to our notice. These two ideas are well-suited for the sever-aided setting and we carefully employed them in our constructions of such schemes.

For our revocable PE scheme following the direct revocation model, the main difficulty was to achieve the full-hiding security, which additionally guarantees the privacy of revocation information. We introduced one IBE instance into the system and took advantage of the anonymity of it.

Specifically, we summarize our results and techniques as follows.

In Chapter 3, we provided an server-aided revocable identity-based encryption (SR-IBE) scheme from lattices. Our construction is more efficient than all existing revocable IBE schemes from lattice assumptions. The detailed comparison was provided in Table 3.1. The heart of our construction is a “double encryption” paradigm which smoothly enables the server to assist in the decryption procedure. Roughly speaking, the plaintext is first encrypted using a hierarchical IBE (HIBE) scheme and the resulted ciphertext component carrying on the message bit is additionally encrypted using a revocable IBE (RIBE) instance. The server, possessing the RIBE private key, is able to recover the HIBE ciphertext, which can be decrypted by the recipient with his HIBE private key. Moreover, it is worth investigating whether such a double encryption paradigm would yield a generic construction for SR-IBE.

In Chapter 4, we formalized the notion of server-aided revocable predicate encryption (SR-PE) and put forward a construction of such scheme from lattices. The model inherits the main advantage of the server-aided revocation mechanism [89]: most of the users’ workloads are delegated to an untrusted server. Our lattice-based SR-PE has constant-size private keys and ciphertexts. It is also worth mentioning that, if we did not assume the availability of the server and let the users perform the server’s work themselves, then our scheme would yield the first (non-server-aided) lattice-based PE following the key-update revocation mechanism. The main difficulty is to make the partially decrypted ciphertext bound to the recipient’s identifying information. To this end, we employed one IBE instance and specified each recipient with an identity-determined matrix. The sever-side token then is embedded by such a matrix - so is the partially decrypted ciphertext. Moreover, this technique seems

difficult to yield SR-PE constructions in a modular way since we also employed a special property of LWE-based encryption schemes.

In Chapter 5, we described a lattice-based instantiation of revocable predicate encryption (RPE) based on the direct revocation approach, in which the revocation information is directly encoded into the ciphertext such that it is not necessary to update keys. We achieved full-hiding security thanks to the anonymity of the IBE instance we added. To combine the IBE block and the ordinary PE block, a splitting technique is adopted. Furthermore, the idea of introducing a IBE system to realize revocable PE schemes is natural when we consider to identify each private key. The challenge is how to enable such an IBE system smoothly interact with other ingredients.

Open Problems. In addition, there are open questions with respect to some of the schemes we have constructed in the thesis.

Our construction in Chapter 3 cannot thwart the decryption key exposure (DKE) attacks, proposed by Seo and Emura [96], which takes account of not only the exposure of a long-term private key but also the exposure of a short-term decryption key. Existing pairing-based revocable IBE schemes satisfying this strong notion all employed a re-randomization technique in the decryption key generation procedure, which seems hard to adapt into the lattice setting. Recently, Takayasu and Watanabe [100] proposed a lattice-based revocable IBE with bounded DKE resistance. They extended Chen et al.’s splitting technique [28] in the framework of cover-free families. The problem of achieving full DKE security for lattice-based revocable IBE schemes is still open.

Recall that all of our three constructions were shown to be secure in the selective manner. Achieving the stronger notion of adaptive security seems to require the underlying RIBE, HIBE, and PE schemes be adaptively secure. For revocable IBEs, as mentioned in [100], the selective setting (which requires the adversary to announce the challenge identity and time period before the execution of the setup algorithm) plays an important role in the security proof and achieving adaptive-ID security seems difficult. For PEs, as far as we know, existing lattice-based schemes [4, 38, 46, 102]

only achieved security in the selective setting. Designing SR-IBE, SR-PE, and RPE schemes achieving adaptive security are therefore left as open problems.

Moreover, it is worth to investigate whether lattice-based revocable cryptosystems can achieve the same (or even better) efficiency as the schemes under other assumptions. Two notable results from pairings were presented. First, Lewko et al. [57] gave a pairing-based instantiation of identity-based revocation system where the size of ciphertexts is linear in the number of revoked users and there is no need to pre-fix the maximum number of revoked users or the number of users. They made use of a “two-equation” technique to obtain this efficiency. Second, Chen et al. [27] gave a generic construction from non-zero inner-products PE to identity-based revocation system, where both private keys and ciphertexts have constant sizes but the size of public parameters is linear in the number of users. They also put forward a pairing-based non-zero inner-products PE scheme. To the best of our knowledge, there is still no such scheme under the **LWE** assumption.

Lattice-based cryptography has grown in popularity due to the potential quantum resistance feature and the capability of building powerful cryptographic primitives, such as fully homomorphic encryption [40] and predicate encryption for circuits [46]. In this thesis, we have provided detailed realizations of three cryptographic functionalities from lattices. In the future, we will continue to investigate how to utilize lattices to design new cryptosystems and to improve the existing constructions.

Bibliography

- [1] Shweta Agrawal, Sanjay Bhattacharjee, Duong Hieu Phan, Damien Stehlé, and Shota Yamada. Efficient public trace and revoke from standard assumptions: extended abstract. In *CCS 2017*, pages 2277–2293. ACM, 2017.
- [2] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572. Springer, 2010.
- [3] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In *CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 98–115. Springer, 2010.
- [4] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *ASIACRYPT 2011*, volume 7073 of *Lecture Notes in Computer Science*. Springer, 2011.
- [5] William Aiello, Sachin Lodha, and Rafail Ostrovsky. Fast digital identity revocation (extended abstract). In *CRYPTO 1998*, volume 1462 of *Lecture Notes in Computer Science*, pages 137–152. Springer, 1998.
- [6] Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC 1996*, pages 99–108. ACM, 1996.

- [7] Miklós Ajtai. Generating hard instances of the short basis problem. In *ICALP 1999*, volume 1644 of *Lecture Notes in Computer Science*, pages 1–9. Springer, 1999.
- [8] Miklós Ajtai, Ravi Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *STOC 2001*, pages 601–610. ACM, 2001.
- [9] Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48(3):535–553, 2011.
- [10] Nuttapong Attrapadung and Hideki Imai. Attribute-based encryption supporting direct/indirect revocation modes. In *Cryptography and Coding 2009*, pages 278–300. Springer, 2009.
- [11] Nuttapong Attrapadung and Hideki Imai. Conjunctive broadcast and attribute-based encryption. In *Pairing 2009*, volume 5671 of *Lecture Notes in Computer Science*, pages 248–265. Springer, 2009.
- [12] Nuttapong Attrapadung and Benoît Libert. Functional encryption for inner product: achieving constant-size ciphertexts with adaptive security or support for negation. In *PKC 2010*, volume 6056 of *Lecture Notes in Computer Science*, pages 384–402. Springer, 2010.
- [13] Joonsang Baek and Yuliang Zheng. Identity-based threshold decryption. In *PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 262–276. Springer, 2004.
- [14] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS 1993*, pages 62–73. ACM, 1993.
- [15] Alexandra Boldyreva, Vipul Goyal, and Virendra Kumar. Identity-based encryption with efficient revocation. In *CCS 2008*, pages 417–426. ACM, 2008.

- [16] Dan Boneh. The decision diffie-hellman problem. In *ANTS-III*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.
- [17] Dan Boneh, Xuhua Ding, Gene Tsudik, and Chi-Ming Wong. A method for fast revocation of public key certificates and security capabilities. In *USENIX Security Symposium*, pages 297–310. USENIX, 2001.
- [18] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. In *CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
- [19] Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT 2014*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556. Springer, 2014.
- [20] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC 2007*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.
- [21] Xavier Boyen. Attribute-based functional encryption on lattices. In *TCC 2013*, volume 7785 of *Lecture Notes in Computer Science*, pages 122–142, 2013.
- [22] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. On the classical hardness of learning with errors. In *STOC 2013*, pages 575–584. ACM, 2013.
- [23] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In *CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012.

- [24] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. In *FOCS 2011*, pages 97–106. IEEE Computer Society, 2011.
- [25] Ran Canetti, Shai Halevi, and Jonathan Katz. A forward-secure public-key encryption scheme. *J. Cryptology*, 20(3):265–294, 2007.
- [26] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.
- [27] Jie Chen, Benoît Libert, and Somindu C. Ramanna. Non-zero inner product encryption with short ciphertexts and private keys. In *SCN 2016*, pages 23–41. Springer, 2016.
- [28] Jie Chen, Hoon Wei Lim, San Ling, Huaxiong Wang, and Khoa Nguyen. Revocable identity-based encryption from lattices. In *ACISP 2012*, volume 7372 of *Lecture Notes in Computer Science*, pages 390–403. Springer, 2012.
- [29] Shantian Cheng and Juanyang Zhang. Adaptive-ID secure revocable identity-based encryption from lattices via subset difference method. In *ISPEC 2015*, volume 9065 of *Lecture Notes in Computer Science*, pages 283–297. Springer, 2015.
- [30] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding, 8th IMA International Conference*, volume 2260 of *Lecture Notes in Computer Science*, pages 360–363. Springer, 2001.
- [31] Hui Cui, Robert H. Deng, Yingjiu Li, and Baodong Qin. Server-aided revocable attribute-based encryption. In *ESORICS 2016*, volume 9879 of *Lecture Notes in Computer Science*, pages 570–587. Springer, 2016.

- [32] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976.
- [33] Xuhua Ding and Gene Tsudik. Simple identity-based cryptography with mediated RSA. In *CT-RSA 2003*, volume 2612 of *Lecture Notes in Computer Science*, pages 193–210. Springer, 2003.
- [34] Farid F. Elwailly, Craig Gentry, and Zulfikar Ramzan. Quasimodo: efficient certificate validation and revocation. In *PKC 2004*, volume 2947 of *Lecture Notes in Computer Science*, pages 375–388. Springer, 2004.
- [35] Amos Fiat and Moni Naor. Broadcast encryption. In *CRYPTO 1993*, volume 773 of *Lecture Notes in Computer Science*, pages 480–491. Springer, 1993.
- [36] Nicolas Gama and Phong Q. Nguyen. Finding short lattice vectors within Mordell’s inequality. In *STOC 2008*, pages 207–216. ACM, 2008.
- [37] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO 1984*, volume 196 of *Lecture Notes in Computer Science*, pages 10–18. Springer, 1984.
- [38] Romain Gay, Pierrick Méaux, and Hoeteck Wee. Predicate encryption for multi-dimensional range queries from lattices. In *PKC 2015*, volume 9020 of *Lecture Notes in Computer Science*, pages 752–776. Springer, 2015.
- [39] Nicholas Genise and Daniele Micciancio. Faster gaussian sampling for trapdoor lattices with arbitrary modulus. *IACR Cryptology ePrint Archive*, 2017:308, 2017.
- [40] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC 2009*, pages 169–178. ACM, 2009.

- [41] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC 2008*, pages 197–206. ACM, 2008.
- [42] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO 2013*, volume 8042 of *Lecture Notes in Computer Science*, pages 75–92. Springer, 2013.
- [43] Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2009.
- [44] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. In *CRYPTO 2015*, volume 9216 of *Lecture Notes in Computer Science*. Springer, 2015.
- [45] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC 2013*, pages 545–554. ACM, 2013.
- [46] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from LWE. In *CRYPTO 2015*, volume 9216 of *Lecture Notes in Computer Science*, pages 503–523. Springer, 2015.
- [47] Sergey Gorbunov and Dhinakaran Vinayagamurthy. Riding on asymmetry: efficient ABE for branching programs. In *ASIACRYPT 2015*, volume 9452 of *Lecture Notes in Computer Science*, pages 550–574. Springer, 2015.
- [48] Vipul Goyal. Certificate revocation using fine grained certificate space partitioning. In *FC 2007*, volume 4886 of *Lecture Notes in Computer Science*, pages 247–259. Springer, 2007.

- [49] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *CCS 2006*, pages 89–98. ACM, 2006.
- [50] Kamil Doruk Gur, Yuriy Polyakov, Kurt Rohloff, Gerard W. Ryan, and Erkey Savas. Implementation and evaluation of improved gaussian sampling for lattice trapdoors. *IACR Cryptology ePrint Archive*, 2017:285, 2017.
- [51] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.
- [52] Jianchang Lai, Yi Mu, Fuchun Guo, Willy Susilo, and Rongmao Chen. Anonymous identity-based broadcast encryption with revocation for file sharing. In *ACISP 2016*, volume 9723 of *Lecture Notes in Computer Science*, pages 223–239. Springer, 2016.
- [53] Kwangsu Lee, Intae Kim, and Seong Oun Hwang. Privacy preserving revocable predicate encryption revisited. *Security and Communication Networks*, 8(3):471–485, 2015.
- [54] Kwangsu Lee, Dong Hoon Lee, and Jong Hwan Park. Efficient revocable identity-based encryption via subset difference methods. *Designs, Codes and Cryptography*, 85(1):39–76, 2017.
- [55] A. K. Lenstra, H. W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.
- [56] Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: attribute-based encryption

- and (hierarchical) inner product encryption. In *EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 62–91. Springer, 2010.
- [57] Allison B. Lewko, Amit Sahai, and Brent Waters. Revocation systems with very small private keys. In *IEEE Symposium on Security and Privacy, 2010*, pages 273–285. IEEE Computer Society, 2010.
- [58] Jin Li, Jingwei Li, Xiaofeng Chen, Chunfu Jia, and Wenjing Lou. Identity-based encryption with outsourced revocation in cloud computing. *IEEE Trans. Computers*, 64(2):425–437, 2015.
- [59] Benoît Libert, Fabrice Mouhartem, and Khoa Nguyen. A lattice-based group signature scheme with message-dependent opening. In *ACNS 2016*, volume 9696 of *Lecture Notes in Computer Science*, pages 137–155. Springer, 2016.
- [60] Benoît Libert, Thomas Peters, and Moti Yung. Group signatures with almost-for-free revocation. In *CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 571–589. Springer, 2012.
- [61] Benoît Libert, Thomas Peters, and Moti Yung. Scalable group signatures with revocation. In *EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 609–627. Springer, 2012.
- [62] Benoît Libert and Jean-Jacques Quisquater. Efficient revocation and threshold pairing based cryptosystems. In *PODC 2003*, pages 163–171. ACM, 2003.
- [63] Benoît Libert and Damien Vergnaud. Adaptive-ID secure revocable identity-based encryption. In *CT-RSA 2009*, volume 5473 of *Lecture Notes in Computer Science*, pages 1–15. Springer, 2009.
- [64] Benoît Libert and Damien Vergnaud. Towards black-box accountable authority IBE with short ciphertexts and private keys. In *PKC 2009*, volume 5443 of *Lecture Notes in Computer Science*, pages 235–255. Springer, 2009.

- [65] San Ling, Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Server-aided revocable predicate encryption: definition and lattice-based instantiation. In *Submission to Designs, Codes and Cryptography*.
- [66] San Ling, Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Revocable predicate encryption from lattices. In *ProvSec 2017*, volume 10592 of *Lecture Notes in Computer Science*, pages 305–326. Springer, 2017.
- [67] Zhen Liu and Duncan S. Wong. Practical ciphertext-policy attribute-based encryption: traitor tracing, revocation, and large universe. In *ACNS 2015*, volume 9092 of *Lecture Notes in Computer Science*, pages 127–146. Springer, 2015.
- [68] Alfred Menezes, Scott A. Vanstone, and Tatsuaki Okamoto. Reducing elliptic curve logarithms to logarithms in a finite field. In *ACM Symposium on Theory of Computing*, pages 80–89. ACM, 1991.
- [69] Silvio Micali. Efficient certificate revocation. In *Technical Report MIT/LCS/TM-542b*, 1996.
- [70] Silvio Micali. Scalable certificate validation and PKI management. In *PKI Research Workshop*, 2002.
- [71] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 465–484. Springer, 2011.
- [72] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: simpler, tighter, faster, smaller. In *EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012.
- [73] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM Journal on Computing*, 37(1):267–302, 2007.

- [74] Daniele Micciancio and Michael Walter. Gaussian sampling over the integers: efficient, generic, constant-time. In *CRYPTO 2017*, volume 10402 of *Lecture Notes in Computer Science*, pages 455–485. Springer, 2017.
- [75] Dalit Naor, Moni Naor, and Jeffery Lotspiech. Revocation and tracing schemes for stateless receivers. In *CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 41–62. Springer, 2001.
- [76] Moni Naor and Benny Pinkas. Efficient trace and revoke schemes. In *FC 2000*, volume 1962 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2000.
- [77] Khoa Nguyen, Huaxiong Wang, and Juanyang Zhang. Server-aided revocable identity-based encryption from lattices. In *CANS 2016*, volume 10052 of *Lecture Notes in Computer Science*, pages 107–123. Springer, 2016.
- [78] Juan Manuel González Nieto, Mark Manulis, and Dongdong Sun. Fully private revocable predicate encryption. In *ACISP 2012*, volume 7372 of *Lecture Notes in Computer Science*, pages 350–363. Springer, 2012.
- [79] Juan Manuel González Nieto, Mark Manulis, and Dongdong Sun. Fully private revocable predicate encryption. *IACR Cryptology ePrint Archive*, 2012:403, 2012.
- [80] A. M. Odlyzko. The rise and fall of knapsack cryptosystems. In *In Cryptology and Computational Number Theory*, pages 75–88. A.M.S, 1990.
- [81] Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In *ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 214–231. Springer, 2009.
- [82] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 191–208. Springer, 2010.

- [83] Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In *CANS 2011*, volume 7092 of *Lecture Notes in Computer Science*, pages 138–159. Springer, 2011.
- [84] Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In *EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 591–608. Springer, 2012.
- [85] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC 2009*, pages 333–342. ACM, 2009.
- [86] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In *CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 2010.
- [87] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 554–571. Springer, 2008.
- [88] Chris Peikert and Brent Waters. Lossy trapdoor functions and their applications. In *STOC 2008*, pages 187–196. ACM, 2008.
- [89] Baodong Qin, Robert H. Deng, Yingjiu Li, and Shengli Liu. Server-aided revocable identity-based encryption. In *ESORICS 2015*, volume 9326 of *Lecture Notes in Computer Science*, pages 286–304. Springer, 2015.
- [90] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC 2005*, pages 84–93. ACM, 2005.
- [91] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

- [92] Amit Sahai, Hakan Seyalioglu, and Brent Waters. Dynamic credentials and ciphertext delegation for attribute-based encryption. In *CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 199–217. Springer, 2012.
- [93] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473. Springer, 2005.
- [94] Yusuke Sakai, Keita Emura, Goichiro Hanaoka, Yutaka Kawai, Takahiro Matsuda, and Kazumasa Omote. Group signatures with message-dependent opening. In *Pairing 2012*, volume 7708 of *Lecture Notes in Computer Science*, pages 270–294. Springer, 2012.
- [95] Claus-Peter Schnorr. Factoring integers and computing discrete logarithms via diophantine approximations. In *EUROCRYPT 1991*, volume 547 of *Lecture Notes in Computer Science*, pages 281–293. Springer, 1991.
- [96] Jae Hong Seo and Keita Emura. Revocable identity-based encryption revisited: security model and construction. In *PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 216–234. Springer, 2013.
- [97] Adi Shamir. How to share a secret. *Commun. ACM*, 22(11):612–613, 1979.
- [98] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO 1984*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1985.
- [99] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.
- [100] Atsushi Takayasu and Yohei Watanabe. Lattice-based revocable identity-based encryption with bounded decryption key exposure resistance. In *ACISP 2017*,

- volume 10342 of *Lecture Notes in Computer Science*, pages 184–204. Springer, 2017.
- [101] Brent Waters. Dual system encryption: realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO 2009*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.
- [102] Keita Xagawa. Improved (hierarchical) inner-product encryption from lattices. In *PKC 2013*, volume 7778 of *Lecture Notes in Computer Science*, pages 235–252. Springer, 2013.
- [103] Shota Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In *EUROCRYPT 2016*, volume 9666 of *Lecture Notes in Computer Science*, pages 32–62. Springer, 2016.
- [104] Yanjiang Yang, Joseph K. Liu, Kaitai Liang, Kim-Kwang Raymond Choo, and Jianying Zhou. Extended proxy-assisted approach: achieving revocable fine-grained encryption of cloud data. In *ESORICS 2015*, volume 9327 of *Lecture Notes in Computer Science*, pages 146–166. Springer, 2015.