博士論文

Enhancing Security of Efficient Advanced Cryptosystems via Reduction to Reliable Assumptions
（高効率な高機能暗号方式の安全性強化手法に関する研究）

勝又秀一

Abstract

The introduction of *public-key cryptography* by Diffie and Hellman (IEEE TIT, 1976) has had an enormous influence on the current information society. Starting with the most fundamental primitive of public-key encryption schemes, the field of public-key cryptography has been enriched with many alluring advanced primitives. In cryptography, when we say a scheme is "secure", we implicitly have in mind the concept of *provable security* — a notion introduced by Glodwasser and Micalli (STOC, 1982). Informally, we have a set of hardness assumptions which we rely on to build provably secure cryptographic primitives. As one can imagine, as the cryptographic primitives become more advanced, it is generally the case that we require a stronger hardness assumptions to construct them. However, proving a cryptographic primitive secure under a strong hardness assumption is undesirable since the security guarantee for the primitive we achieve will be weaker, and moreover, often times we would have to compensate for the security loss by making the concrete instantiation less efficient.

This Ph.D. thesis is a study of different approaches to make advanced cryptosystems more secure by constructing them from weaker, hence more reliable, hardness assumptions. We broadly prepare three measures which we can use to assess the hardness of a problem: whether it is a search problem or a decision problem, whether it is a static-problem or a non-static problem, and whether it is post-quantum or not. Following these three measures, we enhance the security guarantees (and in some cases its concrete efficiency) of advanced cryptographic primitives. The main contributions are contained in the five papers included in this thesis and cover the following primitives: identity-based encryption (IBE) schemes, verifiable random functions (VRF), predicate encryption (PE) schemes, non-zero inner product encryption (NIPE) schemes, and attribute-based signature (ABS) schemes.

Concretely, the first two papers concern IBE schemes. In the first paper, we show an alternative security proof for a state-of-the-art post-quantum IBE scheme and show that we can enhance its security without modifying the original scheme. In the second paper, we provide a general framework for proving IBE schemes by implicitly embedding non-linear polynomial functions in the public parameters and obtain two IBE schemes with better security guarantees and more compact public parameters compared to previous schemes. The third paper concerns VRFs and PE schemes. We show how to encode predicates by shallow arithmetic circuits and how to combine them with VRFs and PEs to obtain schemes with better security guarantees. The forth paper concerns NIPE schemes. We provide two different methods for obtaining NIPE schemes from various assumptions. The final paper concerns ABS schemes. We provide a generic construction of ABS schemes for unbounded circuits and instantiate it with post-quantum tools to obtain the first such scheme based on a post-quantum assumption.

Keywords: Provable Security, Weaker Assumptions, Identity-Based Encryption, Verifiable Random Functions, Predicate Encryption, Non-Zero Inner-Product Encryption, Attribute-Based Signatures

Acknowledgment

First and foremost, I wish to thank Noboru Kunihiro, my advisor. He was my advisor back when I was a bachelor studying cryptography and he was also the one who invited me back to the alluring world of cryptography for my Ph.D., after I briefly left cryptography to study optimization and machine learning during my masters — which I also enjoyed very much. Although the topics I studied never overlapped with Noboru's, I could not have asked for a more perfect advisor. I cannot thank him enough for all the support he has provide me over the past years. No matter what kind of action I took or decision I made, he was always supportive and helpful all the way through, and his trust in me undoubtedly nourished my confidence as a researcher.

Shota Yamada was my first co-author during my Ph.D and was an excellent mentor to me. I am very grateful for his patience and for the long hours he spent listening to my half-baked ideas. His precise comments and clear view always helped me organize my utterly confused thoughts, and his way of thinking taught me how to be formal, logical, and precise, all of which form the backbone of how I do research today. I can say without a doubt that without having been able to work with him, I would have not understood cryptography in a way that I do now.

Takahiro Matsuda was always available to answer any technical question I had and taught me much about formality. He provided me with many valuable comments on the draft of my first single-authored paper and opened up a new world for me. This was one of the turning points during my Ph.D. as I began to see and understand much better the deep theoretical world of cryptography. I am grateful for this.

One of the first researcher I visited abroad was Ali El Kafaarani at Oxford. He opened up his house for me, and helped me both as a researcher and as a friend, and I have been visiting him occasionally ever since. He has a great positive attitude with an unbelievable amount of energy, and often times I was saved by it when I was feeling down. It was always fulfilling to work with him and I am very grateful for this.

I am very grateful to Atsushi Takayasu for always making time for me whenever I was stressed from my studies. The short breaks which we often took on campus in between research always refreshed my mind and allowed me to get back to my research more motivated than before. Tasuku Soma and Yuji Nakatsukasa, whom I known from my former lab during masters, were always open for discussion even if cryptography were not their expertise. Often times conversations with them led me to find connections between seemingly unrelated phenomena. I am very grateful for them.

During my Ph.D., I had the pleasure of doing research at the National Institute of Advanced Industrial Science and Technology (AIST). Goichiro Hanaoka, the head of the cryptography group at AIST, has always been extremely generous and sincere to me, and the research environment he prepared for me at AIST has had a great positive influence on my research. Other researchers at AIST that have deeply influenced my research include Jacob Schuldt, Yusuke Sakai, and Takao Murakami. I am very grateful for all of them.

During my short internship at Mitsubishi Electronic around the end of my masters, I had the fortune of interacting with Katsuyuki Takashima. Even though he was not my direct host, he looked after me throughout my stay and engaged in long hours of discussion. In the early stage of my research, I did not know anything better about cryptography, and as such, he must have suffered from my disorganized and illogical thoughts. I am grateful for his patience. I am also very happy to still be in in touch with him after all this long.

I am also grateful for Phong Nguyen, Takashi Yamakawa, Kazumasa Shinagawa, and Yohei Watanabe. I am especially grateful to Phong for always encouraging me to go abroad and to open up the small world that I was living in. It was always inspiring and thrilling to work with Takashi. Observing his rapid thought process, unusual association and amazing ability to put together seemingly unconnected ideas have always motivated me to strive for more.

Finally, my parents Keiko Katsumata and Kenichi Katsumata and my sister Yuko Katsumata.

Please forgive me any omissions.

List of Publications

This Ph.D. thesis comprises a collection of five publications devoted to enhance the security of advanced cryptosystems through reductions to more reliable assumptions. The concrete cryptographic primitives that we consider in this thesis are: identity-based encryption schemes (in Chapters 3 and 4), verifiable random functions and predicate encryption schemes (in Chapter 5), non-zero inner-product encryption schemes (in Chapter 6), and attribute-based signature schemes (in Chapter 7). The details of the publications which are comprised in each chapter are listed below.

Chapter 3 [KYY18] *Tighter Security Proofs for GPV-IBE in the Quantum Random Oracle Model.* Shuichi Katsumata, Shota Yamada, and Takashi Yamakawa. In the 24nd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2018.

Chapter 4 [KY16] *Partitioning via Non-Linear Polynomial Functions: More Compact IBEs from Ideal Lattices and Bilinear Maps.* Shuichi Katsumata and Shota Yamada. In the 22nd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2016.

Chapter 5 [Kat17] *On the Untapped Potential of Encoding Predicates by Arithmetic Circuits and Their Applications.* Shuichi Katsumata. In the 23nd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2017.

Chapter 6 [KY19b] *Non-Zero Inner Product Encryption Schemes from Various Assumptions: LWE, DDH and DCR.* Shuichi Katsumata and Shota Yamada. In the 22st International Conference on Practice and Theory of Public-Key Cryptography (PKC), 2019.

Chapter 7 [EK18] *Attribute-Based Signatures for Unbounded Circuits in the ROM and Efficient Instantiations from Lattices.* Ali El Kaafarani and Shuichi Katsumata. In the 21st International Conference on Practice and Theory of Public-Key Cryptography (PKC), 2018.

Other articles written during my Ph.D., but not included in this thesis, are:

[EKS17] *Anonymous Reputation Systems Achieving Full Dynamicity from Lattices.* Ali El Kaafarani, Shuichi Katsumata, and Ravital Solomon. In the 22nd International Conference on Financial Cryptography and Data Security (FC), 2017.

[SKAH18] *Attribute-Based Signatures for Unbounded Languages from Standard Assumptions* Yusuke Sakai, Shuichi Katsumata, Nuttapong Attrapadung, and Goichiro Hanaoka. In the 24nd International Conference on the Theory and Application of Cryptology and Information Security (ASIACRYPT), 2018.

[KMT19] *Lattice-based Revocable (Hierarchical) IBE with Decryption Key Exposure Resistance.* Shuichi Katsumata, Takahiro Matsuda, and Atsushi Takayasu. In the 22st International Conference on Practice and Theory of Public-Key Cryptography (PKC), 2019.

[DKNY18] *Constrained PRFs for Bit-fixing from OWFs with Constant Collusion Resistance.* Alex Davidson, Shuichi Katsumata, Ryo Nishimaki, and Shota Yamada. Cryptology ePrint Archive, Report 2018/982, 2018.

[KY19a] *Group Signatures without NIZK: From Lattices in the Standard Model.* Shuichi Katsumata and Shota Yamada. (*Manuscript.*)

[KNYY19] *Designated Verifier/Prover and Preprocessing NIZKs from Diffie-Hellman Assumptions.* Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. (*Manuscript.*)

Contents

Chapter 1

Introduction

1.1 Background

In 1976, Diffie and Hellman published a paper with the title "New Directions in Cryptography" [DH76]. In contrast to the rather simplistic title, the paper introduced the foundation of modern *public-key cryptography* and has had an enormous influence on the current information society. It is not an exaggeration to say that the privacy and security of all the communications we make in our everyday lives through unauthenticated public networks are ensured by public-key cryptography. Though [DH76] introduced the notion of public-key cryptography, they did not provide any candidate constructions. It was Rivest, Shamir, and Adleman in 1978 [RSA78] that introduced the first construction of public-key encryption scheme — by now the famous RSA encryption scheme. The variants of the RSA encryption scheme are still the most widely used today. Soon after, Goldwasser and Micalli [GM82, GM84] provided a more rigorous treatment on the security notion of public-key cryptography and inspired the subsequent works to the concept of *provable security*. However, before diving into the alluring world of provable security, we would like to continue on with the brief history after the introduction of public-key cryptography and see what kind of other fascinating primitives have emerged since then.

Although public-key encryption schemes were a far more advanced and flexible primitive compared to secret-key encryption schemes (which can only be used if a secret-key has been established between the parties), researchers were eager to find more advanced primitives. One such prominent cryptographic primitive is the identity-based encryption (IBE) scheme. The concept of IBE schemes were introduced by Shamir in 1985 [Sha85] soon after the introduction of public-key encryption schemes. An IBE scheme is a public-key cryptographic system where any string, e.g., a user identifier such as an email address or phone number, can be a valid public key. Therefore, a sender can encrypt a message by simply using the receiver's identifier and does not require the receiver to actively setup a public key prior to the communication. In some systems where the identifier of the users are already known, IBE schemes offer a much cleaner and flexible solution to secure communication compared to using standard public-key encryption schemes. To come up with a concrete construction of IBE schemes however took time. It was not until a decade later that Boneh and Franklin [BF01] and Cocks [Coc01] independently constructed the first IBE schemes. The ideas and new mathematical tools (i.e., pairing or bilinear maps) presented in the former construction have found countless new applications and was awarded the Gödel prize in 2013 for its contribution to modern cryptography. Other compelling advanced public-key cryptographic primitives after the emergence of public-key encryption schemes include, but not limited to, are broadcast encryption schemes [FN93], proxy encryption schemes [BBS98], threshold encryption schemes [DF89], attribute-based encryption schemes [SW05, GPSW06], functional encryption schemes [O'N10, BSW11], and fully-homomorphic encryption schemes [RAD78, Gen09]. Furthermore, we would like to point out that advanced cryptographic primitives in the secret-key setting have also shown significant progress: such primitives include group signatures [CVH91], ring signatures [RST01], attribute-based signatures [MPR11], and secret-key counterparts of the above advanced public-key cryptographic primitives.[1]

We now return back to the concept of provable security introduced by Goldwasser and Micali [GM82, GM84], which is considered as now the defacto standard of modern cryptography. To appreciate this concept, we must be aware of the fact that when we state that a certain cryptographic primitive is "secure", the term is quite ambiguous. First of all, what does it even mean for a cryptographic primitive to be secure. Second, once we have a vague meaning of the term "secure", how should we formalize it in a strict mathematical language without ambiguity. Finally, how do we theoretically prove that a cryptographic primitive is secure. Formally treating these questions leads us to the concept of *provable security*.

The first two questions can be answered by formally laying out the security requirements a certain primitive should have. For instance, let us consider public-key encryption schemes. Our intuition tells us that a ciphertext ct should not leak any information of the message M which it encrypts. To capture this in a more formal manner, Goldwasser and Micali [GM82, GM84] introduced the notion of semantic-security for public-key encryption schemes. Informally, a semantically secure public-key encryption scheme demands that any information on the message M that can be efficiently computed from its encryption ct can also be efficiently computed without access to ct. In other words, any information that can be extracted from the ciphertext ct could have been simulated without knowledge of ct, hence, ct does not leak any information on the message M. However, we should note here that the definition of "security" is somewhat objective and in some cases there can be several incomparable security notions for one cryptographic primitive. For example, other security notions capturing what a "secure" public-key encryption scheme should be have been considered, e.g., security against indistinguishable from chosen plaintext attacks (IND-CPA) [GM82], IND-chosen ciphertext attacks (IND-CCA) [RS91], and non-malleability [DDN91]. In some cases these independently introduced security notions turn out to be equivalent, e.g., semantic security implies IND-CPA for public-key encryption schemes and vice-versa, however, in some other cases one notion is stronger than the other, e.g., IND-CPA is not known to imply IND-CCA.

The third question of how to theoretically prove security of a cryptographic primitive can be answered by using the technique known as *reductions*. A reduction works as follows: We first must make an assumption that there exists a hard problem X such that no algorithm can solve.[2] For example, in the seminal work of Diffie and Hellman [DH76], they assumed the hardness of the discrete logarithm (DL) problem, which states that given a group generator $g \in \mathbb{G}$ and some random group element $h \in \mathbb{G}$, it is difficult to find $a \in \mathbb{Z}_{|\mathbb{G}|}$ such that $h = g^a$. We then prove by contradiction that there exists no algorithm $\mathcal{A}$ breaking the security of the cryptographic primitive assuming the hardness of the problem X. Specifically, we *reduce* the problem of breaking the security of the cryptographic primitive to solving the hard problem X; we construct an algorithm $\mathcal{B}$ solving problem X by using the algorithm $\mathcal{A}$ which breaks the security of the cryptographic

[1] The names "public" and "secret" may be misleading. We follow the convention of Impagliazzo and Rudich [IR89] and call any primitives implied from one-way functions as "secret-key primitives" and those that are not as "public-key primitives".

[2] During the introduction, we leave the terms "hard" and "no algorithms" informal for the sake of clarity and readability.

primitive. In other words, unless the hard problem X cannot be solved by any algorithm (which is an assumption), then we can conclude that there exists no algorithm $\mathcal{A}$ that breaks the security of the cryptographic primitive.

To sum up so far, provable security is the concept of formally defining the correct notion of security for a cryptographic primitive and proving its security via a reduction from a hard problem. Therefore, once the correct notion of security is fixed, the problem boils down to what kind of hard problem, hence, what kind of hardness assumption, can be used to show a reduction to the security of the cryptographic primitive in question.[3] We should note that there exist cryptographic primitives with appropriate security definitions that are known to be provably secure without relying on any hardness assumptions such as Shamir's secret sharing scheme [Sha79], non-interactive zero-knowledge for **NP** languages in the hidden-bit model [FLS99], and garbled circuits for $\mathbf{NC}^1$ [IK02]; in particular, they only rely on information theoretical (or statistical) arguments to be proven secure and they are secure against any unbounded probabilistic algorithms. However, in this thesis, we will only be considering primitives that require some type of hardness assumption to prove secure.

Thus far, all provably secure public-key cryptographic schemes (and many secret-key cryptographic schemes) have relied on some type of hardness assumption. The first public-key encryption scheme of Rivest, Shamir, and Adleman [RSA78] relies on the hardness of the RSA problem and the public-key encryption scheme of El Gamal [ElG84] relies on the hardness of the decisional Diffie-Hellman (DDH) problem. However, as one can imagine, when we move to more advanced cryptographic primitives, it becomes more and more difficult to construct them from simple/easy hard problems (or equivalently simple/weak hardness assumptions). Here we say a problem X is easier than a problem Y if there exists a reduction from problem X to Y. Namely, problem X is easy in the sense that an algorithm solving Y can be converted to an algorithm solving X but not the other way around. For example, we all know by definition that the DL problem is harder than the DDH problem and that the DDH problem is harder than the decisional bilinear Diffie-Hellman (DBDH) problem. Here, we emphasize that we are not stating that the DBDH problem is an easy to solve problem. One possible way to interpret this ordering of the hardness of the problems would be as follows: even though the DBDH problem may be easier to solve compared to the DL or DDH problems, i.e., the DBDH problem is a stronger hardness assumption, solving the DBDH problem is nonetheless believed to be difficult, and moreover, due the gap between the hardness, the DBDH problem may potentially have a more flexible algebraic structure which we can exploit to construct more advanced and complex cryptographic primitives. Indeed, we do not know how to construct public-key encryption schemes from the DL problem, but we know how to construct them from the easier DDH problem [ElG84]. Furthermore, the first IBE construction of Boneh and Franklin [BF01] relied on the hardness of the DBDH problem (in the random oracle model), but it took more than a decade and a half till Döttling and Garg solved the long standing open problem of constructing IBE schemes from the weaker DDH assumption [DG17]. This captures our intuition that IBE schemes are more advanced primitives compared to public-key encryption schemes. Even though all problems we use to build cryptographic primitives are believed to be hard, there may exist an implicit gap between the hardness. For instance, it may turn out that it is impossible to construct public-key encryption schemes from the DL problem and there may be a fundamental gap between the DL problem and the DDH problem.

As we have seen so far, although there are some exceptions, in general, we require more complex and structured (resp. stronger) hard problems (resp. hardness assumptions) to construct

[3] We will be using the terms "hard problems" and "hardness assumptions" interchangeably whenever the meaning is clear.

advanced cryptographic primitives. However, one thing we must always keep in mind — and as a matter of fact is the main theme of this thesis — is that a hardness of a problem is no more than an assumption. Put differently, even if we are able to construct appealing advanced cryptographic primitives, if the construction comes at the cost of an extremely strong assumption, then we must take great precaution and check that the assumption really holds. Indeed, there are situations where some assumed hard problems have been shown to be easily solvable soon after their proposal. A famous example may be the public-key encryption scheme based on the variant of the knapsack problem proposed by Merkle and Hellman [MH78]. A few years after their proposal, Shamir proposed an efficient algorithm for the problem [Sha82] and the Merkle-Hellman knapsack problem has not been used ever since. Therefore, although constructing advanced cryptographic primitives are certainly of great importance, we must also always keep in mind of the credibility of the hardness assumption being made as it is the foundation of provable security.

In the next section, we see what kind of assumptions are believed to more reliable than others, hence, more suitable for constructing cryptographic primitives from a security stand point. Then, in the main body of our thesis, we will explore various advanced cryptographic primitives that are provably secure under such reliable assumptions.

1.2 Reliable Assumptions

When constructing cryptographic primitives, it is important to ask ourselves under what type of hard problems can we obtain provable security. Broadly speaking, there are three measures which we can use to assess the hardness of the problems: whether it is a search problem or a decision problem, whether it is a static-problem or a non-static problem, and whether it is post-quantum or not.

The first measure is perhaps easiest to explain through a concrete example. Let us consider the search variant of the famous factorization problem: Given a number $N \in \mathbb{N}$, find an integer d with $1 < d < N$ that divides N. Although this problem is believed not to be **NP**-complete, it is widely suspected to be outside of **P**. Many have tried to come up with a polynomial time algorithm for solving the problem but history has shown that it may be more difficult than the innocent impression casted by the problem. On the other hand, the decision variant of the factorization problem, which states that given a number $N \in \mathbb{N}$, decide whether N is a prime or not, is surprisingly known to be in **P** [AKS04]. In general, search problems are harder than decision problems. A more relative problem to cryptography that lets us appreciate the difference between search and decision problems may be the computational Diffie-Hellman (CDH) problem and the decisional Diffie-Hellman (DDH) problem [DH76]. While the CDH problem asks to compute, i.e., search for, $g^{ab} \in \mathbb{G}$ given a random tuple $(g, g^a, g^b) \in \mathbb{G}^3$, the DDH problem asks to decide whether $Z \in \mathbb{G}$ is equal to g^{ab} or a uniformly random element given a random tuple $(g, g^a, g^b) \in \mathbb{G}^3$. It is easy to verify that the DDH problem is easier than the CDH problem. Notably, if there exists an algorithm solving the CDH problem, it can be trivially used to solve the DDH problem, however, the converse does not seem to be hold. Furthermore, unlike the decision variant of the factorization problem, the DDH problem is believed to be outside of **P** for certain groups $\mathbb{G}$; it can be used as a hardness assumption to construct meaningful cryptographic primitives. Therefore, when one wants to construct a cryptographic scheme, they are given the choice of using either the weaker CDH assumption or the stronger DDH assumption. In general, in case a problem can be casted as both a search and decision problems and both problems are assumed to be hard, then it is preferable from a security point of view to construct a provably secure cryptographic primitive under the search problem, since it provides stronger security guarantees. Specifically, even if at some point the decision problem (and not the search problem) turns out to be an easy problem to solve, the security of a cryptographic scheme based on the search problem will still be intact. We note that these unfortunate situations may truly happen taking into account that there are some groups known as the gap Diffie-Hellman groups [BLS01, OP01] where the CDH problem is assumed to be intractable but the DDH problem has practical solutions.

It is worth pointing out that for some special problems, it is known that the search problem reduces to the decision problem; the decision problem is as hard as the search problem. One notable example is the learning with errors (LWE) problem [Reg05, Reg10]. Informally, the search variant of the LWE problem asks to find a vector $\mathbf{s} \in \mathbb{Z}_q^n$, given a matrix-vector pair $(\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$, where $\mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{e}$ for some short vector $\mathbf{e}$ sampled from a specific distribution. The decision variant of the LWE problem asks to distinguish whether $\mathbf{b} = \mathbf{A}^\top \mathbf{s} + \mathbf{e}$ for some vector $\mathbf{s}$ and short vector $\mathbf{e}$ or $\mathbf{b}$ a uniformly random vector over $\mathbb{Z}_q^m$ given the pair $(\mathbf{A}, \mathbf{b})$. Although at first glance, the decisional version of the LWE problem seems much easier than the search variant, Regev [Reg05] provided a reduction of the search LWE to the decision LWE. Hence, for some problems, it does not make a significant difference whether the search or decision version is used.

The second measure of static or non-static problem is mainly relevant to specific types of problems where the hardness assumption grow dynamically. One of the earliest problems that has the non-static feature is the q-generalized Diffie-Hellman (q-GDH) problem [STW96, NR97]. This problem is an extension of the original DDH problem and asks the following: for a random generator $g \in \mathbb{G}$, given $g^{a_1}, \cdots, g^{a_q} \in \mathbb{G}$ and given all the subset products $g^{\prod_{i \in S} a_i} \in \mathbb{G}$ for any strict subset $S \subset \{1, \cdots, q\}$, compute $g^{a_1 \cdots a_q} \in \mathbb{G}$. Here q is some external parameter which is chosen dynamically, hence, the terminology non-static assumption. It is easy to see that the 2-GDH problem is equivalent to the CDH problem and for any $q \geq 1$, the $(q+1)$-GDH problem is easier than the q-GDH problem. Since most non-static assumptions are parameterized by a value q, non-static assumptions are also often times referred to as *q-type assumptions*. As we mentioned earlier, the growth in the complexity of cryptographic primitives has been accompanied by an analogous growth in the complexity of the assumptions required to prove security; q-type assumptions are no exception. Different types of q-type assumptions have been created in exchange for constructing advanced cryptosystems. Some examples are the q-strong Diffie-Hellman (q-SDH) assumption [MSK02], the q-bilinear Diffie-Hellman inversion (q-BDHI) assumption [BB04b], and the q-bilinear Diffie-Hellman exponent assumption [BBG05, BGW05]. Often times, in a reduction, the value of q is tied to the number of queries that the adversary makes to some oracle. For example, in the IBE scheme of [BB04b], q in the q-BDHI assumption must be at least as large as the number of parties the adversary is able to corrupt, which can (in theory) be an arbitrary large polynomial. Since q-type assumptions are stronger for larger values of q, this means that the assumption we require gets stronger as the adversary is able to corrupt more parties. Therefore, if we were to prove a cryptographic scheme secure under a q-type assumption, we want the value of q to be as small as possible. This is not only of a theoretical concern, since for example Cheon [Che06] showed an attack that recovers the exponent of the q-SDH assumption in time that scales inversely with q. This suggests that cryptographic schemes relying on larger values of q must compensate for using a stronger q-type assumption by setting the parameters large enough to maintain a constant level of security.

The last measure concerns whether a problem is post-quantum or not. Here we say a problem X is post-quantum if it cannot be solved even if quantum algorithms are used. Above we mentioned that the factorization problem is believed to be outside of $\mathbf{P}$. For a quantum computer, however, Shor [Sho94a] presented an algorithm that solves the problem in polynomial time. While quantum computation in not yet viable, the emergence of quantum computers will have a significant impact on cryptography. For instance the RSA encryption scheme will completely be broken since its security relies (roughly) on the hardness of the factorization problem. Moreover, Shor also showed in the same paper that the DL problem can be solved in quantum polynomial time. This in particular implies that all group-based and paring-based assumptions such as the CDH, DDH, DBDH problem, and so on, can be solved efficiently using quantum algorithms. Therefore, when quantum computers are fully realized, most of the cryptosystems used today will be completely insecure. This has provoked the cryptographic community in a search for (classical) cryptographic systems that are secure even in front of quantum algorithms. Recently in 2016, the National Institute of Standards and Technology (NIST) initiated the Post-Quantum Cryptography Standardization, and post-quantum cryptography has been gathering increasingly more attention. From a provable security point of view, a cryptographic scheme is quantumly-secure if the security of the scheme can be based on a hard problem that is post-quantum. One of the promising set of problems that are believed to be post-quantum are those based on lattices. In a seminal work, Ajtai [Ajt96] introduced the short integer solutions (SIS) problem and proved that solving it (on average) is at least as hard as approximating various lattice problems that are presumably post-quantum in the worst case. The aforementioned LWE problem introduced by Regev [Reg05] is another very important post-quantum problem. Similarly to the SIS problem it enjoys a worst-case to average-case reduction to approximating lattice problems that are presumably post-quantum. Originally, lattice-based cryptography, i.e., cryptographic schemes based on the SIS and LWE problems, were most acclaimed for its post-quantum feature and strong security guarantees from worst-case hardness, however, over the past few decades it has proven to be much more. For example, fully-homomorphic encryption (FHE) schemes — the holy grail of cryptography — first envisioned by Rivest et al. [RAD78], were finally given a candidate construction by Gentry [Gen09] based on lattices three decades later. Since then FHE has shown significant improvement both on the practical and theoretical front [BV11, GSW13, CGGI16], and all schemes so far are based on lattices. Other than FHE, lattices have provided the only known realizations of other advanced and powerful cryptographic primitives, such as attribute-based encryption schemes with arbitrary access policies [GVW13, BGG$^+$14a], fully-homomorphic signatures [GVW15b], and indistinguishable obfuscation [GGH$^+$13]. It may seem that lattice-based cryptography is all powerful, however, there are certain cryptographic primitives that we know how to construct from non-post-quantum problems but not from lattice-based problems. Some long standing open problems are constructing lattice-based non-interactive zero-knowledge proofs [BFM88] and broadcast encryption schemes [FN93]. Therefore, it is important to keep in mind what we can construct from lattices and how efficiently we can construct them compared to the non-post-quantum counterparts.

1.3 Our Contribution

In this thesis, our goal is to construct advanced cryptographic primitives efficiently while basing their securities on more reliable assumptions: search problems, static-assumptions or post-quantum assumptions. Below, we provide a brief summary of the contributions made in each subsequent chapters.

1.3.1 Summary of Chapter 3

In 2008, Gentry, Peikert, and Vaikuntanathan [GPV08] proposed the first identity-based encryption (GPV-IBE) scheme based on a post-quantum assumption, namely, the LWE assumption. Since their proof was only made in the random oracle model (ROM) instead of the *quantum* random oracle model (QROM), it remained unclear whether the scheme was truly post-quantum or not. In 2012, Zhandry [Zha12b] developed new techniques to be used in the QROM and proved security of GPV-IBE in the QROM, hence answering in the affirmative that GPV-IBE is indeed post-quantum. However, since the general technique developed by Zhandry incurred a large reduction loss, there was a wide gap between the concrete efficiency and security level provided by GPV-IBE in the ROM and QROM. Furthermore, regardless of being in the ROM or QROM, GPV-IBE is not known to have a tight reduction in the multi-challenge setting. Considering that in the real-world an adversary can obtain many ciphertexts, it is desirable to have a security proof that does not degrade with the number of challenge ciphertext.

In Chapter 3, we provide a much tighter proof for the GPV-IBE in the QROM in the single-challenge setting. In addition, we also show that a slight variant of the GPV-IBE has an almost tight reduction in the multi-challenge setting both in the ROM and QROM, where the reduction loss is independent of the number of challenge ciphertext. Our proof departs from the traditional partitioning technique and resembles the approach used in the public-key encryption scheme of Cramer and Shoup [CS98]. Our proof strategy allows the reduction algorithm to program the random oracle the same way for all identities and naturally fits the QROM setting where an adversary may query a superposition of all identities in one random oracle query. Notably, our proofs are much simpler than the one by Zhandry and conceptually much easier to follow for cryptographers not familiar with quantum computation. Although at a high level, the techniques used for the single and multi-challenge setting are similar, the technical details are quite different. For the multi-challenge setting, we rely on the Katz-Wang technique [KW03] to overcome some obstacles regarding the leftover hash lemma.

1.3.2 Summary of Chapter 4

In Chapter 4, we present new adaptively secure identity-based encryption (IBE) schemes. One of the distinguishing properties of the schemes is that it achieves shorter public parameters than previous schemes. Both of our schemes follow the general framework presented in the recent IBE scheme of Yamada [Yam16], employed with novel techniques tailored to meet the underlying algebraic structure to overcome the difficulties arising in our specific setting. Specifically, we obtain the following:

- Our first scheme is proven secure under the ring learning with errors (RLWE) assumption and achieves the best asymptotic space efficiency among existing schemes from the same assumption. The main technical contribution is in our new security proof that exploits the ring structure in a crucial way. Our technique allows us to greatly weaken the underlying hardness assumption (e.g., we assume the hardness of RLWE with a fixed polynomial approximation factor whereas Yamada's scheme requires a super-polynomial approximation factor) while improving the overall efficiency.
- Our second IBE scheme is constructed on bilinear maps and is secure under the 3-computational bilinear Diffie-Hellman exponent assumption. This is the first IBE scheme based on the hardness of a computational/search problem, rather than a decisional problem such as DDH and DLIN on bilinear maps with sub-linear public parameter size.

1.3.3 Summary of Chapter 5

Predicates are used in cryptography as a fundamental tool to control the disclosure of secrets. However, how to embed a particular predicate into a cryptographic primitive is usually not given much attention. In Chapter 5, we formalize the idea of encoding predicates as arithmetic circuits and observe that choosing the right encoding of a predicate may lead to an improvement in many aspects such as the efficiency of a scheme or the required hardness assumption. In particular, we develop two predicate encoding schemes with different properties and construct cryptographic primitives that benefit from these: verifiable random functions (VRFs) and predicate encryption (PE) schemes.

- We propose two VRFs on bilinear maps. Both of our schemes are secure under a non-interactive q-type assumption where q is only poly-logarithmic in the security parameter, and they achieve either a poly-logarithmic verification key size or proof size. This is a significant improvement over prior works, where all previous schemes either require a strong hardness assumption or a large verification key and proof size.

- We propose a lattice-based PE scheme for the class of *multi-dimensional equality* ($\mathsf{MultD\text{-}Eq}$) predicates. This class of predicate is expressive enough to capture many of the appealing applications that motivates PE schemes. Our scheme achieves the best in terms of the required approximation factor for LWE (we only require $\mathsf{poly}(\lambda)$) and the decryption time. In particular, all existing PE schemes that support the class of $\mathsf{MultD\text{-}Eq}$ predicates either require a subexponential LWE assumption or an exponential decryption time (in the dimension of the $\mathsf{MultD\text{-}Eq}$ predicates).

1.3.4 Summary of Chapter 6

In non-zero inner product encryption (NIPE) schemes, ciphertexts and secret keys are associated with vectors and decryption is possible whenever the inner product of these vectors does *not* equal zero. So far, much effort on constructing bilinear map-based NIPE schemes have been made and this has lead to many efficient schemes. However, the constructions of NIPE schemes without bilinear maps are much less investigated. The only known other NIPE constructions are based on lattices, however, they are all highly inefficient due to the need of converting inner product operations into circuits or branching programs.

To remedy our rather poor understanding regarding NIPE schemes without bilinear maps, we provide two methods for constructing NIPE schemes: a direct construction from lattices and a generic construction from functional encryption schemes for inner products (LinFE). For our first direct construction, it highly departs from the traditional lattice-based constructions and we rely heavily on new tools concerning Gaussian measures over *multi-dimensional lattices* to prove security. For our second generic construction, using the recent constructions of LinFE schemes as building blocks, we obtain the first NIPE constructions based on the DDH and DCR assumptions. In particular, we obtain the first NIPE schemes *without* bilinear maps or lattices.

1.3.5 Summary of Chapter 7

Attribute-based signature (ABS), originally introduced by Maji et al. [MPR11], represents an essential mechanism to allow for fine-grained authentication. A user associated with an attribute x can sign w.r.t. a given public policy C only if his attribute satisfies C, i.e., $C(x) = 1$. So far, much effort on constructing bilinear map-based ABS schemes have been made, where the state-of-the-art scheme of Sakai et al. [SAH16] supports the very wide class of *unbounded* circuits as policies. However, construction of ABS schemes without bilinear maps are less investigated, where it was not until recently that Tsabary [Tsa17] showed a lattice-based ABS scheme supporting *bounded* circuits as policies, at the cost of weakening the security requirement.

In Chapter 7, we affirmatively close the gap between ABS schemes based on bilinear maps and lattices by constructing the first lattice-based ABS scheme for *unbounded circuits* in the random oracle model. We start our work by providing a generic construction of ABS schemes for unbounded-circuits in the random oracle model, which in turn implies that one-way functions are sufficient to construct ABS schemes. To prove security, we formalize and prove a generalization of the Forking Lemma, which we call *"general multi-forking lemma with oracle access"*, capturing the situation where the simulator is interacting with some algorithms he *cannot* rewind, and also covering many features of the recent lattice-based ZKPs. This, in fact, was a formalization lacking in many existing anonymous signatures from lattices so far (e.g., group signatures). Therefore, this formalization is believed to be of independent interest. Finally, we provide a concrete instantiation of our generic ABS construction from lattices by introducing a new Σ-protocol, that highly departs from the previously known techniques, for proving possession of a valid signature of the lattice-based signature scheme of Boyen [Boy10].

Chapter 2

Preliminary

In this chapter, we prepare notations and tools that will be used throughout the thesis.

2.1 Notation

In this section, we first provide the notations that will be used. Whenever the meaning of any symbols become unclear when reading through the thesis, please use this section as reference.

We use non-italic bold lowercase letters (e.g., $\mathbf{v}$) for vectors with entries in $\mathbb{R}$ and italic bold lowercase letters (e.g., $\boldsymbol{v}$) for vectors with entries in rings or number fields. Unless stated otherwise we typically view vectors in their row form, however, in some special cases, e.g., Chapter 4, we view them as column vectors for notational convenience. Matrices are denoted by uppercase bold letters analogously. For a vector $\mathbf{v} \in \mathbb{R}^n$, denote $\|\mathbf{v}\|_p$ as the L_p-norm, where $p = 2$ is the standard Euclidean norm. For a matrix $\mathbf{R} \in \mathbb{R}^{n\times n}$, denote $\|\mathbf{R}\|_{\mathsf{GS}}$ as the longest column of the Gram-Schmidt orthogonalization of $\mathbf{R}$ and denote $s_1(\mathbf{R})$ as the largest singular value (also known as the spectral norm). We occasionally drop the subscript of the norm when it is clear from context. We denote $[\cdot|\cdot]$ (resp. $[\cdot;\cdot]$) as the horizontal (resp. vertical) concatenation of vectors and matrices. Denote $\mathbf{I}_m$ as the $m \times m$ identity matrix and $\mathbf{0}_{n\times m}$ as the $n \times m$ zero matrix. We occasionally view elements in $\mathbb{Z}_p$ as elements in $\mathbb{Z}$ by its obvious embedding.

We use $\{\cdot\}$ to denote sets and use $(\cdot)$ to denote a finite ordered list of elements. When we use notations such as $(w_{i,j})_{(i,j)\in[n]\times[m]}$ for $n, m \in \mathbb{N}$, we assume the elements are sorted in the lexicographical order. For $n, m \in \mathbb{N}$ with $n \leq m$, denote $[n]$ as the set $\{1, \cdots, n\}$ and $[n, m]$ as the set $\{n, \cdots, m-1, m\}$. For a (quotient) polynomial ring R over $\mathbb{Z}$, we denote $[-n, n]_R \subseteq R$ as the set of elements in R with all coefficients in the interval $[-n, n]$. For a finite set S, we let $U(S)$ denote the uniform distribution over S. For a distribution D and integer $k > 0$, define $(D)^k$ as the distribution $\prod_{i\in[k]} D$. For a distribution or random variable X we write $x \leftarrow X$ to denote the operation of sampling a random x according to X. For a set S, we write $s \leftarrow S$ as a shorthand for $s \leftarrow U(S)$. Let X and Y be two random variables over some finite set S_X, S_Y, respectively. The *statistical distance* $\Delta(X, Y)$ between X and Y is defined as $\Delta(X, Y) = \frac{1}{2}\Sigma_{s\in S_X\cup S_Y} |\Pr[X = s] - \Pr[Y = s]|$. The *min-entropy* of a random variable X is defined as $\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X = x])$, where the base of the logarithm is taken to be 2 throughout the thesis. For a bit $b \in \{0, 1\}$, $\bar{b}$ denotes $1-b$. For sets $\mathcal{X}$ and $\mathcal{Y}$, $\mathsf{Func}(\mathcal{X}, \mathcal{Y})$ denotes the set of all functions from $\mathcal{X}$ to $\mathcal{Y}$. A function $f : \mathbb{N} \to \mathbb{R}_{\geq 0}$ is said to be *negligible*, if for all c, there exists λ_0 such that $f(\lambda) < 1/\lambda^c$ for all $\lambda > \lambda_0$. We denote by $\mathsf{negl}(\lambda)$ a negligible function in λ. We also denote by $\mathsf{poly}(\lambda)$ as a polynomial function in λ. We say that the two distributions are *statistically close* or *negligibly close* when the statistical distance is negligible. Furthermore, *overwhelming* probability is used in case the probability is negligibly close to 1. Finally, we use the short hand PPT for probabilistic polynomial time algorithm.

2.2 Lattices and Gaussian Distributions

Lattices are one of the central tools we use through the thesis. Depending on the security notion or efficiency one aims for, there are several types of lattices one can use. In this section we introduce the most standard notion of lattices. Later on, we introduce the notion of ideal lattices. We should note that ideal lattices are by now also considered a "standard" notion due to the much effort brought into the research of lattices this past decade. However, due to its rather complex algebraic nature, we believe providing details of standard (non-ideal) lattices in a separate section ease the presentation.

2.2.1 Lattices

An n-dimensional (full rank) lattice $\Lambda \subseteq \mathbb{R}^n$ is the set of all integer linear combinations of some set of n linearly independent basis vectors $\mathbf{B} = \{\mathbf{b}_1, \dots, \mathbf{b}_n\} \subseteq \mathbb{R}^n$, $\Lambda = \{\sum_{i \in [n]} z_i \mathbf{b}_i | \mathbf{z} \in \mathbb{Z}^n\}$. For positive integers q, n, m, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{u} \in \mathbb{Z}_q^n$, the m-dimensional "shifted" integer lattice is defined as $\Lambda_{\mathbf{u}}^{\perp}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m | \mathbf{A}\mathbf{z}^T = \mathbf{u}^T \mod q\}$. We simply write $\Lambda^{\perp}(\mathbf{A})$ in case $\mathbf{u} = \mathbf{0}$.

Gaussian Measures over Lattices. For $\sigma > 0$ and $\mathbf{c} \in \mathbb{R}^n$, the n-dimensional Gaussian function $\rho_{\sigma,\mathbf{c}} : \mathbb{R}^n \to (0, 1]$ is defined as $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi\|\mathbf{x} - \mathbf{c}\|_2^2/\sigma^2)$. We omit the subscript $\mathbf{c}$ when it is taken to be $\mathbf{0}$. The (spherical) continuous Gaussian distribution D_σ over $\mathbb{R}^n$ is the distribution with density function proportional to ρ_σ. When the dimension n is not clear from context, we explicitly write it as D_s^n. More generally, for any matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$, denote $D_{\mathbf{B}}$ as the distribution of $\mathbf{x}\mathbf{B}^T$ where $\mathbf{x}$ is distributed as D_1^m. A well known fact is that for any two matrices $\mathbf{B}_1, \mathbf{B}_2$, the sum of an independent sample from $D_{\mathbf{B}_1}$ and $D_{\mathbf{B}_2}$ is distributed as $D_{\mathbf{C}}$ where $\mathbf{C} = (\mathbf{B}_1\mathbf{B}_1^T + \mathbf{B}_2\mathbf{B}_2^T)^{1/2}$.

For an n-dimensional lattice Λ, the discrete Gaussian distribution over Λ with center $\mathbf{c}$ and parameter σ is defined as $D_{\Lambda,\sigma,\mathbf{c}}(\mathbf{x}) = \rho_{\sigma,\mathbf{c}}(\mathbf{x})/\rho_{\sigma,\mathbf{c}}(\Lambda)$ for all $\mathbf{x} \in \Lambda$, where $\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{\mathbf{x} \in \Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$. Furthermore, for an n-dimensional shifted lattice $\Lambda + \mathbf{t}$, we define the Gaussian distribution $D_{\Lambda+\mathbf{t},\sigma}$ with center $\mathbf{c} = \mathbf{0}$ and parameter σ as the process of adding the vector $\mathbf{t}$ to a sample from $D_{\Lambda,\sigma,-\mathbf{t}}$. We omit the subscripts σ and $\mathbf{c}$ when they are taken to be 1 and $\mathbf{0}$, respectively. We also extend the above definition to polynomial rings as well. Specifically, we define the discrete Gaussian distribution $D_{\Lambda+\mathbf{u},r}^{\mathsf{coeff}}$ over a (quotient) polynomial ring R in X over $\mathbb{R}$. The discrete Gaussian distribution $D_{\Lambda+\mathbf{u},r}^{\mathsf{coeff}}$ is the distribution of $a = \sum_{i=0}^{n-1} \alpha_i X^i \in R$ where the coefficient vector $[\alpha_0, \dots, \alpha_{n-1}] \in \mathbb{R}^n$ is sampled from the discrete Gaussian distribution $D_{\Lambda+\mathbf{u},r}$. This definition naturally extends to vectors $\boldsymbol{a} \in R^k$ in case of nk-dimensional lattices. Finally, we call D a B-bounded distribution, if all the elements in the support of D have absolute value smaller than B.

Below, we provide several useful lemmas we use to analyze the behavior of discrete Gaussian distributions. The first lemma provides us with useful tools that the reduction algorithm can use during the security proof. It may be first helpful to view the lemma in comparison with trapdoor hash functions.

Lemma 2.1 ([GPV08], Lem. 5.2, Cor. 5.4 and Adapted from [ALS16], Lem. 9)**.** *Let q be a prime* *or some power of a prime*[1] *p and let n, m be positive integers such that $m \geq 2n \log q$. Let σ be any positive real such that $\sigma \geq \omega(\sqrt{\log n})$. Then for $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ and $\mathbf{e} \leftarrow D_{\mathbb{Z}^m,\sigma}$, the distribution of $\mathbf{u} = \mathbf{A}\mathbf{e} \mod q$ is statistically close to uniform over $\mathbb{Z}_q^n$.*

Furthermore, fix $\mathbf{u} \in \mathbb{Z}_q^n$ and let $\mathbf{t} \in \mathbb{Z}^m$ be an arbitrary solution to $\mathbf{A}\mathbf{t} = \mathbf{u} \mod q$. Then the conditional distribution of $\mathbf{e} \leftarrow D_{\mathbb{Z}^m,\sigma}$, given $\mathbf{A}\mathbf{e} = \mathbf{u} \mod q$ for a uniformly random $\mathbf{A}$ in $\mathbb{Z}_q^{n\times m}$ is exactly $D_{\Lambda^\perp(\mathbf{A})+\mathbf{t},\sigma}$ with all but negligible probability.

The following lemma allows us to bound the singular value of a matrix whose columns are sampled from a Gaussian distribution.

Lemma 2.2 ([MP12], Lem. 2.8 and Lem. 2.9)**.** *Let m, k be positive integers, $\{\sigma_i\}_{i=1}^k$ a set of positive reals and denote $\sigma_{max} = \max_i\{\sigma_i\}$. Let $\mathbf{R} \in \mathbb{Z}^{m\times k}$ be a matrix where its i-th column is sampled from $D_{\mathbb{Z}^m,\sigma_i}$. Then there exists a universal constant $C > 0$ such that we have $s_1(\mathbf{R}) \leq C \cdot \sigma_{max}(\sqrt{m} + \sqrt{k})$ with all but negligible probability in m.*

The following lemma shows us the importance of a basis with a small singular value. Specifically, it tells us how small a vector we can sample from the lattice $\Lambda^\perp(\mathbf{A})$.

Lemma 2.3 ([ABB10], Lem. 8)**.** *Let n, m, q be positive integers with $m > n$, $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be a matrix, $\mathbf{u} \in \mathbb{Z}_q^n$ be a vector, $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m\times m}$ be a basis for $\Lambda^\perp(\mathbf{A})$, and $\sigma > \|\mathbf{T}_\mathbf{A}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m})$. Then, if we sample a vector $\mathbf{x} \leftarrow D_{\Lambda^\perp_\mathbf{u}(\mathbf{A}),\sigma}$, we have $\Pr[\|\mathbf{x}\|_2 > \sqrt{m}\sigma] < \mathsf{negl}(n)$.*

In some cases, a version of the above lemma with a more concrete parameter settings is useful. The following lemma is obtained by combining Lem. 4.4 in [MR07] and Lem. 5.3 in [GPV08].

Lemma 2.4 ([MR07, GPV08])**.** *Let $\sigma > 16\sqrt{\log 2m/\pi}$ and $\mathbf{u}$ be any vector in $\mathbb{Z}_q^n$. Then, for all but q^{-n} fraction of $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, we have that*

$$\Pr_{\mathbf{x}\leftarrow D_{\Lambda^\perp_\mathbf{u},\sigma}(\mathbf{A})}[\|\mathbf{x}\|_2 > \sqrt{m}\sigma] < 2^{-(m-1)}.$$

The following lemma can be obtained by a straightforward combination of Lem. 2.6, Lem. 2.10, and Lem. 5.3 in [GPV08] (See also [PR06, Pei07]). It tells us that a vector sampled from the discrete Gaussian distribution has high min-entropy.

Lemma 2.5 ([PR06, Pei07, GPV08])**.** *Let $\sigma > 16\sqrt{\log 2m/\pi}$ and $\mathbf{u}$ be any vector in $\mathbb{Z}_q^n$. Then, for all but q^{-n} fraction of $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, we have*

$$\mathbf{H}_\infty(D_{\Lambda^\perp_\mathbf{u}(\mathbf{A}),\sigma}) \geq m - 1.$$

The following lemma is a part of the contribution of the work presented in Chapter 4. Since this lemma is also used in many of the other chapters, we believe it to be better to provide the details in the preliminaries. The following noise rerandomization lemma is typically used by the reduction algorithm during the security proof; the reduction algorithm runs algorithm ReRand to create a challenge ciphertext that is distributed statistically close to the real word. Although, there are techniques such as "noise flooding" (e.g., [GKPV10, DGK+10]) that achieves the same effect, the following lemma achieves the same effect with the minimal amount of noise addition and hence leads to a more efficient scheme.

[1]Note that for the case $q = p^k$ for some $k \in \mathbb{N}$, we set the statistical distance to be $n^{-\omega(1)}$ rather than $2^{-\Omega(n)}$ as in [ALS16], Lem. 9.

Lemma 2.6 (Noise Rerandomization, [KY16], Lem. 1)**.** *Let q, ℓ, m be positive integers and r a positive real satisfying $r > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log \ell})\}$. Let $\mathbf{b} \in \mathbb{Z}_q^m$ be arbitrary and $\mathbf{x}$ chosen from $D_{\mathbb{Z}^m,r}$. Then there exists a PPT algorithm* ReRand *that for any $\mathbf{V} \in \mathbb{Z}^{m\times\ell}$ and positive real $\sigma > s_1(\mathbf{V})$,* $\mathsf{ReRand}(\mathbf{V}, \mathbf{b}+\mathbf{x}, r, \sigma)$ *outputs $\mathbf{b}'^{\top} = \mathbf{b}^{\top}\mathbf{V} + \mathbf{x}'^{\top} \in \mathbb{Z}_q^{\ell}$ where $\mathbf{x}'$ is distributed statistically close to $D_{\mathbb{Z}^{\ell},2r\sigma}$.*

Proof. Before beginning the main proof of Lemma 2.6 on noise rerandomization, we recall the following two lemmas. Note that Lemma 2.8, a special case of the claim from [Reg05], is restated in order to make the comparison between Lemma 2.7 more clear. Both lemma concerns convolution between discrete and continuous Gaussian distributions.

Lemma 2.7 ([Pei10], Special Case of Theorem 3.1)**.** *Let n be a positive integer and r be a positive real satisfying $r \geq \omega(\sqrt{\log n})$. Then, if we choose $\mathbf{x}_1$ from the continuous Gaussian D_r^n and then choose $\mathbf{x}_2$ from the discrete Gaussian $D_{\mathbb{Z}^n-\mathbf{x}_1,r}$, then $\mathbf{x}_1+\mathbf{x}_2$ is distributed statistically close to the discrete Gaussian $D_{\mathbb{Z}^n,\sqrt{2}r}$.*

Lemma 2.8 ([Reg05], Special Case of Claim 3.9)**.** *Let n be a positive integer and let r a positive real satisfying $r \geq \omega(\sqrt{\log n})$. Then, if we choose $\mathbf{x}_1$ from the continuous Gaussian D_r^n and choose $\mathbf{x}_2$ from the discrete Gaussian $D_{\mathbb{Z}^n,r}$, $\mathbf{x}_1+\mathbf{x}_2$ is distributed statistically close to the continuous Gaussian $D_{\sqrt{2}r}^n$.*

Now armed with the above lemmas, we begin the main proof below. The ReRand algorithm proceeds by sampling vectors $\mathbf{c}, \mathbf{d}$ and $\mathbf{f}$ as follows:

1. sample $\mathbf{c}$ from the continuous Gaussian distribution D_r^m,
2. sample $\mathbf{d}$ from the continuous Gaussian distribution $D_{\sqrt{2}r(\sigma^2\mathbf{I}_\ell-\mathbf{V}^T\mathbf{V})^{1/2}}$, and
3. sample $\mathbf{f}$ from the discrete Gaussian $D_{\mathbb{Z}^{\ell}-(\mathbf{c}\mathbf{V}+\mathbf{d}),\sqrt{2}r\sigma}$.

Observe the distribution of $\mathbf{d}$ is well-defined. The ReRand algorithm outputs the following vector:

$$\mathbf{b}' = ((\mathbf{b}+\mathbf{x})+\mathbf{c})\,\mathbf{V}+\mathbf{d}+\mathbf{f} = \mathbf{b}\mathbf{V} + \underbrace{(\mathbf{x}+\mathbf{c})\mathbf{V}+\mathbf{d}+\mathbf{f}}_{\mathbf{x}':=\text{"noise term"}} \in \mathbb{Z}_q^{\ell}.$$

We analyse the noise term and show that it is distributed as in the statement. Let $\mathbf{x}' = (\mathbf{x}+\mathbf{c})\mathbf{V}+\mathbf{d}+\mathbf{f}$. Observe that by Lemma 2.8, $\mathbf{x}+\mathbf{c}$ is distributed as the continuous Gaussian distribution $D_{\sqrt{2}r}^m$. Therefore, $(\mathbf{x}+\mathbf{c})\mathbf{V}$ is distributed as the distribution $D_{\sqrt{2}r\mathbf{V}^T}$. Since $\mathbf{d}$ is sampled from the continuous Gaussian distribution $D_{\sqrt{2}r(\sigma^2\mathbf{I}_\ell-\mathbf{V}^T\mathbf{V})^{1/2}}$, it follows that $\mathbf{y} = (\mathbf{x}+\mathbf{c})\mathbf{V}+\mathbf{d}$ is distributed as a spherical continuous Gaussian $D_{\sqrt{2}r\sigma}^{\ell}$. Next, observe that since $\mathbf{x}\mathbf{V} \in \mathbb{Z}^{\ell}$, the two distributions $D_{\mathbb{Z}^{\ell}-\mathbf{y},\sqrt{2}r\sigma}$ and $D_{\mathbb{Z}^{\ell}-(\mathbf{c}\mathbf{V}+\mathbf{d}),\sqrt{2}r\sigma}$ are equivalent by definition. Therefore, by Lemma 2.7, adding $\mathbf{f}$ chosen from the discrete Gaussian $D_{\mathbb{Z}^{\ell}-(\mathbf{c}\mathbf{V}+\mathbf{d}),\sqrt{2}r\sigma}$ to $\mathbf{y}$, which we can do without knowledge of the unknown value $\mathbf{x}$, we can discretize $\mathbf{y}$. Hence $\mathbf{x}' = \mathbf{y}+\mathbf{f}$ is distributed according to the discrete Gaussian $D_{\mathbb{Z}^{\ell},2r\sigma}$ as in the above statement. □

Random Matrices. The following lemmas state the properties of random matrices. They will be used to obtain a more precise analysis of our lattice-based scheme.

Lemma 2.9 ([LPRTJ05, ABB10])**.** *Let m, k be positive integers such that $k \geq m$. If $\mathbf{R}$ is sampled uniformly in $\{-1,1\}^{m\times k}$ then $s_1(\mathbf{R}) \leq 20\sqrt{m+k}$ with overwhelming probability in m.*

Lemma 2.10. *Let ℓ, n, k be positive integers and set $m = nk$, and let D be a B-bounded distribution. Let $\mathbf{R} \leftarrow D^{\ell \times m}$ and $\mathbf{U}$ be an arbitrary block diagonal matrix $\mathbf{U} = \mathrm{diag}(\mathbf{U}^{(1)}, \cdots \mathbf{U}^{(n)}) \in \{0,1\}^{m \times m}$ where $\mathbf{U}^{(w)} \in \{0,1\}^{k \times k}$ for $w \in [n]$. Then, there exists a universal constant $C > 0$ such that we have $s_1(\mathbf{RU}) \le C \cdot Bm\sqrt{k} = C \cdot Bnk^{3/2}$ with all but negligible probability in m.*

Proof. We first show $\mathbf{RU}$ is subgaussian with parameter $B\sqrt{mk}$. Note that we say that a random matrix $\mathbf{X}$ is *subgaussian* with parameter $\sigma > 0$ if all of its one-dimensional marginals $\mathbf{u}^\top \mathbf{X} \mathbf{v}$ for unit vectors $\mathbf{u}, \mathbf{v}$ are subgaussian with parameters σ, i.e., $\mathbb{E}[\exp(s \cdot \mathbf{u}^\top \mathbf{X} \mathbf{v})] \le \exp(\sigma^2 s^2/2)$. Observe that

$$\mathbf{u}^\top \mathbf{RUv} = \sum_{i=1}^{\ell} \sum_{t=1}^{m} \mathbf{R}_{i,t} \Big(u_i \sum_{j=1}^{m} \mathbf{U}_{t,j} v_j \Big),$$

where $\mathbf{R}_{i,t}$ is the (i,t)-th element of $\mathbf{R}$. (Other terms $\mathbf{U}_{t,j}, \mathbf{u}_i, \mathbf{v}_j$ are defined analogously.) Then, we have

$$\begin{aligned}
\mathbb{E}[\exp(s \cdot \mathbf{u}^\top \mathbf{RUv})] &= \mathbb{E}\Big[\exp\Big(\sum_{i=1}^{\ell} \sum_{t=1}^{m} \mathbf{R}_{i,t} \big(u_i \sum_{j=1}^{m} \mathbf{U}_{t,j} v_j\big)\Big)\Big] \\
&= \prod_{i=1}^{\ell} \prod_{t=1}^{m} \mathbb{E}\Big[\exp\Big(s\mathbf{R}_{i,t} \big(u_i \sum_{j=1}^{m} \mathbf{U}_{t,j} v_j\big)\Big)\Big] \\
&\le \prod_{i=1}^{\ell} \prod_{t=1}^{m} \exp\Big(B^2 s^2 \big(u_i \sum_{j=1}^{m} \mathbf{U}_{t,j} v_j\big)^2/2\Big) && (2.1) \\
&\le \prod_{i=1}^{\ell} \prod_{t=1}^{m} \exp\Big(B^2 s^2 u_i^2 \big(\sum_{j=1}^{m} \mathbf{U}_{t,j}^2\big)\big(\sum_{j=1}^{m} v_j^2\big)/2\Big) && (2.2) \\
&\le \exp(B^2 s^2 \sum_{i=1}^{\ell} \sum_{t=1}^{m} u_i^2 k/2) && (2.3) \\
&= \exp(B^2 s^2 mk/2),
\end{aligned}$$

where Eq. (2.1) follows from the fact that any B-bounded symmetric random variable X (i.e., $|X| \le B$) is a subgaussian with parameter B, Eq. (2.2) follows from the CauchySchwarz inequality, and Eq. (2.3) follows from the fact that $\mathbf{v}$ is a unit vector and that there are at most k ones in each row of $\mathbf{U}$. Hence, we have that $\mathbf{RU}$ is a subgaussian parameter with parameter $B\sqrt{mk}$. Finally, using the Lem. 2.9 of [MP12] (See [Ver11] for further details), we obtain the statement in the above lemma. ☐

Lemma 2.11 (Leftover Hash Lemma)**.** *Let $q > 2$ be a prime, m, n, k be positive integers such that $m > (n+1)\log q + \omega(\log n)$, $k = \mathsf{poly}(n)$ and let $\mathbf{R} \leftarrow \{-1,1\}^{m \times k}$. Let $\mathbf{A}$ and $\mathbf{B}$ be matrices chosen uniformly in $\mathbb{Z}_q^{n \times m}$ and $\mathbb{Z}_q^{n \times k}$ respectively. Then the distribution of $(\mathbf{A}, \mathbf{AR})$ is negligibly close in n to the distribution of $(\mathbf{A}, \mathbf{B})$.*

2.2.2 Algorithms for Sampling over Discrete Gaussian Distributions

Here, we introduce algorithms for sampling vectors according to a discrete Gaussian distribution. First, we introduce a special matrix named the *gadget matrix* $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ presented in [MP12].

Without loss of generality, we will always assume that $n|m$. Here, $\mathbf{G}$ is a full rank matrix such that the lattice $\Lambda^{\perp}(\mathbf{G})$ has a publicly known basis $\mathbf{T}_{\mathbf{G}}$ with $\|\mathbf{T}_{\mathbf{G}}\|_{\mathsf{GS}} \leq \sqrt{5}$. With an abuse of notation, we also define a deterministic polynomial time algorithm $\mathbf{G}^{-1}$ that given an input $\mathbf{U} \in \mathbb{Z}_q^{n \times m}$ outputs a matrix $\mathbf{V} \in \{0,1\}^{m \times m}$ such that $\mathbf{G}\mathbf{V} = \mathbf{U} \mod q$. In particular, for any $t \in \mathbb{Z}_q$, $\mathbf{G}^{-1}(t \cdot \mathbf{G})$ returns a block diagonal matrix with n square matrices with size m/n along its diagonals.

Using this notion of gadget matrix, we are now ready to describe the sampling algorithms. We note that the matrix $\mathbf{G}$ below does not necessary have to be the gadget matrix as long as the trapdoor for the lattice $\Lambda^{\perp}(\mathbf{G})$ is known. However, due to its simple structure, the gadget matrix is typically used as input to the algorithm.

Lemma 2.12. ([GPV08, ABB10, CHKP10, MP12, BLP$^+$13]) *Let $n, m, q > 0$ be integers with $m > 3n\lceil \log q \rceil$.*

- $\mathsf{TrapGen}(1^n, 1^m, q) \to (\mathbf{A}, \mathbf{T}_{\mathbf{A}})$*: There exists a randomized algorithm that outputs a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a full-rank matrix $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$, where $\mathbf{T}_{\mathbf{A}}$ is a basis for $\Lambda^{\perp}(\mathbf{A})$, $\mathbf{A}$ is statistically close to uniform and $\|\mathbf{T}_{\mathbf{A}}\|_{\mathsf{GS}} = O(\sqrt{n \log q})$.*
- $\mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}, \mathbf{u}, \mathbf{T}_{\mathbf{A}}, \sigma) \to \mathbf{e}$ *: There exists a randomized algorithm that, given matrices $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n \times m}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, a basis $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m \times m}$ for $\Lambda^{\perp}(\mathbf{A})$, and a Gaussian parameter $\sigma > \|\mathbf{T}_{\mathbf{A}}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^{2m}$ sampled from a distribution which is* $\mathsf{negl}(n)$*-close to $D_{\Lambda^{\perp}_{\mathbf{u}}([\mathbf{A}|\mathbf{B}]), \sigma}$.*
- $\mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}, t, \mathbf{u}, \mathbf{T}_{\mathbf{G}}, \sigma) \to \mathbf{e}$*: There exists a randomized algorithm that, given a full-rank matrix $\mathbf{A}, \mathbf{G} \in \mathbb{Z}_q^{n \times m}$, an invertible element $t \in \mathbb{Z}_q$, a matrix $\mathbf{R} \in \mathbb{Z}^{m \times m}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, a basis $\mathbf{T}_{\mathbf{G}}$ for $\Lambda^{\perp}(\mathbf{G})$, and a Gaussian parameter $\sigma > s_1(\mathbf{R}) \cdot \|\mathbf{T}_{\mathbf{G}}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^{2m}$ sampled from a distribution which is* $\mathsf{negl}(n)$*-close to $D_{\Lambda^{\perp}_{\mathbf{u}}([\mathbf{A}|\mathbf{A}\mathbf{R}+t\mathbf{G}]), \sigma}$.*
- $\mathsf{Sample}\mathbb{Z}(\sigma)$ *: a randomized algorithm that, given a Gaussian parameter $\sigma > 16(\sqrt{\log 2m/\pi})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^m$ sampled from a distribution $2^{-\Omega(n)}$-close to $D_{\mathbb{Z}^m, \sigma}$.*

2.2.3 Hardness Assumption over "Standard" Lattices

Learning With Errors. We define the Learning with Errors (LWE) problem first introduced by Regev [Reg05], and further define a variant of LWE called the First-is-Errorless LWE (FE.LWE) problem introduced by [BLP$^+$13]. Both problems are shown to be as hard as approximating the worst-case GapSVP problems. In particular, the FE.LWE problem is proven to be essentially as hard as the LWE problem.

Definition 2.1 (LWE and FE.LWE)**.** *For integers $n = n(\lambda), m = m(n), q = q(n) > 2$, an error distribution $\chi = \chi(n)$ over $\mathbb{Z}$, and a PPT algorithm $\mathcal{A}$, an advantage for the* learning with errors problem $\mathsf{LWE}_{n,m,q,\chi}$ *of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,m,q,\chi}} = \left| \Pr\left[\mathcal{A}\left(\{\mathbf{a}_i\}_{i=1}^m, \{\mathbf{a}_i^{\top}\mathbf{s} + x_i\}_{i=1}^m\right) = 1\right] - \Pr\left[\mathcal{A}\left(\{\mathbf{a}_i\}_{i=1}^m, \{v_i\}_{i=1}^m\right) = 1\right] \right|$$

where $\mathbf{a}_i \leftarrow \mathbb{Z}_q^n$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $x_i \leftarrow \chi$, $v_i \leftarrow \mathbb{Z}_q$ for each $i \in [m]$. We say that the LWE *assumption holds if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{LWE}_{n,m,q,\chi}}$ is negligible for all PPT $\mathcal{A}$.*

In addition, we define the first-is-errorless learning with errors problem $\mathsf{FE.LWE}_{n,m,q,\chi}$*, which is the LWE problem where the first sample is noise free, i.e., we have $x_1 = 0$ instead of $x_1 \leftarrow \chi$. The advantage for the* $\mathsf{FE.LWE}_{n,m,q,\chi}$ *problem of $\mathcal{A}$ is defined analogously to above.*

The next result shows that the $\mathsf{FE.LWE}$ problem is as hard as the LWE problem.

Theorem 2.1 (LWE to $\mathsf{FE.LWE}$. [BLP+13], Lem. 4.3)**.** *For any integer $n \geq 2, m, q \geq 1$, and error distribution χ over $\mathbb{Z}$, if there exists a PPT algorithm $\mathcal{A}$ that solves $\mathsf{FE.LWE}_{n,m,q,\chi}$ with advantage ϵ, then it can be converted into a PPT algorithm $\mathcal{B}$ that solves $\mathsf{LWE}_{n-1,m,q,\chi}$ with advantage at least $\epsilon \cdot (1 - \sum_p p^{-n})$, with the sum going over all prime factors of q.*

The (decisional) LWE problem with a prime modulus q was first shown to be as hard as approximating the worst-case GapSVP problem by [Reg05]. Several works [Pei09, ACPS09, MM11, MP12, BLP+13, PRS17] handling the case of non-prime modulus q have appeared in the literatures.

Short Integer Solution. We define the Short Integer Solution (SIS) problem introduced by [Ajt96].

Definition 2.2 (SIS)**.** *For integers $n = n(\lambda), m = m(n), q = q(n) > 2$, a positive real β and a PPT algorithm $\mathcal{A}$, an advantage for the* short integer solution *problem $\mathsf{SIS}_{n,m,q,\beta}$ of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{SIS}_{n,m,q,\beta}} = \Pr[\mathcal{A}(\mathbf{A}, \beta) \to \mathbf{x} : \mathbf{A}\mathbf{x} = \mathbf{0} \mod q \wedge \|\mathbf{x}\|_\infty \leq \beta \wedge \mathbf{x} \neq \mathbf{0}],$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$. We say that the SIS assumption holds if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{SIS}_{n,m,q,\beta}}$ is negligible for all PPT $\mathcal{A}$.

If $m, \beta = \mathsf{poly}(n)$ and $q > \sqrt{n}\beta$, then the $\mathsf{SIS}_{n,m,q,\beta}$ problem is at least as hard as SIVP_γ for some $\gamma = \beta \cdot \tilde{O}(\sqrt{nm})$. Further details on the parameters can be found in [Ajt96, Mic04, MR07, GPV08, MP13].

2.3 Rings and Ideal Lattices

In this section, we provide backgrounds on rings and ideal lattices. We try to provide a minimum exposition of rings and ideal lattices to keep it self-contained. We provide more details for the interested readers in Section 2.5.

2.3.1 Background of Rings

Preparation. Let n be a power of 2 and set $m = 2n$. Define the ring $R = \mathbb{Z}[X]/(\Phi_m(X))$, where $\Phi_m(X) = X^n + 1$ is the mth cyclotomic polynomial. For an integer q, denote R_q as $R/qR = \mathbb{Z}[X]/(q, \Phi_m(X))$. By viewing the elements in R as $n-1$ degree polynomials in $\mathbb{Z}[X]$, we can consider a natural coefficient embedding of R onto the integer lattice $\mathbb{Z}^n$. Namely, we define the coefficient embedding $\phi : R \to \mathbb{Z}^n$ that maps $a = \sum_{i=0}^{n-1} \alpha_i X^i \in R$ to $[\alpha_0, \alpha_1, \ldots, \alpha_{n-1}] \in \mathbb{Z}^n$. We extend the coefficient embedding naturally to vectors and matrices. On the other hand, we can also identify R as the subring of anti-circulant matrices in $\mathbb{Z}^{n \times n}$ by viewing each ring element $a \in R$ as a linear transformation $r \to a \cdot r$ of R. Concretely, we define the ring homomorphism $\mathrm{rot} : R \to \mathbb{Z}^{n \times n}$ that sends $a \in R$ to a matrix in $\mathbb{Z}^{n \times n}$ such that the i-th row is $\phi(a \cdot X^{i-1} \mod \Phi_m(X)) \in \mathbb{Z}^n$. Note that the first row of $\mathrm{rot}(a)$ is $\phi(a)$. Similarly to above, the definition of the map rot naturally extends to vectors and matrices. We provide some useful formulas on ring elements in the Section 2.5.1.

Norms in R. We define the Euclidean length for an element $a \in R$ and a vector $\boldsymbol{v} \in R^k$ by identifying R with $\mathbb{Z}^n$ through the coefficient embedding.[2] Therefore, when we say a vector $\boldsymbol{v}$ in R^k is "short", we mean that $\|\phi(\boldsymbol{v})\|_2$ is small. We also define the largest singular value of a matrix $\boldsymbol{R} \in R^{s\times t}$ by identifying the ring R with $\mathbb{Z}^{n\times n}$ through the map rot.[3] Namely, $s_1(\boldsymbol{R}) := \max_{\|\mathbf{z}\|_2=1}\|\mathbf{z}\cdot \mathrm{rot}(\boldsymbol{R})\|_2$. Note that this definition allows us to consider singular values of an element in R as well.

Properties for Elements in R. As with matrices with entries in $\mathbb{R}$, we have similar singular value bounds for matrices with elements in R. Namely, we can bound the singular value of a random matrix chosen from $[-b,b]_R^{s\times t}$. Recall that an element of $[-b,b]_R$ is an element in R with all of its coefficients in the interval $[-b,b]$.

Lemma 2.13 ([DM15], Special case of Fact 1)**.** *Let b be a positive integer and $\boldsymbol{R}$ be a $s\times t$ matrix chosen uniformly at random from $[-b,b]_R^{s\times t}$. Then, there exists a universal constant $C(\approx 1/\sqrt{2\pi})$ such that*

$$\Pr[s_1(\boldsymbol{R}) \geq C\cdot b\sqrt{n}\cdot(\sqrt{s}+\sqrt{t}+\omega(\sqrt{\log n}))] = \mathsf{negl}(n)$$

We note that similarly to matrices with entries in $\mathbb{R}$, we have $s_1(\boldsymbol{R}_1\boldsymbol{R}_2) \leq s_1(\boldsymbol{R}_1)s_1(\boldsymbol{R}_2)$ for all $\boldsymbol{R}_1, \boldsymbol{R}_2 \in R^{k\times k}$, which follows from the fact that rot is a ring homomorphism. Furthermore, it also holds when $\boldsymbol{R}_1$ is replaced by an element a in R.

Regularity Lemma. The former Lemma shows that there exists a quotient ring $R_q = R/(q, \Phi_m(X))$ that acts roughly as a field, or in other words, R_q has exponentially many invertible elements. The latter Lemma is a ring analogue of the standard lattice regularity lemma.

Lemma 2.14. *Let q be a prime such that $q \equiv 3 \mod 8$ and n be a power of 2. Then, $\Phi_{2n}(X) = X^n + 1$ splits as $X^n+1 \equiv t_1t_2 \mod q$ for two irreducible polynomials $t_1 = X^{n/2}+uX^{n/4}-1$ and $t_2 = X^{n/2}-uX^{n/4}-1$ in $\mathbb{Z}_q[X]$ where $u^2 \equiv -2 \mod q$. Furthermore, all $x \in R_q$ satisfying $\|\phi(x)\|_2 < \sqrt{q}$ are invertible, i.e., $x \in R_q^*$.*

Proof. The first part of the lemma is taken from Lemma 2.3 of [SSTX09]. Therefore, we only prove the latter part of the lemma, which is implicit in [SS11]. If $x \notin R_q^*$, $x \in \langle t_1\rangle$ or $x \in \langle t_2\rangle$ holds over R_q. We assume that the former holds without loss of generality. Then, $t \in \langle t_1, q\rangle$ holds over R. Thus, $\mathcal{N}(x) = \mathcal{N}(\langle x\rangle) \geq \mathcal{N}(\langle t_1, q\rangle) = q^{n/2}$, where $\mathcal{N}$ is the (field) norm. (See [SS11] for the definition.) Then, by using the additive geometric mean it can be seen that $\|\sigma(x)\|_2 = \sqrt{\sum_{i=1}^n|\sigma_i(x)|^2} \geq \sqrt{n}\cdot \sqrt[2n]{\prod_{i=1}^n|\sigma_i(x)|^2} = \sqrt{n}\cdot\sqrt[n]{\mathcal{N}(x)} \geq \sqrt{nq}$ holds. Since $\|\sigma(x)\|_2 = \sqrt{n}\|\phi(x)\|_2$, the statement follows. □

Lemma 2.15 (Regularity Lemma)**.** *Let n be a power of 2, q be a prime larger than $4n$ such that $q \equiv 3 \mod 8$, and ℓ, k', k, ρ be positive integers satisfying $\ell, k' \geq 1$, $k \geq 2$, $\rho < \frac{1}{2}\sqrt{q/n}$. Define the family of hash functions $\mathcal{H} = \{h_{\boldsymbol{A}}(\boldsymbol{x}) : [-\rho,\rho]_R^k \to R_q^{k'}\}$, where $h_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x}$ for $\boldsymbol{A} \in R_q^{k'\times k}$, $\boldsymbol{x} \in R_q^{k\times 1}$. Then, $\mathcal{H}$ is a universal hash family. Furthermore, for $\boldsymbol{A} \xleftarrow{\$} R_q^{k'\times k}$ and $\boldsymbol{X} \xleftarrow{\$} [-\rho,\rho]_R^{k\times\ell}$, we have*

$$\Delta((\boldsymbol{A}, \boldsymbol{A}\boldsymbol{X})\ ;\ (\boldsymbol{A}, U(R_q^{k'\times\ell}))) \leq \frac{\ell}{2}\cdot\sqrt{\left(\frac{q^{k'}}{(2\rho+1)^k}\right)^n}.$$

[2] We could have identified the Euclidean length by the *canonical* embedding as done in other works. However, for our special case where n is power of 2, the lengths are equivalent up to a factor of $\sqrt{n}$. (See Section 2.5.4 for further detail.)

[3] For the special case where n is a power of 2, $s_1(\boldsymbol{R})$ defined by the coefficient and canonical embeddings are both equivalent to the one defined by the map rot. (See Section 2.5.4 for further detail.)

Proof. We first show the former part of the lemma. Let $\boldsymbol{x}_1 \neq \boldsymbol{x}_2 \in R_q^{k\times 1}$ be arbitrary elements in $[-\rho,\rho]_R^k$ and set $\boldsymbol{z} = \boldsymbol{x}_1 - \boldsymbol{x}_2 \in R_q^{k\times 1}$. Then we have $\boldsymbol{z} \in [-2\rho, 2\rho]_R^k$. Assume for some $\boldsymbol{A} \in R_q^{k'\times k}$, we have $h_{\boldsymbol{A}}(\boldsymbol{x}_1) = h_{\boldsymbol{A}}(\boldsymbol{x}_2)$, i.e., $h_{\boldsymbol{A}}(\boldsymbol{z}) = 0$. Since, $\boldsymbol{x}_1 \neq \boldsymbol{x}_2$, there exists $j \in [k]$ such that the jth coefficient of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are different. Then, by Lemma 2.14, since $\|\phi(z_j)\|_2 \leq 2\rho\sqrt{n} < \sqrt{q}$, z_j must be invertible. Therefore, $\boldsymbol{a}_j = z_j^{-1}\sum_{i\neq j} z_i \boldsymbol{a}_i$ where $\boldsymbol{a}_i \in R_q^{k'\times 1}$ is the ith column of $\boldsymbol{A}$. The probability of a random $\boldsymbol{A} \in R_q^{k'\times k}$ satisfying this condition is exactly $1/q^{nk'} = 1/|R_q|^{k'}$. Hence $\mathcal{H}$ is universal. We then show the latter part of the lemma. We observe that the case of $\ell = 1$ follows from the leftover hash lemma since the min-entropy of $\boldsymbol{X}$ is $(1/(2\rho+1))^{kn}$ in this case. The case of $\ell \geq 2$ immediately follows from a standard hybrid argument. □

2.3.2 Trapdoors for Rings

Define the gadget matrix $\boldsymbol{g}_b = [1|b|\cdots|b^{k'-1}|\boldsymbol{0}] \in R_q^k$, where b is a positive integer and $k \geq k' = \lceil \log_b q \rceil$. When $k = k'$ and $b = 2$, this corresponds to the matrix representation of the gadget matrix $\mathbf{G} \in \mathbb{Z}_q^{n\times nk}$ often used in the literatures by properly rearranging the rows and columns of $\mathrm{rot}(\boldsymbol{g}_2)$. The following algorithms are simple modification of traditional lattice based algorithms (See Lemma 2.12), however, owing to the conversion to the ring setting and the fact that we view vectors in their row form, it may seem unclear at first. We provide some discussions concerning the $\mathsf{TrapGen}$ algorithm below.

Lemma 2.16. *Let n be a power of 2, q be a prime larger than $4n$ such that $q \equiv 3 \mod 8$, and b, ρ be a positive integer satisfying $\rho < \frac{1}{2}\sqrt{q/n}$. Furthermore, define $\log_1(\cdot) := \log_2(\cdot)$. Then, there exist polynomial time algorithms with the properties below:*

- $\mathsf{TrapGen}(1^n, 1^k, q, \rho) \to (\boldsymbol{a}, \boldsymbol{T_a})$ ([MP12], Lemma 5.3): *a randomized algorithm that, when $k \geq 2\log_\rho q$, outputs a vector $\boldsymbol{a} \in R_q^k$ and a matrix $\boldsymbol{T_a} \in R^{k\times k}$, where $\mathrm{rot}(\boldsymbol{a}^T)^T \in \mathbb{Z}_q^{n\times nk}$ is a full-rank matrix and $\mathrm{rot}(\boldsymbol{T_a}) \in \mathbb{Z}^{nk\times nk}$ is a basis for $\Lambda^\perp(\mathrm{rot}(\boldsymbol{a}^T)^T)$ such that $\boldsymbol{a}$ is $\mathsf{negl}(n)$-close to uniform and $\|\mathrm{rot}(\boldsymbol{T_a})\|_{\mathsf{GS}} = O(b\rho\cdot\sqrt{n\log_\rho q})$.*[4]

- $\mathsf{SampleLeft}(\boldsymbol{a}, \boldsymbol{b}, u, \boldsymbol{T_a}, \sigma) \to \boldsymbol{e}$ ([CHKP10]): *a randomized algorithm that, given vectors $\boldsymbol{a}, \boldsymbol{b} \in R_q^k$ where $\mathrm{rot}(\boldsymbol{a}^T)^T, \mathrm{rot}(\boldsymbol{b}^T)^T \in \mathbb{Z}_q^{n\times nk}$ are full-rank, an element $u \in R_q$, a matrix $\boldsymbol{T_a} \in R^{k\times k}$ such that $\mathrm{rot}(\boldsymbol{T_a}) \in \mathbb{Z}^{nk\times nk}$ is a basis for $\Lambda^\perp(\mathrm{rot}(\boldsymbol{a}^T)^T)$, and a Gaussian parameter $\sigma > \|\mathrm{rot}(\boldsymbol{T_a})\|_{\mathsf{GS}}\cdot\omega(\sqrt{\log nk})$, outputs a vector $\boldsymbol{e} \in R^{2k}$ sampled from a distribution which is $\mathsf{negl}(n)$-close to $D^{\mathsf{coeff}}_{\Lambda^\perp_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T|\mathrm{rot}(\boldsymbol{b}^T)^T]),\sigma}$, i.e., $[\boldsymbol{a}|\boldsymbol{b}]\boldsymbol{e}^T = u$ and $\phi(\boldsymbol{e}) \in \mathbb{Z}^{2nk}$ is distributed according to $D_{\Lambda^\perp_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T|\mathrm{rot}(\boldsymbol{b}^T)^T]),\sigma}$.*

- $\mathsf{SampleRight}(\boldsymbol{a}, \boldsymbol{g}_b, \boldsymbol{R}, y, u, \boldsymbol{T}_{\boldsymbol{g}_b}, \sigma) \to \boldsymbol{e}$ *where $\boldsymbol{b} = \boldsymbol{aR} + y\boldsymbol{g}_b$* ([ABB10]): *a randomized algorithm that, given vectors $\boldsymbol{a}, \boldsymbol{g}_b \in R_q^k$ such that $\mathrm{rot}(\boldsymbol{a}^T)^T, \mathrm{rot}(\boldsymbol{g}_b)$*[5] *$\in \mathbb{Z}_q^{n\times nk}$ are full-rank matrices, elements $y \in R_q^*, u \in R_q$, a matrix $\boldsymbol{R} \in R^{k\times k}$, a matrix $\boldsymbol{T}_{\boldsymbol{g}_b} \in R^{k\times k}$ such that $\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b}) \in \mathbb{Z}^{nk\times nk}$ is a basis for $\Lambda^\perp(\mathrm{rot}(\boldsymbol{g}_b))$, and a Gaussian parameter $\sigma > s_1(\boldsymbol{R})\cdot\|\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b})\|_{\mathsf{GS}}\cdot\omega(\sqrt{\log nk})$, outputs a vector $\boldsymbol{e} \in R^{2k}$ sampled from a distribution which is $\mathsf{negl}(n)$-close to $D^{\mathsf{coeff}}_{\Lambda^\perp_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T|\mathrm{rot}(\boldsymbol{b}^T)^T]),\sigma}$, i.e., $[\boldsymbol{a}|\boldsymbol{b}]\boldsymbol{e}^T = u$ and $\phi(\boldsymbol{e}) \in \mathbb{Z}^{2nk}$ is distributed according to $D_{\Lambda^\perp_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T|\mathrm{rot}(\boldsymbol{b}^T)^T]),\sigma}$.*

[4] We combine several lemmas from [MP12] and the regularity lemma (Lemma 2.15) to show correctness of $\mathsf{TrapGen}$. See below for further detail. Further, the unusual lattice $\Lambda^\perp(\mathrm{rot}(\boldsymbol{a}^T)^T)$ is used only to be consistent with the other algorithms. Namely, we could have instead defined the trapdoor for the lattice $\Lambda^\perp(\mathrm{rot}(\boldsymbol{a}))$.

[5] We have $\mathrm{rot}(\boldsymbol{g}_b^T)^T = \mathrm{rot}(\boldsymbol{g}_b)$ since all the entries of $\boldsymbol{g}_b$ are integers.

- ([MP12]:) *Let $k \geq \lceil \log_b q \rceil$. There exists a publicly known matrix $\boldsymbol{T}_{\boldsymbol{g}_b}$ such that $\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b}) \in \mathbb{Z}^{nk \times nk}$ is a basis for the lattice $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{g}_b))$ and $\|\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b})\|_{\mathsf{GS}} \leq \sqrt{b^2+1}$. Furthermore, there exists a deterministic polynomial time algorithm $\boldsymbol{g}_b^{-1}$ which takes input $\boldsymbol{u} \in R_q^k$ and outputs $\boldsymbol{R} = \boldsymbol{g}_b^{-1}(\boldsymbol{u})$ such that $\boldsymbol{R} \in [-b, b]_R^{k \times k}$ and $\boldsymbol{g}_b \boldsymbol{R} = \boldsymbol{u}$.*

Note that we abuse the notation $\boldsymbol{g}_b^{-1}$ by viewing it as a function rather than a vector. Namely, for any $\boldsymbol{u} \in R_q^k$ there are many choices for $\boldsymbol{R} \in R^{k \times k}$ such that $\boldsymbol{g}_b \boldsymbol{R} = \boldsymbol{u}$, and $\boldsymbol{g}_b^{-1}(\boldsymbol{u})$ is a function that deterministically outputs a particular short matrix from the possible candidates. Since we have $s_1(\boldsymbol{R}) \leq b \cdot nk$ for any $\boldsymbol{R} \in [-b, b]_R^{k \times k}$, $s_1(\boldsymbol{g}_b^{-1}(\boldsymbol{u})) \leq bnk$ holds for arbitrary $\boldsymbol{u} \in R_q^k$.

Although it may be straight-forward to check, since the correctness of $\mathsf{TrapGen}$ in Lemma 2.16 is not explicitly provided in [MP12], we include a succinct proof below for completeness.

Proof. The proof follows by combining several Lemmas from [MP12] and our Lemma 2.13 and Lemma 2.15. First for simplicity asssume k is even, i.e., $k = 2k'$ for some $k' \in \mathbb{N}$, and assume that $k' = \lceil \log_b q \rceil$ for some positive integer b. We first show that $\boldsymbol{a} = [\boldsymbol{a}' | \boldsymbol{g}_b - \boldsymbol{a}'\boldsymbol{R}]$ is distributed uniformly at random over R^k when $\boldsymbol{a}' \xleftarrow{\$} R^{k'}$ and $\boldsymbol{R} \xleftarrow{\$} [-\rho, \rho]_R^{k' \times k'}$. This follows from Lemma 2.15, since we have

$$\frac{k'}{2}\sqrt{\left(\frac{q}{(2\rho+1)^{k'}}\right)^n} \leq \frac{k'}{2}\left(\frac{q}{(2\rho)^{k'}}\right)^{\frac{n}{2}} \leq \frac{k'}{2}\left(\frac{1}{2^{k'}}\right)^{\frac{n}{2}} \leq \frac{k'}{2^{n+1}} = \mathsf{negl}(n),$$

when $1 < \rho < \frac{1}{2}\sqrt{q/n}$, $k' \geq \log_\rho q$ and k' is polynomial in n. Similar result holds for the case $\rho = 1$. Note that in the case of $\rho = 1$, we define $\log_1 q := \log_2 q$. Next, by the property of $\boldsymbol{g}_b$ there exists a publicly known basis $\boldsymbol{T}_{\boldsymbol{g}_b} \in R^{k' \times k'}$ such that $\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b})$ is a basis for $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{g}_b^T)^T)$ (or equivallently for $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{g}_b)))$ such that $\|\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{g}_b})\|_{\mathsf{GS}} \leq \sqrt{b^2+1}$. We also have $s_1(\boldsymbol{R}) \leq O(\rho \cdot \sqrt{nk'})$ with all but negligible probability from Lemma 2.13. Then using the fact that $\mathrm{rot}(\boldsymbol{R}^T)^T$ (resp. $\mathrm{rot}(\boldsymbol{R})$) is a $\boldsymbol{g}_b$-trapdoor for $\mathrm{rot}(\boldsymbol{a}^T)^T$ (resp. $\mathrm{rot}(\boldsymbol{a})$) and by combining the ring version of Theorem 4.1 and Lemma 5.3 from [MP12], we obtain a basis $\boldsymbol{T}_{\boldsymbol{a}}$ such that $\|\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{a}})\|_{\mathsf{GS}} = O(b\rho \cdot \sqrt{n \log_\rho q})$. Note that we obtain bases for both $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{a}^T)^T)$ and $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{a}))$ from $\boldsymbol{T}_{\boldsymbol{a}}$ by properly rearanging $\boldsymbol{T}_{\boldsymbol{a}}$. □

2.3.3 Hardness Assumption over Ideal lattices

The ring LWE problem was introduced by Lyubashevsky et al. [LPR10]. They showed that solving it on the average is as hard as (quantumly) solving several standard problems on ideal lattices in the worst case.

Definition 2.3 (RLWE). *For positive integers $n = n(\lambda)$, $k = k(n)$, a prime integer $q = q(n) > 2$, an error distribution $\chi = \chi(n)$ over R_q, and an PPT algorithm $\mathcal{A}$, an advantage for the RLWE problem $\mathsf{RLWE}_{n,k,q,\chi}$ of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{RLWE}_{n,k,q,\chi}} = |\Pr[\mathcal{A}(\{(a_i, v_i)\}_{i=1}^k) \to 1] - \Pr[\mathcal{A}(\{(a_i, a_i s + e_i)\}_{i=1}^k) \to 1]|$$

where $a_1, \ldots, a_k, v_1, \ldots, v_k, s \xleftarrow{\$} R_q$ and $e_1, \ldots, e_k \xleftarrow{\$} \chi$. We say that $\mathsf{RLWE}_{n,k,q,\chi}$ assumption holds if $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{RLWE}_{n,k,q,\chi}}$ is negligible for all PPT $\mathcal{A}$.

Theorem 2.2. *Let α be a positive real, m be a power of 2, ℓ be an integer, $\Phi_m(X) = X^n + 1$ be the mth cyclotomic polynomial where $m = 2n$, and $R = \mathbb{Z}[X]/(\Phi_m(X))$. Let $q \equiv 3 \mod 8$ be a (polynomial size) prime such that there is another prime $p \equiv 1 \mod m$ satisfying $p \leq q \leq 2p$. Let also $\alpha q \geq n^{3/2}k^{1/4}\omega(\log^{9/4} n)$. Then, there is a probabilistic polynomial-time quantum reduction from $\tilde{O}(\sqrt{n}/\alpha)$-approximate SIVP (or SVP) to $\mathsf{RLWE}_{n,k,q,\chi}$ with $\chi = D_{\mathbb{Z}^n,\alpha q}^{\mathsf{coeff}}$.*

Although the proof is obtained by combining many of the previous results, since we were not able to find a proof compiled in a single paper, we include the proof in Section 2.5.2 for completeness. Due to the Linnik's theorem and Dirichlet's theorem on arithmetic progressions, we have that there are sufficiently many primes p and q satisfying the assumption of the theorem.

2.4 Cryptographic Primitives

In this section we provide a minimum exposition of cryptographic primitives. We intentionally exclude the explanation of cryptographic primitives that only appears in limited chapters for readability of the thesis.

2.4.1 Pseudorandom Functions

A pseudorandom function family is a pair of PPT algorithms $\mathsf{PRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Eval})$, such that the key generation algorithm $\mathsf{PRF.Gen}(1^\lambda)$ takes as input the security parameter, and outputs a seed $r \in \{0,1\}^\lambda$. The evaluation algorithm $\mathsf{PRF.Eval}(r, \mathbf{x})$ takes a seed $r \in \{0,1\}^\lambda$ and input $\mathbf{x} \in \{0,1\}^m$ and returns a bit string $y \in \{0,1\}^\eta$, where $m = m_\lambda$ is the input length and $\eta = \eta_\lambda$ is the output length. The security notion of an PRF is defined below:

Definition 2.4 (PRF)**.** *We say that a pair of PPT algorithms* $\mathsf{PRF} = (\mathsf{PRF.Gen}, \mathsf{PRF.Eval})$ *is a pseudorandom function if for all PPT adversaries* $\mathcal{A}$*, the advantage defined below is negligible:*

$$\mathsf{Adv}^{\mathsf{PRF}}_{\mathcal{A}}(\lambda) := \left| \Pr\left[\mathcal{A}^{\mathsf{RF}}(1^\lambda) = 1\right] - \Pr\left[\mathcal{A}^{\mathsf{PRF.Gen}(r,\cdot)}(1^\lambda) = 1\right] \right|$$

where $\mathsf{RF} \leftarrow \mathsf{Func}(\{0,1\}^m, \{0,1\}^\eta)$ *and* $r \leftarrow \mathsf{PRF.Gen}(1^\lambda)$*.*

2.4.2 Identity-based Encryption

Syntax. We use the standard syntax of IBE [BF01]. Let $\mathcal{ID}$ be the ID space of the scheme. If a collision resistant hash function $CRH : \{0,1\}^* \to \mathcal{ID}$ is available, one can use an arbitrary string as an identity. An IBE scheme is defined by the following four algorithms.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input a security parameter 1^λ and outputs a master public key mpk and a master secret key msk.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID}) \to \mathsf{sk}_{\mathsf{ID}}$: The key generation algorithm takes as input the master public key mpk, the master secret key msk, and an identity $\mathsf{ID} \in \mathcal{ID}$. It outputs a private key $\mathsf{sk}_{\mathsf{ID}}$. We assume that ID is implicitly included in $\mathsf{sk}_{\mathsf{ID}}$.

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M}) \to C$: The encryption algorithm takes as input a master public key mpk, an identity $\mathsf{ID} \in \mathcal{ID}$, and a message M. It outputs a ciphertext C.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C) \to \mathsf{M}$ or $\perp$: The decryption algorithm takes as input the master public key mpk, a private key $\mathsf{sk}_{\mathsf{ID}}$, and a ciphertext C. It outputs the message M or $\perp$, which means that the ciphertext is not in a valid form.

Correctness. We require correctness of decryption: that is, for all λ, all $\mathsf{ID} \in \mathcal{ID}$, and all M in the specified message space,

$$\Pr[\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})) = \mathsf{M}] = 1 - \mathsf{negl}(\lambda)$$

holds, where the probability is taken over the randomness used in $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$, $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$, and $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$.

Security. We now define the security for an IBE scheme Π. This security notion is defined by the following game between a challenger and an adversary $\mathcal{A}$. Let $\mathsf{CTSam}(\cdot)$ be a sampling algorithm that takes as input a master public key of the scheme and outputs an element in the ciphertext space.

- Setup. At the outset of the game, the challenger runs $\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$ and gives mpk to $\mathcal{A}$. The challenger also picks a random coin $\mathsf{coin} \leftarrow \{0,1\}$ and keeps it secretly. After given mpk, $\mathcal{A}$ can adaptively make the following two types of queries to the challenger. These queries can be made in any order and arbitrarily many times.

Secret Key Queries. If $\mathcal{A}$ submits $\mathsf{ID} \in \mathcal{ID}$ to the challenger, the challenger returns $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$.

Challenge Queries. If $\mathcal{A}$ submits a message M^* and an identity $\mathsf{ID}^* \in \mathcal{ID}$ to the challenger, the challenger proceeds as follows. If $\mathsf{coin} = 0$, it runs $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^*, \mathsf{M}^*) \to C^*$ and gives the challenge ciphertext C^* to $\mathcal{A}$. If $\mathsf{coin} = 1$, it chooses the challenge ciphertext C^* from the distribution $\mathsf{CTSam}(\mathsf{mpk})$ as $C^* \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ at random and gives it to $\mathcal{A}$.

We prohibit $\mathcal{A}$ from making a challenge query for an identity ID^* such that it has already made a secret key query for the same $\mathsf{ID} = \mathsf{ID}^*$ and vice versa.

- Guess. Finally, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. The advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\Pi}(\lambda) = \left| \Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2} \right|.$$

We say that Π is adaptively-anonymous secure, if there exists efficiently sampleable distribution $\mathsf{CTSam}(\mathsf{mpk})$ and the advantage of any PPT $\mathcal{A}$ is negligible in the above game. The term anonymous captures the fact that the ciphertext does not reveal the identity for which it was sent to. (Observe that $\mathsf{CTSam}(\mathsf{mpk})$ depends on neither of C^* nor M^*.)

Security without Anonymity. We also define the standard adaptive security (without anonymity) as in [Wat05] for Π via a similar game to the above. To define adaptive security, we change the challenge phase as follows.

- Challenge Phase. $\mathcal{A}$ outputs two messages M_0, M_1 and an identity $\mathsf{ID}^\star \in \mathcal{ID}$, on which it wishes to be challenged. Then, the challenger picks a random coin $\mathsf{coin} \xleftarrow{\$} \{0,1\}$, runs $\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^\star, \mathsf{M}_{\mathsf{coin}}) \to C^\star$, and gives the challenge ciphertext $C^\star$ to $\mathcal{A}$.

We say that Π is adaptively secure, if the advantage of any PPT $\mathcal{A}$ is negligible. We note that adaptively-anonymous security implies adaptive security. Namely, the former is a stronger security notion.

Single Challenge Security. We can also consider a variant of the above security definition where we restrict the adversary to make the challenge query only once during the game. We call this security notion "single challenge adaptive anonymity", and call the notion without the restriction "multi challenge security". By a simple hybrid argument, we can show that these definitions are in fact equivalent in the sense that one implies another. However, the proof that the former implies the latter incurs a huge security reduction loss that is linear in the number of challenge queries. Therefore, in some cases when we want to focus on tight security reductions, we typically differentiate these two notions.

2.5 Supplementary Materials for Rings and Ideal Lattices

In this section, we provide supplementary materials for rings and ideal lattices, which we believe to be useful for understanding them in more depth. However, since the thesis is readable without these details, this section may be safely skipped on first read.

2.5.1 Supplementary Note on Ring Elements

Useful Formulas. In hope of making the thesis more accessible, we provide some formulas on ring elements when viewed as vectors/matrices over $\mathbb{Z}$. Let R denote the polynomial ring $\mathbb{Z}[X]/(\Phi_m(X))$ for m a power of 2 and recall that we can view elements of R as $\mathbb{Z}^n$ through the coefficient embedding $\phi(\cdot)$ and as the subring of anti-circulant marices in $\mathbb{Z}^{n\times n}$ through the ring homomorphism $\mathrm{rot}(\cdot)$. In addition, vectors are viewed in their row forms. All of the following statement holds when we view the polynomial ring $R_q = \mathbb{Z}[X]/(q, \Phi_m(X))$ as $\mathbb{Z}_q$.

First of all, for any element $s \in R$, vectors $\boldsymbol{a}, \boldsymbol{e} \in R^k$ and matrix $\boldsymbol{R} \in R^{k\times\ell}$ recall that we have the following:

$$\begin{array}{lll} \phi(s) \in \mathbb{Z}^n, & \phi(\boldsymbol{a}) \in \mathbb{Z}^{nk}, & \\ \mathrm{rot}(s) \in \mathbb{Z}^{n\times n}, & \mathrm{rot}(\boldsymbol{a}) \in \mathbb{Z}^{n\times nk}, & \mathrm{rot}(\boldsymbol{R}) \in \mathbb{Z}^{nk\times n\ell}. \end{array}$$

Then, we obtain the following formulas through simple calculation:

1. $\phi(s\boldsymbol{a}) = \phi(s)\mathrm{rot}(\boldsymbol{a}) \in \mathbb{Z}^{nk}$
2. $\phi(\boldsymbol{a}\boldsymbol{e}^T) = \phi(\boldsymbol{a})\mathrm{rot}(\boldsymbol{e}^T) \in \mathbb{Z}^{n}$
3. $\phi(\boldsymbol{a}\boldsymbol{R}) = \phi(\boldsymbol{a})\mathrm{rot}(\boldsymbol{R}) \in \mathbb{Z}^{n\ell}$
4. $\mathrm{rot}(\boldsymbol{a}\boldsymbol{R}) = \mathrm{rot}(\boldsymbol{a})\mathrm{rot}(\boldsymbol{R}) \in \mathbb{Z}^{n\times n\ell}$

Gaussian Sampling. The second formula above is mainly used to bridge the gap between the Gaussian sampling algorithms for normal lattices and for ideal lattices (see Sec. 2.3 Lem. 2.16). Suppose we wish to sample a short vector $\boldsymbol{e} \in R^k$ (from a certain distribution we discuss later) such that $\boldsymbol{a}\boldsymbol{e}^T = u$, where $\boldsymbol{a} \in R^k$ and $u \in R$. Note that this comes up during the KeyGen procedure in our lattice-based construction. Applying the second formula in slightly a different order, we obtain the following:

$$\begin{aligned} & \phi(u) = \phi(\boldsymbol{a}\boldsymbol{e}^T) = \phi(\boldsymbol{e})\mathrm{rot}(\boldsymbol{a}^T) = \left(\mathrm{rot}(\boldsymbol{a}^T)^T\phi(\boldsymbol{e})^T\right)^T \\ \Leftrightarrow \quad & \mathrm{rot}(\boldsymbol{a}^T)^T\phi(\boldsymbol{e})^T = \phi(u)^T \in \mathbb{Z}_q^n. \end{aligned}$$

Note that in general $\mathrm{rot}(\boldsymbol{a}) \neq \mathrm{rot}(\boldsymbol{a}^T)^T$. Therefore, we only have to sample a vector $\mathbf{e} \in \mathbb{Z}^{nk}$ from the coset $\Lambda^{\perp}_{\phi(\boldsymbol{u})}(\mathrm{rot}(\boldsymbol{a}^T)^T)$ and map it back to its ring representation $\boldsymbol{e} = \phi^{-1}(\mathbf{e}) \in R^k$ to obtain a short sample $\boldsymbol{e}$ such that $\boldsymbol{a}\boldsymbol{e}^T = u$. This can be done easily by using a basis $\mathrm{rot}(\boldsymbol{T}_{\boldsymbol{a}})$ for the lattice $\Lambda^{\perp}(\mathrm{rot}(\boldsymbol{a}^T)^T)$.

2.5.2 Proof of Theorem 2.2

Here, we prove Theorem 2.2. Note that the proof is obtained by the straightforward combination of previous results (in particular, those of [LPR10] and [LS15]). However, to the best of our knowledge, there are no papers explicitly proving the theorem. This section is included for the purpose of completeness.

2.5.3 Gaussians over Ideal Lattices

We give a brief overview of Gaussians over ideal lattices and introduce the notations we will be using. We refer the general definitions of rings and ideal lattices to the works of [LPR10, LPR13]. In what follows, ζ_m is the primitive mth root of unity for $m > 2$, $\Phi_m(X)$ is the mth cyclotomic polynomial, $K = \mathbb{Q}(\zeta_m)$ is the mth cyclotomic number field of degree $n = \varphi(m)$, $R = \mathbb{Z}[\zeta_m] \cong \mathbb{Z}[X]/(\Phi_m(X))$ is the ring of integers of K[6] , $R^\vee \subseteq K$ is the dual ring and $K_\mathbb{R} = K \otimes_\mathbb{Q} \mathbb{R}$ is the field tensor product. Furthermore, the number field K has exactly n ring embeddings $\sigma_i : K \to \mathbb{C}$ that maps ζ_m to each of the complex roots of the cyclotomic polynomial $\Phi_m(X)$. The canonical embedding $\sigma : K \to \mathbb{C}^n$ is then defined as $\sigma(a) \to (\sigma_i(a))_{i\in\mathbb{Z}_m^*}$.

The Space H. Recall that when working with K (or $K_\mathbb{R}$) under the canonical embedding σ, it is convenient to use the following subspace $H \subseteq \mathbb{C}^n$,

$$H = \{(x_j)_{j\in\mathbb{Z}_m^*} \mid \forall j \in \mathbb{Z}_m^*,\ x_j = \overline{x_{m-j}} \in \mathbb{C}\}.$$

The space H is isomorphic as a real vector space to $K_\mathbb{R}$ via σ. Furthermore, the space H is a $\mathbb{R}$ vector space generated by the columns of the following basis matrix $\mathbf{T}$,

$$\mathbf{T} = \frac{1}{\sqrt{2}} \begin{bmatrix} \mathbf{I}_{n/2} & i\mathbf{J}_{n/2} \\ \mathbf{J}_{n/2} & -i\mathbf{I}_{n/2} \end{bmatrix} \in \mathbb{C}^{n\times n},$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{J}$ is the matrix with ones on the anti-diagonal. Let $\mathbf{h}_j$ denote the jth column of $\mathbf{T}$. Then, for any $a \in K$, there is a unique $\mathbf{v} = (v_1, \dots, v_n) \in \mathbb{R}^n$ such that $\sigma(a) = \mathbf{T}\mathbf{v}^T = \sum_{j\in[n]} v_j \mathbf{h}_j$, where σ denotes the canonical embedding.

Gaussians over H. For $r > 0$, the Gaussian function $\rho_r : H \to (0, 1]$ over H is defined as, $\rho_r(\mathbf{x}) = \exp(-\pi\|\mathbf{x}\|_2^2/r^2)$ for all $\mathbf{x} \in H$. By appropriately normalizing the Gaussian function ρ_r, we obtain the *continuous spherical Gaussian distribution* D_r *over* H. We use the basis $\{\mathbf{h}_j\}_{j\in n}$ to define the *continuous elliptical* Gaussian distribution as in [LPR10]. Let $\mathbf{r} = [r_1, \dots, r_n] \in \mathbb{R}^n_{>0}$ be a vector of positive real numbers such that $r_j = r_{n+1-j}$ for all $j \in [n]$. Then a sample $\mathbf{x}$ from the elliptical Gaussian distribution $D_\mathbf{r}$ over H is given by $\sum_{j\in[n]} v_j \mathbf{h}_j$, where each v_j are chosen independently from the one-dimensional Gaussian distribution D_{r_j} over $\mathbb{R}$. One can check that in case all r_j are the same, this distribution coincides with the above spherical Gaussian distribution, since we have $x_j = \overline{x_{n+1-j}}$ for $\mathbf{x} \in H$. In case we want to explicitly express the domain in which the Gaussian distribution is defined over, we use a superscript to denote it, e.g., $D_\mathbf{r}^H$.

The *discrete* (*spherical*) Gaussian is defined similarly to the standard lattices in $\mathbb{R}^n$. Namely, for a lattice in $\Lambda \subset H$, a vector $\mathbf{u} \in H$ and a real $r > 0$, the discrete Gaussian distribution over the coset $\Lambda + \mathbf{u}$ is defined as $D_{\Lambda+\mathbf{u},r}(\mathbf{x}) = \rho_r(\mathbf{x})/\rho_r(\Lambda + \mathbf{u})$ for all $\mathbf{x} \in \Lambda + \mathbf{u}$.

Gaussians over $K_\mathbb{R}$. Using the canonical embedding $\sigma : K_\mathbb{R} \to H$ (which is an isomorphism), we can consider a *continuous Gaussian distribution* $D_\mathbf{r}^{K_\mathbb{R}}$ *over* $K_\mathbb{R}$ induced by $D_\mathbf{r}^H$. Recall we can uniquely express any $a \in K_\mathbb{R}$ as a $\mathbb{R}$-linear combination of the power basis $\{\zeta_m^i\}_{i=0}^{n-1}$. Namely, if we denote $\boldsymbol{\zeta}$ as the ordered power basis, then $a = \phi(a)\boldsymbol{\zeta}^T$ for all $a \in K_\mathbb{R}$, where ϕ denotes the coefficient embedding. Next, let Δ_m (or CRT_m) denote the matrix corresponding with evaluating a polynomial at all the primitive mth root of unity, i.e., $\sigma(a) = \Delta_m\phi(a)^T \in H$ for all $a \in K_\mathbb{R}$. Then, using this expression a sample $a \in K_\mathbb{R}$ from $D_\mathbf{r}^{K_\mathbb{R}}$ is given by $\phi(a)\boldsymbol{\zeta}^T$, where $\mathbf{x} \in H$ is sampled from the continuous Gaussian distribution $D_\mathbf{r}^H$ and $\phi(a)$ is set as $\Delta_m^{-1}\mathbf{x}^T$. By definition, we have $D_\mathbf{r}^{K_\mathbb{R}}(a) = D_\mathbf{r}^H(\sigma(a))$. Furthermore, recalling the definition of $D_\mathbf{r}^H$, we can also view $D_\mathbf{r}^{K_\mathbb{R}}$

[6] Note that in our main body, we view R as $\mathbb{Z}[X]/(X^n + 1)$ w.l.o.g .

being induced by $D_{\mathbf{r}}^{\mathbb{R}^n} = D_{r_1} \times \cdots \times D_{r_n}$. Concretely, a sample $a \in K_{\mathbb{R}}$ of $D_{\mathbf{r}}^{K_{\mathbb{R}}}$ can also be obtained by first sampling $\mathbf{v} \in \mathbb{R}^n$ from $D_{\mathbf{r}}^{\mathbb{R}^n}$ and then setting a to satisfy $\phi(a) = \Delta_m^{-1}\mathbf{T}\mathbf{v}^T$.

Gaussians over Fractional Ideals I in K. Recall that a fractional ideal I in K is a set such that $dI \subseteq R$ is an integral ideal for some $d \in R$ and that has a $\mathbb{Z}$-basis $U = \{u_1, \ldots, u_n\} \subseteq K$. Therefore, under the canonical embedding σ, the ideal yields a rank n lattice $\sigma(\mathcal{I})$ in H having basis $\{\sigma(u_1), \ldots, \sigma(u_n)\} \subset H$. We call this lattice $\sigma(\mathcal{I})$ created by the fractional ideal $\mathcal{I}$ as an *ideal lattice*. As in the case of $K_{\mathbb{R}}$, we can consider a *discrete Gaussian distribution over the ideal* $\mathcal{I}$. For a fractional ideal $\mathcal{I} \subset K$, element $t \in K$, and real $r > 0$, the discrete Gaussian distribution over $\mathcal{I} + t$ is defined as $D_{\mathcal{I}+t,r}(a) = D_{\sigma(\mathcal{I})+\sigma(t),r}(\sigma(a))$ for all $a \in \mathcal{I} + t$.

Discretization over Ideal Lattices. Theorem 3.1 of [Pei10] holds for lattices in H. Therefore, we can use it to discretize the contiunous Gaussian distribution $D_r^{K_{\mathbb{R}}}$ to the discrete Gaussian distribution $D_{I+t,r'}$ as follows. Note that $\eta_\epsilon(I)$ denotes the smoothing parameter for the ideal lattice $\sigma(I)$.

Lemma 2.17. *Let s, s_1, s_2 be positive reals such that $s^2 \geq s_1^2 + s_2^2$. Let I be a fractional ideal in K and t an element in $K_{\mathbb{R}}$. Further assume that $s_1 \geq \eta_\epsilon(I)$ for some positive $\epsilon \leq 1/2$.Then, if we choose a_2 from the continuous Gaussian $D_{s_2}^{K_{\mathbb{R}}}$ over $K_{\mathbb{R}}$ and then choose a_1 from the discrete Gaussian D_{I+t-a_2,s_1}, then $a_1 + a_2$ is within statistical distance 8ϵ of the discrete Gaussian $D_{I+t,s}$.*

Proof. The statement is a direct result of [Pei10], Theorem 3.1 by noticing the following facts: $D_{s_2}^{K_{\mathbb{R}}}(a) = D_{s_2}^{H}(\sigma(a))$ for all $a \in K_{\mathbb{R}}$, $D_{I+t,s_1}(a) = D_{\sigma(I)+\sigma(t),s_1}(\sigma(a))$ for all $a \in I + t, t \in K_{\mathbb{R}}$, and that $\sigma(I)$ embeds as a lattice in H. □

2.5.4 Power of 2 Polynomial Rings

Here, we discuss the power of 2 polynomial rings and its properties. For the special case when m is a power of 2, the mth cyclotomic polynomial is given as $\Phi_m(X) = X^n + 1$ where $n = \varphi(m) = m/2$. Therefore, $R \cong \mathbb{Z}[X]/(X^n + 1)$. For this special case, all the columns of Δ_m are orthogonal to each other and we have $\Delta_m^{-1} = \frac{1}{n}\Delta_m^*$, where Δ_m^* is the conjugate transpose. In other words, $\frac{1}{\sqrt{n}}\Delta_m$ is a unitary matrix. Using the properties $\sigma(a) = \Delta_m \phi(a)^T$ and $\phi(\boldsymbol{b}\boldsymbol{R}) = \phi(\boldsymbol{b})\mathrm{rot}(\boldsymbol{R})$ for any element $a \in K_{\mathbb{R}}$, vector $\boldsymbol{b} \in K_{\mathbb{R}}^s$ and matrix $\boldsymbol{R} \in K_{\mathbb{R}}^{s\times t}$, we obtain the following facts:

- $\|\sigma(a)\|_2 = \sqrt{n}\|\phi(a)\|_2$,
- $s_1(\boldsymbol{R}) = \max\limits_{\boldsymbol{x}\in R^t\setminus\{\mathbf{0}\}} \dfrac{\|\sigma(\boldsymbol{x}\boldsymbol{R})\|_2}{\|\sigma(\boldsymbol{x})\|_2} = \max\limits_{\boldsymbol{x}\in R^t\setminus\{\mathbf{0}\}} \dfrac{\|\phi(\boldsymbol{x}\boldsymbol{R})\|_2}{\|\phi(\boldsymbol{x})\|_2} = \max\limits_{\mathbf{z}\in\mathbb{R}^{tn}\setminus\{\mathbf{0}\}} \dfrac{\|\mathbf{z}\cdot\mathrm{rot}(\boldsymbol{R})\|_2}{\|\mathbf{z}\|_2}$.

Recalling the definition of the continuous Gaussian distribution $D_r^{K_{\mathbb{R}}}$ and the fact that the space H has matrix $\mathbf{T}$ as its basis, $D_r^{K_{\mathbb{R}}}$ can be described by the procedure of first sampling $\mathbf{v} \xleftarrow{\$} D_r^m$, then outputting $a = \phi(a)\boldsymbol{\zeta}^T$ where $\phi(a)$ is set as $\frac{1}{\sqrt{n}}(\frac{1}{\sqrt{n}}\Delta_m^*)\mathbf{T}\mathbf{v}^T$. Therefore, since $\frac{1}{\sqrt{n}}\Delta_m^*$ and $\mathbf{T}$ are both unitary matrices, a sample from $D_r^{K_{\mathbb{R}}}$ is simply an element with its coefficients sampled from $D_{\sqrt{n}r}^m$. Finally, for the special power of 2 polynomial ring, we have $R^\vee = \frac{1}{n}R$.

2.5.5 Ring LWE on Number Fields

We start with recalling the definition of RLWE assumption on number fields (more precisely, on $K_{\mathbb{R}}$), whose hardness is shown directly in previous works.

Definition 2.5 (RLWE on $K_{\mathbb{R}}$)**.** *For integers $n = n(\lambda)$, $k = k(n)$, a prime integer $q = q(n) > 2$, a family of error distribution $\Psi = \Psi(n)$ over $K_{\mathbb{R}}$, and an PPT algorithm $\mathcal{A}$, an advantage for the RLWE problem $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,k,q,\Psi}$ of $\mathcal{A}$ is defined as follows:*

$$\mathsf{Adv}^{\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,k,q,\Psi}}_{\mathcal{A}} = |\Pr[\mathcal{A}^{\mathcal{O}_{s,\chi}}(1^\lambda, n, k, q) \to 1] - \Pr[\mathcal{A}^{\mathcal{O}_\$}(1^\lambda, n, k, q) \to 1]|$$

where $s \overset{\$}{\leftarrow} R^\vee_q, \chi \overset{\$}{\leftarrow} \Psi$. The oracles $\mathcal{O}_\$$ and $\mathcal{O}_{s,\chi}$ are specified as follows.

$\mathcal{O}_{s,\chi}$: *When called, it picks $a \overset{\$}{\leftarrow} R_q$, $e \overset{\$}{\leftarrow} \chi$ and returns $(a, as/q + e)$.*

$\mathcal{O}_\$$: *When called, it returns $(a, v) \overset{\$}{\leftarrow} R_q \times K_{\mathbb{R}}/R^\vee$.*

Both oracles can be called at most k times. If there is no bound on the number of calls, we denote $k = \infty$. In case Ψ consists of a single distribution χ we simply treat the set Ψ as a distribution and write $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,k,q,\chi}$, and for this particular case $\mathcal{A}$ further receives as input the distribution χ used by the oracle. We say that $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,k,q,\Psi}$ assumption holds if $\mathsf{Adv}^{\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,k,q,\Psi}}_{\mathcal{A}}$ is negligible for all PPT $\mathcal{A}$.

In [LPR10], it is shown that solving $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,\infty,p,\Psi}$ with prime p such that $p \equiv 1 \mod m$ and certain Ψ is as hard as quantumly approximating SIVP (or SVP) on ideal lattices in the worst case. In the subsequent work [LS15], it is shown that the former can be further reduced to $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,\infty,q,\Psi'}$ with any q and a certain Ψ'. In what follows, $\Psi_{\leq\alpha}$ denotes the family of all elliptical Gaussian distributions $D^{K_{\mathbb{R}}}_{\mathbf{r}}$ where each parameter $r_i \leq \alpha$. Furthermore, Υ_β is a certain family of distribution that is parametrized by $\beta \in \mathbb{R}$. Since the precise definition is not necessary for our purpose, we omit this and refer to [LPR10, LS15]. Then, we have the following results.

Lemma 2.18 ([LPR10], Theorem 3.6)**.** *Let $\beta > 0$ and let $p \geq 2$, $p \equiv 1 \mod m$ be a polynomially bounded prime such that $\beta p \geq \omega(\sqrt{\log n})$. Then there is a probabilistic polynomial-time quantum reduction from $\tilde{O}(\sqrt{n}/\beta)$-approximate SIVP (or SVP) to $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,\infty,p,\Upsilon_\beta}$.*

Lemma 2.19 ([LS15], From Lemma 4.22, 4.24, and 4.26)**.** *Let $p, q \geq 2$ be polynomially bounded primes and $\alpha, \beta \in (0, 1)$ such that $\alpha \geq \beta \cdot \max\{1, p/q\} \cdot n^{3/4}\omega(\log^2 n)$ and $\beta p \geq \omega(\sqrt{\log n/n})$. There exists a polynomial reduction from $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,\infty,p,\Upsilon_\beta}$ to $\mathsf{RLWE}^{K_{\mathbb{R}}}_{n,\infty,q,\Psi_{\leq\alpha}}$.*

By combining the above Lemmas, we obtain the hardness of the RLWE with arbitrary modulus q for a skewed Gaussian. In the next step (Lemma 2.22), we further reduce it to the RLWE with spherical Gaussian. To prepare for the proof, we define Réniy Divergence (of order 2) and review its properties following [LPR10, BLL$^+$15].

Definition 2.6 (Rényi Divergence)**.** *Let us consider two density functions $P, Q : \mathbb{R}^n \to \mathbb{R}^{\geq 0}$ where $P(\mathbf{x}) = 0$ whenever $Q(\mathbf{x}) = 0$. We define the Rényi divergence $RD(P\|Q)$ as*

$$RD(P\|Q) = \int_{\mathbb{R}^n} \frac{P(\mathbf{x})^2}{Q(\mathbf{x})} \mathrm{d}\mathbf{x}.$$

For Rényi Divergence, the following properties hold. For any distribution P and Q, we have $RD(P\|P) = 1$ and $RD(P\|Q) \geq 1$. Let us assume that P (resp. Q) is a direct product of independent distributions P_1 and P_2 (resp. Q_1 and Q_2). Then, we have $RD(P\|Q) = RD(P_1 \times P_2\|Q_1 \times Q_2) = RD(P_1\|Q_1) \cdot RD(P_2\|Q_2)$.

Lemma 2.20 ([LPR10], Claim 5.15)**.** *Let $r_1, \dots r_n \in \mathbb{R}^+$ and $s_1, \dots, s_n \in \mathbb{R}^+$ be such that for all i, $|s_i/r_i - 1| < \sqrt{\log n/n}$. Then, there exists an polynomial $f_{RD} : \mathbb{N} \to \mathbb{R}$ such that $RD(D_{r_1} \times \cdots \times D_{r_n} \| D_{s_1} \times \cdots \times D_{s_n}) = f_{RD}(n)$.*

Lemma 2.21 (Implicit in [LPR10])**.** *Let P and Q denote distributions with $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$. Let $A \subseteq \mathrm{Supp}(Q)$ be any set. Then, we have $Q(A) \geq P(A)^2/RD(P\|Q)$ where $P(A)$ and $Q(A)$ are measure of A under P and Q, respectively.*

Here, we review the proof of [LPR10] that converts the error distribution from the skewed Gaussian to the spherical Gaussian.

Lemma 2.22 (Adapted from [LPR10], Lemma 5.16)**.** *Let q be a polynomially bounded prime, k a positive integer and $\alpha, \beta \in (0,1)$. There exists a polynomial time reduction from $\mathsf{RLWE}^{K_\mathbb{R}}_{n,\infty,q,\Psi_{\leq\alpha}}$ to $\mathsf{RLWE}^{K_\mathbb{R}}_{n,k,q,\chi}$ with $\chi = D^{K_\mathbb{R}}_{\xi}$ where $\xi = \alpha(nk/\log(nk))^{1/4}$.*

Proof. We construct an adversary $\mathcal{B}$ against $\mathsf{RLWE}^{K_\mathbb{R}}_{n,\infty,q,\Psi_{\leq\alpha}}$ from adversary $\mathcal{A}$ that solves $\mathsf{RLWE}^{K_\mathbb{R}}_{n,k,q,\chi}$ with non-negligible advantage $\epsilon(\lambda)$. By assumption, there exists a constant $c \in \mathbb{N}$ such that $\epsilon(\lambda) > 1/\lambda^c$ for infinitely many $\lambda \in \mathbb{N}$.

Reduction. $\mathcal{B}$ is equipped with an oracle $\mathcal{O}$ and its task is to distinguish whether $\mathcal{O} = \mathcal{O}_{s,\chi'}$ or $\mathcal{O} = \mathcal{O}_\$$, where $\chi' = D^{K_\mathbb{R}}_{\mathbf{r}} \xleftarrow{\$} \Psi_{\leq\alpha}$. $\mathcal{B}$ proceeds as follows. It first obtains estimate $\hat{p}_0$ for the probability

$$p_0 := \Pr[\mathcal{A}(\{(a_i, v_i)\}_{i=1}^k) \to 1 | (a_1, v_1), \dots, (a_k, v_k) \xleftarrow{\$} R_q \times K_\mathbb{R}/R^\vee]$$

by running $\mathcal{A}$ on $N := 100\lambda^{2c+1}$ fresh inputs. It then repeats the following $M := 4\lambda^{2c+1} f_{RD}(nk)$ times, where f_{RD} is the polynomial specified in Lemma 2.20.

- It picks random $s' \xleftarrow{\$} R_q^\vee$ and $e'_1, \dots, e'_k \xleftarrow{\$} D^{K_\mathbb{R}}_\xi$. Then it obtains estimate $\hat{p}_1(s', e'_1, \dots, e'_k)$ for the probability

  $$p_1(s', e'_1, \dots, e'_k) := \Pr[\mathcal{A}(\{(a_i, v_i + a_i s'/q + e'_i)\}_{i=1}^k) \to 1 | (a_1, v_1), \dots, (a_k, v_k) \xleftarrow{\$} \mathcal{O}]$$

  by running $\mathcal{A}$ on N fresh inputs. This can be done by calling the oracle Nk times.

If it happens that $|\hat{p}_1(s', e'_1, \dots, e'_k) - \hat{p}_0| > 1/4\lambda^c$ at any point during the loop, it outputs 1. Otherwise it outputs 0.

Analysis. It is clear that $\mathcal{B}$ is a (probabilistic) polynomial time algorithm. It suffices to show that $\mathcal{B}$ has overwhelming advantage when $\epsilon > 1/\lambda^c$. We note that by the Hoeffding bound, $|p_0 - \hat{p}_0| < 1/10\lambda^c$ and $|p_1(s', e'_1, \dots, e'_k) - \hat{p}_1(s', e'_1, \dots, e'_k)| < 1/10\lambda^c$ for any $(s', e'_1, \dots, e'_k)$ hold except for probability $e^{-N\cdot(1/10\lambda^c)^2} < 2^{-\lambda}$. In the following, we assume that these always hold.

We first observe that if the oracle $\mathcal{O} = \mathcal{O}_\$$, it is clear that both inputs to $\mathcal{A}$ follow the the uniform distribution over $R_q \times K_\mathbb{R}/R^\vee$. Therefore, $p_0 = p_1(s', e'_1, \dots, e'_k)$ holds for any $(s', e'_1, \dots, e'_k)$. Thus,

$$\begin{aligned} |\hat{p}_0 - \hat{p}_1(s', e'_1, \dots, e'_k)| &\leq |\hat{p}_0 - p_0| + |p_0 - p_1(s', e'_1, \dots, e'_k)| + |p_1(s', e'_1, \dots, e'_k) - \hat{p}_1(s', e'_1, \dots, e'_k)| \\ &\leq 1/10\lambda^c + 1/10\lambda^c < 1/4\lambda^c. \end{aligned}$$

Hence, $\mathcal{B}$ outputs 0 with all but negligible probability.

Next, let us consider the case where $\mathcal{O} = \mathcal{O}_{s,\chi'}$. In this case, during the loop, an input to $\mathcal{A}$ is of the form $\{(a_i, a_i(s+s')/q + e_i + e'_i)\}_{i=1}^k$ where $e_i \xleftarrow{\$} D^{K_\mathbb{R}}_{\mathbf{r}}$ and $e'_i \xleftarrow{\$} D^{K_\mathbb{R}}_\xi$ for $i \in [k]$. Let us define the vector $\mathbf{r}'$ with coordinates $r'^2_j = \xi^2 - r_j^2$. We claim that the average of $p_1(s', e'_1, \ldots, e'_k)$ over $e'_1, \ldots, e'_k$ chosen independently from $D_{\mathbf{r}'}^{K_\mathbb{R}}$ (rather than $D_\xi^{K_\mathbb{R}}$, which is the actual distribution) is at least $1/\lambda^c$ far from p_0. This can be seen by observing that the error terms $e_i + e'_i$ are distributed as $D_{\mathbf{r}}^{K_\mathbb{R}} + D_{\mathbf{r}'}^{K_\mathbb{R}} = D_\xi^{K_\mathbb{R}}$ and by our assumption on $\mathcal{A}$. Let us define S as the set of all tuples $(s', e'_1, \ldots, e'_k)$ such that $|p_1(s', e'_1, \ldots, e'_k) - p_0| > 1/2\lambda^c$. By the averaging argument, we have that the measure of S over $U(R_q) \times (D_{\mathbf{r}'}^{K_\mathbb{R}})^k$ is at least $1/2\lambda^c$. Now, let us consider the measure of S over $U(R_q) \times (D_\xi^{K_\mathbb{R}})^k$, which is the actual distribution. By the definition of $D_{\mathbf{r}}^{K_\mathbb{R}}$ and since $1 \le \xi/\sqrt{\xi^2 - r'^2_i} \le \xi/\sqrt{\xi^2 - \alpha^2} \le 1 + \sqrt{\log(nk)/nk}$, we have

$$RD(U(R_q) \times (D_{\mathbf{r}'}^{K_\mathbb{R}})^k \| U(R_q) \times (D_\xi^{K_\mathbb{R}})^k) = RD((D_{\mathbf{r}'}^{K_\mathbb{R}})^k \| (D_\xi^{K_\mathbb{R}})^k) = f_{RD}(nk)$$

by Lemma 2.20. Hence, by Lemma 2.21, we have that the measure of S over $U(R_q) \times (D_\xi^{K_\mathbb{R}})^k$ is at least $1/4\lambda^{2c} f_{RD}(nk)$. Therefore, B picks $(s', e'_1, \ldots, e'_k)$ in S at least once during the loop except for probability $(1 - 1/4\lambda^{2c} f_{RD}(nk))^M < 2^{-\lambda}$. Furthermore, for $(s', e'_1, \ldots, e'_k) \in S$, we have that

$$\begin{aligned} |\hat{p}_1(s', e'_1, \ldots, e'_k) - \hat{p}_0| &\ge |p_1(s', e'_1, \ldots, e'_k) - p_0| - |\hat{p}_1(s', e'_1, \ldots, e'_k) - p_1(s', e'_1, \ldots, e'_k)| - |\hat{p}_0 - p_0| \\ &> 1/2\lambda^c - 1/10\lambda^c - 1/10\lambda^c > 1/4\lambda^c \end{aligned}$$

Therefore, $\mathcal{B}$ outputs 1 with all but negligible probability in this case. □

Finally, we discretize the error distribution and get rid of $R^\vee$ by scaling it appropriately. The following $\mathsf{RLWE}_{n,k,q,\chi}$ is the problem we considered in the main body of our work (cf. Definition 2.3).

Lemma 2.23. *Let m be a power of 2, $n = \varphi(m) = m/2$, k be an integer, $q \equiv 3 \mod 8$ be a prime number, and ξ a positive real satisfying $\xi \ge \omega(\sqrt{\log n/n})/q$. There exists a polynomial time reduction from $\mathsf{RLWE}^{K_\mathbb{R}}_{n,k,q,\chi}$ with $\chi = D_\xi^{K_\mathbb{R}}$ to $\mathsf{RLWE}_{n,k,q,\chi}$ with $\chi = D^{\mathsf{coeff}}_{\mathbb{Z}^n, \sqrt{2}nq\xi}$.*

Proof. To show the theorem, it suffices to show an efficient transformation T that takes $\{(a_i, v_i)\}_{i=1}^k \in (R_q \times K_\mathbb{R}/R^\vee)^k$ chosen from either $\mathcal{O}_\$$ or $\mathcal{O}_s$ as input and has the following properties.

- If $(a_i, v_i) \xleftarrow{\$} \mathcal{O}_\$$ for $i \in [k]$, the output of T is uniform over $(R_q \times R_q)^k$.
- If $(a_i, v_i) \xleftarrow{\$} \mathcal{O}_s$ for $i \in [k]$, the output of T is of the form $\{(a_i, a_i s' + e'_i)\}_{i=1}^k$ where $s' \xleftarrow{\$} R_q$ and $e'_1 \ldots, e'_k \xleftarrow{\$} D^{\mathsf{coeff}}_{\mathbb{Z}^n, \sqrt{2}nq\xi}$.

Given $\{(a_i, v_i)\}_{i=1}^k$, T first discretizes $v_i \in K_\mathbb{R}/R^\vee$ to $\bar{v}_i \in \frac{1}{q}R^\vee/R^\vee$ while preserving the correct error distribution by adding samples d_i chosen from $D_{\frac{1}{q}R^\vee - v'_i, \xi}$ to each v_i where $v'_i = v_i \mod \frac{1}{q}R^\vee$. We show the validity of this procedure. The case when the input to T is from $\mathcal{O}_\$$ is trivial. Hence, we assume the input was from $\mathcal{O}_s$, i.e., $v_i = a_i s + e_i$ for $e_i \xleftarrow{\$} D_\xi^{K_\mathbb{R}}$. For the special case when m is a power of 2, we have $\eta_\epsilon(\frac{1}{q}R^\vee) = \omega(\sqrt{\log n/n})/q$ for some negligible $\epsilon > 0$. Therefore, by the condition on ξ and from Lemma 2.17, $\bar{e}_i = e_i + d_i$ is distributed negligibly close to the discrete Gaussian distribution $D_{\frac{1}{q}R^\vee, \sqrt{2}\xi}$ when $e_i \xleftarrow{\$} D_\xi^{K_\mathbb{R}}$ and $d_i \xleftarrow{\$} D_{\frac{1}{q}R^\vee - e_i, \xi}$. Since $e_i = v'_i \mod \frac{1}{q}R^\vee$, this d_i has the same distribution as the d_i sampled in the above procedure. Therefore, T outputs $\bar{v}_i = a_i s + \bar{e}_i$ where $\bar{e}_i \xleftarrow{\$} D_{\frac{1}{q}R^\vee, \sqrt{2}\xi}$ if the input is from $\mathcal{O}_s$.

Then, T sets $v_i' = qn\bar{v}_i$ in order to move into R. We can see that $\{v_i'\}_{i=1}^k$ are uniformly distributed over R_q when the oracle is $\mathcal{O}_\$$. This is because $R^\vee = \frac{1}{n}R$, which holds whenever m is a power of 2. When $\mathcal{O} = \mathcal{O}_s$, we have $v_i' = a_i ns + qne_i$. We can see that $s' := ns$ is uniformly random over R_q. We can also see that the distribution of qne_i follows $D_{R,\sqrt{2}qn\xi}$. We complete the proof by observing that for m a power of 2, we have $D_{R,\sqrt{2}qn\xi} = D^{\mathsf{coeff}}_{\mathbb{Z}^n,\sqrt{2n}q\xi}$, which follows from the fact that $\phi(R) = \mathbb{Z}^n$ and $\|\sigma(a)\| = \sqrt{n}\|\phi(a)\|$ for any $a \in K_\mathbb{R}$. Recall that $D^{\mathsf{coeff}}_{\mathbb{Z}^n,\sqrt{2n}q\xi}$ is the distribution of $a \in R$ where the coefficient vector of a is sampled from $D_{\mathbb{Z}^n,\sqrt{2n}q\xi}$

$\square$

Chapter 3

Tighter Security Proofs for GPV-IBE in the Quantum ROM

3.1 Introduction

Shor [Sho94b] in his breakthrough result showed that if a quantum computer is realized, then almost all cryptosystems used in the real world will be broken. Since then, a significant amount of studies have been done in the area of post-quantum cryptography, whose motivation is constructing cryptosystems secure against quantum adversaries. Recently in 2016, the National Institute of Standards and Technology (NIST) initiated the Post-Quantum Cryptography Standardization, and since then post-quantum cryptography has been gathering increasingly more attention.

Random Oracles in Quantum World. In general, security proofs of practical cryptographic schemes are given in the random oracle model (ROM) [BR93], which is an idealized model where a hash function is modeled as a publicly accessible oracle that computes a random function. Boneh et al. [BDF⁺11] pointed out that the ROM as in the classical setting is not reasonable when considering security against quantum adversaries, since quantum adversaries may compute hash functions over quantum superpositions of many inputs. Considering this fact, as a reasonable model against quantum adversaries, they proposed a new model called the quantum random oracle model (QROM), where a hash function is modeled as a *quantumly accessible* random oracle. As discussed in [BDF⁺11], many commonly-used proof techniques in the ROM do not work in the QROM. Therefore even if we have a security proof in the ROM, we often require new techniques to obtain similar results in the QROM.

Identity-based Encryption in QROM. Identity-Based Encryption (IBE) is a generalization of a public key encryption scheme where the public key of a user can be any arbitrary string such as an e-mail address. The first IBE scheme based on a post-quantum assumption is the one proposed by Gentry, Peikert and Vaikuntanathan (GPV-IBE) [GPV08], which is based on the learning with errors (LWE) assumption [Reg05]. To this date, GPV-IBE is still arguably the most efficient IBE scheme that is based on a hardness assumption that resists quantum attacks. However, since their original security proof was made in the ROM instead of the QROM, it was unclear if we could say the scheme is truly post-quantum. Zhandry [Zha12b] answered this in the affirmative by proving that the GPV-IBE is indeed secure in the QROM under the LWE assumption, hence truly post-quantum, by developing new techniques in the QROM.

[0]The contents of this chapter is based on the work presented at Asiacrypt 2018 under the title "Tighter Security Proofs for GPV-IBE in the Quantum ROM".

Tight Security of GPV-IBE. However, if we consider the tightness of the reduction, the security proof of the GPV-IBE by Zhandry [Zha12b] does not provide a satisfactory security. Specifically, GPV-IBE may be efficient in the ROM, but it is no longer efficient in the QROM. In general, a cryptographic scheme is said to be tightly secure under some assumption if breaking the security of the scheme is as hard as solving the assumption. More precisely, suppose that we proved that if there exists an adversary breaking the security of the scheme with advantage ϵ and running time T, we can break the underlying assumption with advantage ϵ' and running time T'. We say that the scheme is tightly-secure if we have $\epsilon'/T' \approx \epsilon/T$. By using this notation, Zhandry gave a reduction from the security of GPV-IBE to the LWE assumption with $\epsilon' \approx \epsilon^2/(Q_{\mathsf{H}}+Q_{\mathsf{ID}})^4$ and $T' \approx T + (Q_{\mathsf{H}}+Q_{\mathsf{ID}})^2 \cdot \mathsf{poly}(\lambda)$ where Q_{H} denotes the number of hash queries, Q_{ID} denotes the number of secret key queries, λ denotes the security parameter, and poly denotes some fixed polynomial. Though the reduction is theoretically interesting, the meaning of the resulting security bound in a realistic setting is unclear. For example, if we want to obtain 128-bit security for the resulting IBE, and say we had $\epsilon = 2^{-128}$, $Q_{\mathsf{H}} = 2^{100}$, $Q_{\mathsf{ID}} = 2^{20}$, then even if we ignore the blowup for the running time, we would have to start from at least a 656-bit secure LWE assumption, which incurs a significant blowup of the parameters. Indeed, Zhandry left it as an open problem to give a tighter reduction for the GPV-IBE.

Multi-Challenge Tightness. The standard security notion of IBE considers the setting where an adversary obtains only one challenge ciphertext. This is because security against adversaries obtaining many challenge ciphertexts can be reduced to the security in the above simplified setting. However, as pointed out by Hofheinz and Jager [HJ12], tightness is not preserved in the above reduction since the security degrades by the number of ciphertexts. Therefore tightly secure IBE in the single-challenge setting does not imply tightly secure IBE in the multi-challenge setting. On the other hand, in the real world, it is natural to assume that an adversary obtains many ciphertexts, and thus tight security in the multi-challenge setting is desirable. However, there is no known security proof for the GPV-IBE or its variant that does not degrade with the number of challenge ciphertexts even in the classical setting.

3.1.1 Our Contribution

We provide much tighter security proofs for the GPV-IBE in the QROM in the single-challenge setting. Furthermore, we provide a multi-challenge tight variant of GPV-IBE that is secure both in the ROM and QROM. In the following, we describe the tightness of our security proofs by using the same notation as in the previous section.

- In the single-challenge setting, we give a reduction from the security of GPV-IBE to the LWE assumption with $\epsilon' \approx \epsilon$ and $T' = T + (Q_{\mathsf{H}} + Q_{\mathsf{ID}})^2 \cdot \mathsf{poly}(\lambda)$. If we additionally assume quantumly secure pseudorandom functions (PRFs), then we further obtain a tighter reduction, which gives $\epsilon' \approx \epsilon$ and $T' = T + (Q_{\mathsf{H}} + Q_{\mathsf{ID}}) \cdot \mathsf{poly}(\lambda)$. This is the first security proof for GPV-IBE whose security bound does not degrade with Q_{H} or Q_{ID} even in the classical setting. We note that the same security bound can be achieved without assuming PRFs in the classical ROM.

- We give a slight variant of GPV-IBE scheme whose multi-challenge security is reduced to the LWE assumption with $\epsilon' = \epsilon/\mathsf{poly}(\lambda)$ and $T' \approx T + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}})^2 \cdot \mathsf{poly}(\lambda)$ where Q_{ch} denotes the number of challenge queries. If we additionally assume quantumly secure PRFs, then we further obtain a tighter reduction. Namely, ϵ' is the same as the above, and $T' = T + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot \mathsf{poly}(\lambda)$. This is the first variant of the GPV-IBE scheme whose security bound does not degrade with Q_{ch} even in the classical setting. We note that the same security bound can be achieved without assuming PRFs in the classical ROM.

Moreover, our security proofs are much simpler than the one by Zhandry [Zha12b]. In his work, he introduced new techniques regarding indistinguishability of oracles against quantum adversaries. Though his techniques are general and also useful in other settings (e.g., [Zha12a]), it involves some arguments on quantum computation, and they are hard to follow for cryptographers who are not familiar with quantum computation. On the other hand, our proofs involve a minimal amount of discussions about quantum computation, and our proofs are done almost similar to the counterparts in the classical ROM.

3.2 Technical Overview

GPV-IBE. First, we briefly describe the GPV-IBE [GPV08], which is the main target of this chapter. A master public key is a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a master secret key is its trapdoor $\mathbf{T}_{\mathbf{A}} \in \mathbb{Z}^{m\times m}$, which enables one to compute a short vector $\mathbf{e} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\mathbf{e} = \mathbf{u}$ given an arbitrary vector $\mathbf{u} \in \mathbb{Z}_q^n$. A private key $\mathsf{sk}_{\mathsf{ID}}$ for an identity $\mathsf{ID} \in \mathcal{ID}$ is a short vector $\mathbf{e} \in \mathbb{Z}_q^m$ such that $\mathbf{A}\mathbf{e} = \mathbf{u}_{\mathsf{ID}}$ where $\mathbf{u}_{\mathsf{ID}} = \mathsf{H}(\mathsf{ID})$ for a hash function $\mathsf{H} : \mathcal{ID} \to \mathbb{Z}_q^n$, which is modeled as a random oracle. A ciphertext for a message $\mathsf{M} \in \{0,1\}$ consists of $c_0 = \mathbf{u}_{\mathsf{ID}}^\top \mathbf{s} + x + \mathsf{M}\lfloor q/2 \rceil$ and $\mathbf{c}_1 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}$. Here $\mathbf{s}$ is a uniformly random vector over $\mathbb{Z}_q^n$ and $x, \mathbf{x}$ are small "noise" terms where each entries are sampled from some specific Gaussian distribution χ. Decryption can be done by computing $w = c_0 - \mathbf{c}_1^\top \mathbf{e}_{\mathsf{ID}} \in \mathbb{Z}_q$ and deciding if w is closer to 0 or to $\lfloor q/2 \rceil$ modulo q.

Security Proof in Classical ROM. The above IBE relies its security on the LWE assumption, which informally states the following: given a uniformly random matrix $[\mathbf{A}|\mathbf{u}] \leftarrow \mathbb{Z}_q^{n\times(m+1)}$ and some vector $\mathbf{b} \in \mathbb{Z}_q^{m+1}$, there is no PPT algorithm that can decide with non-negligible probability whether $\mathbf{b}$ is of the form $[\mathbf{A}|\mathbf{u}]^\top \mathbf{s} + \mathbf{x}'$ for some $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and $\mathbf{x}' \leftarrow \chi^{m+1}$, or a uniformly random vector over $\mathbb{Z}_q^{m+1}$, i.e., $\mathbf{b} \leftarrow \mathbb{Z}_q^{m+1}$. Below, we briefly recall the original security proof in the classical ROM given by Gentry et al. [GPV08] and see how the random oracle is used by the reduction algorithm. The proof relies on a key lemma which states that we can set $\mathsf{H}(\mathsf{ID})$ and $\mathbf{e}$ in the "reverse order" from the real scheme. That is, we can first sample $\mathbf{e}$ from some distribution and program $\mathsf{H}(\mathsf{ID}) := \mathbf{A}\mathbf{e}$ so that their distributions are close to uniformly random as in the real scheme. In the security proof, a reduction algorithm guesses $i \in [Q]$ such that the adversary's i-th hash query is the challenge identity ID^* where Q denotes the number of hash queries made by the adversary. Then for all but the i-th hash query, the reduction algorithm programs $\mathsf{H}(\mathsf{ID})$ in the above manner, and for the i-th query, it programs the output of $\mathsf{H}(\mathsf{ID}^*)$ to be the vector $\mathbf{u}$ contained in the LWE instance that is given as the challenge. Specifically, the reduction algorithm sets the challenge user's identity vector $\mathbf{u}_{\mathsf{ID}^*}$ as the random vector $\mathbf{u}$ contained in the LWE instance. If the guess is correct, then it can embed the LWE instance into the challenge ciphertexts c_0^* and $\mathbf{c}_1^*$; in case it is a valid LWE instance, then $(c_0^*, \mathbf{c}_1^*)$ is properly set to $(\mathbf{u}_{\mathsf{ID}^*}^\top \mathbf{s} + x + \mathsf{M}\lfloor q/2 \rceil, \mathbf{A}^\top \mathbf{s} + \mathbf{x})$ as in the real scheme. Therefore, the challenge ciphertext can be switched to random due to the LWE assumption. After this switch, M is perfectly hidden and thus the security of GPV-IBE is reduced to the LWE assumption. Since the reduction algorithm programs the random oracle in the same way except for the challenge identity, this type of proof methodology is often times referred to as the "all-but-one programming".

Security Proof in QROM in [Zha12b]. Unfortunately, the above proof cannot be simply extended to a proof in the QROM. The reason is that in the QROM, even a single hash query can be a superposition of *all* the identities. In such a case, to proceed with the above all-but-one programming approach, the reduction algorithm would have to guess a single identity out of all the possible identities which he hopes that would be used as the challenge identity ID^* by the adversary. Obviously, the probability of the reduction algorithm being right is negligible, since the number of possible identities is exponentially large. This is in sharp contrast with the ROM setting, where the reduction algorithm was allowed to guess the single identity out of the polynomially many (classical) random oracle queries made by the adversary. Therefore, the all-but-one programming as in the classical case cannot be used in the quantum case. To overcome this barrier, Zhandry [Zha12b] introduced a useful lemma regarding what he calls the semi-constant distribution. The semi-constant distribution with parameter $0 < p < 1$ is a distribution over functions from $\mathcal{X}$ to $\mathcal{Y}$ such that a function chosen according to the distribution gives the same fixed value for random p-fraction of all inputs, and behaves as a random function for the rest of the inputs. He proved that a function according to the semi-constant distribution with parameter p and a random function cannot be distinguished by an adversary that makes Q oracle queries with advantage greater than $\frac{8}{3}Q^4p^2$. In the security proof, the reduction algorithm partitions the set of identities into controlled and uncontrolled sets. The uncontrolled set consists of randomly chosen p-fraction of all identities, and the controlled set is the complement of it. The reduction algorithm embeds an LWE instance into the uncontrolled set, and programs the hash values for the controlled set so that the decryption keys for identities in the controlled set can be extracted efficiently. Then the reduction algorithm works as long as the challenge identity falls inside the uncontrolled set and all identities for secret key queries fall inside the controlled set (otherwise it aborts). By appropriately setting p, we can argue that the probability that the reduction algorithm does not abort is non-negligible, and thus the security proof is completed. Though this technique is very general and useful, a huge reduction loss is inherent as longs as we take the above strategy because the reduction algorithm has to abort with high probability. It may be useful to point out for readers who are familiar with IBE schemes in the standard model that the above technique is conceptually very similar to the partitioning technique which is often used in the context of adaptively secure IBE scheme in the standard model [Wat05, ABB10, CHKP10]. The reason why we cannot make the proof tight is exactly the same as that for the counterparts in the standard model.

Our Tight Security Proof in QROM. As discussed above, we cannot obtain a tight reduction as long as we use a partitioning-like technique. Therefore we take a completely different approach, which is rather similar to that used in the public key encryption scheme of Cramer and Shoup [CS98], which has also been applied to the pairing-based IBE construction of Gentry [Gen06]. The idea is that we simulate in a way so that we can create exactly one valid secret key for every identity. Note that this is opposed to the partitioning technique (and the all-but-one programming technique) where the simulator cannot create a secret key for an identity in the uncontrolled set. To create the challenge ciphertext, we use the one secret key we know for that challenge identity. If the adversary can not tell which secret key the ciphertext was created from and if there are potentially many candidates for the secret key, we can take advantage of the entropy of the secret key to statistically hide the message.

In more detail, the main observation is that the secret key $\mathbf{e}$, i.e. a short vector $\mathbf{e}$ such that $\mathbf{A}\mathbf{e} = \mathbf{u}$, retains plenty of entropy even after fixing the public values $\mathbf{A}$ and $\mathbf{u}$. Therefore, by programming the hash value $\mathbf{u}$ of an identity, we can easily create a situation where the simulator knows exactly one secret key out of the many possible candidates. Furthermore, the simulator knowing a secret key $\mathbf{e}_{\mathsf{ID}^*}$ such that $\mathbf{A}\mathbf{e}_{\mathsf{ID}^*} = \mathbf{u}_{\mathsf{ID}^*}$, can simulate the challenge ciphertext by creating $c_0^* = \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{c}_1^* + \mathsf{M}\lfloor q/2 \rceil$ and $\mathbf{c}_1^* = \mathbf{A}^\top \mathbf{s} + \mathbf{x}$. Here, the key observation is that we no longer require the LWE instance $(\mathbf{u}_{\mathsf{ID}^*}, \mathbf{u}_{\mathsf{ID}^*}^\top \mathbf{s} + x)$ to simulate the challenge ciphertext. Though the distribution of c_0^* simulated as above is slightly different from that of the real ciphertext due to the difference in the noise distributions, we ignore it in this overview. In the real proof, we overcome this problem by using the noise rerandomization technique by Katsumata and Yamada [KY16]. Then we use the LWE assumption to switch $\mathbf{c}_1^*$ to random. Finally, we argue that $\mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{c}_1^*$ is almost uniform if the min-entropy of $\mathbf{e}_{\mathsf{ID}^*}$ is high and $\mathbf{c}_1^*$ is uniformly random due to the leftover hash lemma. Therefore, all information of the message M is hidden and thus the proof is completed.

Finally, we observe that the above proof naturally fits in the QROM setting. The crucial difference from the partitioning technique is that in our security proof we program the random oracle in the same way for all identities. Therefore even if an adversary queries a superposition of all identities, the simulator can simply quantumly perform the programming procedure for the superposition. Thus the proof in the classical ROM can be almost automatically converted into the one in the QROM in this case.

Tight Security in Multi-Challenge Setting. Unfortunately, the above idea does not extend naturally to the tightly-secure multi-challenge setting. One can always prove security in the multi-challenge setting starting from a scheme that is single-challenge secure via a hybrid argument, however, as mentioned by Hofheinz and Jager [HJ12], this type of reduction does not preserve tightness. A careful reader may think that the above programming technique can be extended to the multi-challenge setting, hence bypassing the hybrid argument. We briefly explain why this is not the case. Informally, in the above proof, the reduction algorithm embeds its given LWE instance $(\mathbf{A}, \mathbf{A}^\top \mathbf{s} + \mathbf{x})$ into the challenge ciphertext by creating $(c_0^* = \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{c}_1^* + \mathsf{M}\lfloor q/2 \rceil, \mathbf{c}_1^* = \mathbf{A}^\top \mathbf{s} + \mathbf{x})$, where $\mathbf{e}_{\mathsf{ID}^*}$ is the secret key of the challenge user $\mathbf{u}_{\mathsf{ID}^*}$. Therefore, since the $\mathbf{c}_1^*$ component of every ciphertext is an LWE instance for the same public matrix $\mathbf{A}$, to simulate multiple challenge ciphertexts in the above manner, the reduction algorithm must be able to prepare a special type of LWE instance $(\mathbf{A}, \{\mathbf{A}^\top \mathbf{s}^{(k)} + \mathbf{x}^{(k)}\}_{k \in [N]})$, where $N = \mathsf{poly}(\lambda)$ is the number of challenge ciphertext queried by the adversary. It can be easily seen that this construction is tightly-secure in the multi-challenge setting with the same efficiency as the single-challenge setting, *if* we assume that this special type of LWE problem is provided to the reduction algorithm as the challenge. However, unfortunately, we still end up losing a factor of N in the reduction when reducing the standard LWE problem to this special LWE problem. In particular, we only shifted the burden of having to go through the N hybrid arguments to the assumption rather than to the scheme. As one may have noticed, there is a way to bypass the problem of going through the N hybrid arguments by using conventional techniques (See [Reg05, Reg10]) of constructing an unlimited number of fresh LWE instances given a fixed number of LWE instances. However, this techniques requires the noise of the newly created LWE instances to grow proportionally to the number of created instances. In particular, to create the above special LWE instance from a standard LWE instance, we require the size of the noise $\mathbf{x}^{(k)}$ to grow polynomially with N, where recall that N can be an arbitrary polynomial. Hence, although we can show a tightly secure reduction in the multi-challenge setting, for the concrete parameters of the scheme to be independent of N, we need to assume the super-polynomial LWE assumption to cope with the super-polynomial noise blow up. This is far more inefficient than in the single-challenge setting where we only require a polynomial LWE assumption.

To overcome this problem, we use the "lossy mode" of the LWE problem. It is well known that the secret vector $\mathbf{s}$ is uniquely defined given an LWE instance $(\mathbf{A}, \mathbf{A}^\top \mathbf{s} + \mathbf{x})$ for large enough samples. A series of works, e.g., [GKPV10, BKPW12, AKPW13, LSSS17] have observed that if we instead sample $\mathbf{A}$ from a special distribution that is computationally indistinguishable from the uniform distribution, then $(\mathbf{A}, \mathbf{A}^\top \mathbf{s} + \mathbf{x})$ leaks almost no information of the secret $\mathbf{s}$, hence the term "lossy mode". This idea can be leveraged to prove (almost) tight security of the above single-challenge construction, where the reduction loss is independent of the number of challenge ciphertext. A first attempt of using this idea is as follows: During the security proof of the GPV-IBE, we first change the public matrix $\mathbf{A}$ to a lossy matrix $\tilde{\mathbf{A}}$ and generate the secret keys and program the random oracle in the same way as before. To create the challenge ciphertexts, the reduction algorithm honestly samples $\mathbf{s}^{(k)}$, $x^{(k)}$, $\mathbf{x}^{(k)}$ and sets $(c_0^* = \mathbf{u}_{\mathsf{ID}^*}^\top \mathbf{s}^{(k)} + x^{(k)} + \mathsf{M}^{(k)} \lfloor q/2 \rceil, \mathbf{c}_1^* = \mathbf{A}^\top \mathbf{s}^{(k)} + \mathbf{x}^{(k)})$. Now, it may seem that owing to the lossy mode of LWE, we can rely on the entropy of the secret vector $\mathbf{s}^{(k)}$ to argue that c_0^* is distributed uniformly random via the leftover hash lemma. The main difference between the previous single-challenge setting is that we can rely on the entropy of the secret vector $\mathbf{s}^{(k)}$ rather than on the entropy of the secret key $\mathbf{e}_{\mathsf{ID}^*}$. Since each challenge ciphertext is injected with fresh entropy and we can argue statistically that a single challenge ciphertext is not leaking any information on the message, the reduction loss will be independent of the number of challenge ciphertext query N.

Although the above argument may seem correct at first glance, it incurs a subtle but a fatal flaw, thus bringing us to our proposed construction. The problem of the above argument is how we use the leftover hash lemma. To use the lemma correctly, the vector $\mathbf{u}_{\mathsf{ID}^*}$ viewed as a hash function is required to be universal. This is true in case $\mathbf{u}_{\mathsf{ID}^*}$ is set as $\mathbf{A}\mathbf{e}_{\mathsf{ID}^*}$, where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ and $\mathbf{e}_{\mathsf{ID}^*}$ is sampled from some appropriate distribution. However, this is *not* true anymore once we change $\mathbf{A}$ to a lossy matrix $\tilde{\mathbf{A}}$, since $\tilde{\mathbf{A}}$ now lives in an exponentially small subset of $\mathbb{Z}_q^{n \times m}$, hence, we can no longer rely on the entropy of $\mathbf{s}^{(k)}$ to statistically hide the message. To overcome this problem, our final idea is to use the Katz-Wang [KW03] technique. Specifically, we slightly alter the encryption algorithm of GPV-IBE to output the following instead:

$$c_0 = \mathbf{u}_{\mathsf{ID}||0}^\top \mathbf{s} + x_0 + \mathsf{M}\lfloor q/2 \rceil, \quad c_1 = \mathbf{u}_{\mathsf{ID}||1}^\top \mathbf{s} + x_1 + \mathsf{M}\lfloor q/2 \rceil, \quad \text{and} \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x},$$

where $\mathbf{u}_{\mathsf{ID}||b} = H(\mathsf{ID}||b)$ for $b \in \{0, 1\}$. During the security proof, the reduction algorithm sets $\mathbf{u}_{\mathsf{ID}||0}$ and $\mathbf{u}_{\mathsf{ID}||1}$ so that one of them is uniformly random over $\mathbb{Z}_q^n$ and the other is constructed as $\mathbf{A}\mathbf{e}_{\mathsf{ID}}$. Then, for the ciphertext c_b corresponding to the uniformly random vector $\mathbf{u}_{\mathsf{ID}||b}$, we can correctly use the leftover hash lemma to argue that c_b statistically hides the message M. By going through one more hybrid argument, we can change both c_0, c_1 into random values that are independent of the message M. Note that instead of naively using the Katz-Wang technique, by reusing the $\mathbf{c}_2$ component, the above GPV-IBE variant only requires one additional element in $\mathbb{Z}_q$ compared to the original GPV-IBE. Furthermore, in the actual construction, we do not require the noise terms x_0, x_1 in c_0, c_1 since we no longer rely on the LWE assumption to change c_0, c_1 into random values. Our construction and security reduction does not depend on the number of challenge ciphertext query N and in particular, can be proven under the polynomial LWE assumption, which is only slightly worse than the single-challenge construction. In addition, due to the same reason as the single-challenge setting, our classical ROM proof can be naturally converted to a QROM proof.

3.2.1 Discussion.

Similar Techniques in Other Works. The idea to simulate GPV-IBE in a way so that we can create exactly one valid secret key for every secret key query is not new. We are aware of few works that are based on this idea. Gentry, Peikert, and Vaikuntanathan [GPV08] mentioned that by using this technique, they can prove the security of the GPV-IBE in the standard model based on a non-standard interactive variant of the LWE assumption. However, they only gave a sketch of the proof and did not give a formal proof. Alwen et al. [ADN+10] use the idea to construct an identity-based hash proof system (IB-HPS) based on the mechanism of GPV-IBE. We note that they assume the modulus q to be super-polynomial. Outside the context of identity-based primitives, Applebaum et al. [ACPS09] and Bourse [BDPMW16] provide an analysis of rerandomizing LWE samples which can be seen as a refinement of the idea mentioned in [GPV08]. [ACPS09] constructs a KDM-secure cryptosystem based on the LWE problem and [BDPMW16] shows a simple method for constructing circuit private fully homomorphic encryption schemes (FHE) based on the lattice-based FHE scheme of Gentry et al. [GSW13]. Both of their analysis only requires the modulus q to be polynomial. In summary, though similar ideas have been used, all of the previous works are irrelevant to tight security or the security in the QROM.

On Running Time of Reductions. In the above overview, we ignore the running time of reductions. Though it seems that the above described reductions run in almost the same time as that of the adversaries, there is a significant blowup by the square of the number of queries due to a subtle problem of simulating random oracles against quantum adversaries. In the classical ROM, when we simulate a random oracle in security proofs, we usually sample a random function in a lazy manner. That is, whenever an adversary queries a point that has not been queried before, a reduction algorithm samples a fresh randomness and assigns it as a hash value for that point. However, this cannot be done in the QROM because an adversary may query a superposition of all the inputs in a single query. Therefore a reduction algorithm has to somehow commit to the hash values of all inputs at the beginning of the simulation.

Zhandry [Zha12b] proved that an adversary that makes Q queries cannot distinguish a random function and a $2Q$-wise independent hash function via quantum oracle accesses. Therefore we can use a $2Q$-wise independent hash to simulate a random oracle. However, if we take this method, the simulator has to evaluate a $2Q$-wise independent hash function for each hash query, and this is the reason why the running time blowups by $\Omega(Q^2)$.

One possible way to avoid this huge blowup is to simulate a random oracle by a PRF secure against quantum accessible adversaries. Since the time needed to evaluate a PRF is some fixed polynomial in the security parameter, the blowup for the running time can be made $Q \cdot \mathsf{poly}(\lambda)$ which is significantly better than $\Omega(Q^2)$. However, in order to use this method, we have to additionally assume the existence of quantumly secure PRFs. Such PRFs can be constructed based on any quantumly-secure one-way function [Zha12a], and thus they exist if the LWE assumption holds against quantum adversaries. However, the reduction for such PRFs are non-tight and thus we cannot rely on them in the context of tight security. Our suggestion is to use a real hash function to implement PRFs and to assume that it is a quantumly secure PRF. We believe this to be a natural assumption if we are willing to idealize a hash function as a random oracle. (See also the discussion in Section 3.3.2.)

3.2.2 Related Work

Schemes in QROM. Boneh et al. [BDF+11] introduced the QROM, and gave security proofs for the GPV-signature [GPV08] and a hybrid variant of the Bellare-Rogaway encryption [BR93] in the QROM. We note that their security proof for the GPV-signature is tight. Zhandry [Zha12b] proved that GPV-IBE and full-domain hash signatures are secure in the QROM. Targhi and Unruh [TU16] proposed variants of Fujisaki-Okamoto transformation and OAEP that are secure in the QROM. Some researchers studied the security of the Fiat-Shamir transform in the QROM [ARU14, Unr15, Unr17]. Unruh [Unr14b] proposed a revocable quantum timed-release encryption scheme in the QROM. Unruh [Unr14a] proposed a position verification scheme in the QROM.

Recently, some researchers studied tight securities in the QROM. Alkim et al. [ABB+17] proved that the signature scheme known as TESLA [BG14] is tightly secure under the LWE assumption. Saito et al. [SXY18] proposed a tightly CCA secure variant of the Bellare-Rogaway encryption. Kiltz et al. [KLS18] gave a tight reduction for the Fiat-Shamir transform in the QROM.

Tightly Secure IBEs. The first tightly secure IBE scheme from lattices in the single challenge setting and in the standard model was proposed by Boyen and Li [BL16]. While the construction is theoretically interesting and elegant, it is very inefficient and requires LWE assumption with super-polynomial approximation factors. As for the construction from bilinear maps, the first tightly secure IBE from standard assumptions in the single challenge setting and in the random oracle model was proposed by Katz and Wang [KW03]. Coron [Cor09] gave a tight reduction for a variant of the original Boneh-Franklin IBE [BF01]. Later, the first realization in the standard model was proposed by Chen and Wee [CW13]. In the subsequent works, it is further extended to the multi-challenge setting [HKS15, AHY15, GDCC16]. They are efficient but are not secure against quantum computers.

3.3 Preparation

3.3.1 Quantum Computation

We briefly give some backgrounds on quantum computation. We refer to [NC00] for more details. A state $|\psi\rangle$ of n qubits is expressed as $\sum_{x\in\{0,1\}^n} \alpha_x |x\rangle \in \mathbb{C}^{2^n}$ where $\{\alpha_x\}_{x\in\{0,1\}^n}$ is a set of complex numbers such that $\sum_{x\in\{0,1\}^n} |\alpha_x|^2 = 1$ and $\{|x\rangle\}_{x\in\{0,1\}^n}$ is an orthonormal basis on $\mathbb{C}^{2^n}$ (which is called a computational basis). If we measure $|\psi\rangle$ in the computational basis, then the outcome is a classical bit string $x \in \{0,1\}^n$ with probability $|\alpha_x|^2$, and the state becomes $|x\rangle$. An evolution of quantum state can be described by a unitary matrix U, which transforms $|x\rangle$ to $U|x\rangle$. A quantum algorithm is composed of quantum evolutions described by unitary matrices and measurements. We also consider a quantum oracle algorithm, which can quantumly access to certain oracles. The running time $\mathsf{Time}(\mathcal{A})$ of a quantum algorithm $\mathcal{A}$ is defined to be the number of universal gates (e.g., Hadamard, phase, CNOT, and $\pi/8$ gates) and measurements required for running $\mathcal{A}$. (An oracle query is counted as a unit time if $\mathcal{A}$ is an oracle algorithm.) Any efficient classical computation can be realized by a quantum computation efficiently. That is, for any function f that is classically computable, there exists a unitary matrix U_f such that $U_f |x, y\rangle = |x, f(x) \oplus y\rangle$, and the number of universal gates to express U_f is linear in the size of a classical circuit that computes f.

Quantum random oracle model. Boneh et al. [BDF+11] introduced the quantum random oracle model (QROM), which is an extension of the usual random oracle model to the quantum setting. Roughly speaking, the QROM is an idealized model where a hash function is idealized to be a quantumly accessible oracle that simulates a random function. More precisely, in security proofs in the QROM, a random function $\mathsf{H} : \mathcal{X} \to \mathcal{Y}$ is uniformly chosen at the beginning of the experiment, and every entity involved in the system is allowed to access to an oracle that is given $\sum_{x,y} \alpha_{x,y} |x, y\rangle$ and returns $\sum_{x,y} \alpha_{x,y} |x, \mathsf{H}(x) \oplus y\rangle$. We denote a quantum algorithm $\mathcal{A}$ that accesses to the oracle defined as above by $\mathcal{A}^{|\mathsf{H}\rangle}$. In the QROM, one query to the random oracle is counted as one unit time. As in the classical case, we can implement two random oracles H_0 and H_1 from one random oracle H by defining $\mathsf{H}_0(x) := \mathsf{H}(0||x)$ and $\mathsf{H}_1(x) := \mathsf{H}(1||x)$. More generally, we can implement n random oracles from one random oracle by using $\lfloor \log n \rfloor$-bit prefix of an input as index of random oracles.

As shown by Zhandry [Zha12b], a quantum random oracle can be simulated by a family of

$2Q$-wise independent hash functions against an adversary that quantumly accesses to the oracle at most Q times. As a result, he obtained the following lemma.

Lemma 3.1. *([Zha12b, Thereom 6.1].) Any quantum algorithm $\mathcal{A}$ making quantum queries to random oracles can be efficiently simulated by a quantum algorithm $\mathcal{B}$, which has the same output distribution, but makes no queries. Especially, if $\mathcal{A}$ makes at most Q queries to a random oracle $\mathsf{H} : \{0,1\}^a \to \{0,1\}^b$, then $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + Q \cdot T_{a,b}^{2Q\text{-wise}}$ where $T_{a,b}^{2Q\text{-wise}}$ denotes the time to evaluate a $2Q$-wise independent hash function from $\{0,1\}^a$ to $\{0,1\}^b$.*

The following lemma was shown by Boneh et al. [BDF$^+$11]. Roughly speaking, this lemma states that if an oracle outputs independent and almost uniform value for all inputs, then it is indistinguishable from a random oracle even with quantum oracle accesses.

Lemma 3.2. *([BDF$^+$11, Lemma 3].) Let $\mathcal{A}$ be a quantum algorithm that makes at most Q oracle queries, and $\mathcal{X}$ and $\mathcal{Y}$ be arrbitrary sets. Let $\mathcal{H}$ be a distribution over $\mathsf{Func}(\mathcal{X},\mathcal{Y})$ such that when we take $\mathsf{H} \xleftarrow{\$} \mathcal{H}$, for each $x \in \mathcal{X}$, $\mathsf{H}(x)$ is identically and independently distributed according to a distribution D whose statistical distance is within ϵ from uniform. Then for any input z. we have*

$$\Delta(\mathcal{A}^{|\mathsf{RF}\rangle}(z), \mathcal{A}^{|\mathsf{H}\rangle}(z)) \leq 4Q^2\sqrt{\epsilon}$$

where $\mathsf{RF} \leftarrow \mathsf{Func}(\mathcal{X},\mathcal{Y})$ and $\mathsf{H} \leftarrow \mathcal{H}$.

3.3.2 Quantum Pseudorandom Function.

We review the definition of quantum-accessible pseudorandom functions (PRFs) [BDF$^+$11].

Definition 3.1 (Quantum-accessible PRF)**.** *We say that a function $F : \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ is a quantum-accessible pseudorandom function if for all PPT adversaries $\mathcal{A}$, its advantage defined below is negligible:*

$$\mathsf{Adv}_{\mathcal{A},F}^{\mathsf{PRF}}(\lambda) := \left| \Pr\left[\mathcal{A}^{|\mathsf{RF}\rangle}(1^\lambda) = 1\right] - \Pr\left[\mathcal{A}^{|F(K,\cdot)\rangle}(1^\lambda) = 1\right] \right|$$

where $\mathsf{RF} \leftarrow \mathsf{Func}(\mathcal{X},\mathcal{Y})$ and $K \leftarrow \mathcal{K}$.

Zhandry [Zha12a] proved that some known constructions of classical PRFs including the tree-based construction [GGM86] and lattice-based construction [BPR12] are also quantum-accessible PRFs. However, these reductions are non-tight, and thus we cannot rely on these results when aiming for tight security. Fortunately, we can use the following lemma which states that we can use a quantum random oracle as a PRF similarly to the classical case.

Lemma 3.3. *([SXY18, Lemma 2.2]) Let ℓ be an integer. Let $\mathsf{H}\colon \{0,1\}^\ell \times \mathcal{X} \to \mathcal{Y}$ and $\mathsf{H}'\colon \mathcal{X} \to \mathcal{Y}$ be two independent random functions. If an unbounded time quantum adversary $\mathcal{A}$ makes a query to H at most Q_H times, then we have*

$$\left|\Pr[\mathcal{A}^{|\mathsf{H}\rangle,|\mathsf{H}(K,\cdot)\rangle}(1^\lambda) = 1 \mid K \leftarrow \{0,1\}^\ell] - \Pr[\mathcal{A}^{|\mathsf{H}\rangle,|\mathsf{H}'\rangle}(1^\lambda) = 1]\right| \leq Q_\mathsf{H} \cdot 2^{\frac{-\ell+1}{2}}.$$

Remark 3.1. *(Using PRFs to make IBE stateless.) We say that an IBE scheme is stateful if the key generation algorithm has to record all previously issued secret keys, and always outputs the same secret key for the same identity. By the technique by Goldreich [Gol86], a stateful scheme can be converted to a stateless one (in which the key generation algorithm need not remember previous executions) by using PRFs. Since PRFs exist in the QROM without assuming any computational assumption as shown in Lemma 3.3, if we make the key size of PRFs sufficiently large, this conversion hardly affects the tightness. Therefore in this chapter, we concentrate on constructing tightly secure stateful IBE scheme for simplicity.*

3.3.3 Hardness Assumption

In Section 2.2.3 we defined the LWE problem for classical adversaries. Below, we define the LWE assumption against adversaries that can access to a quantum random oracle as is done by Boneh et al. [BDF+11].

Definition 3.2 (Learning with Errors relative to Quantum Random Oracle)**.** *Let n, m, q and χ be the same as in Lemma 2.1, and a, b be some positive integers. For a PPT algorithm $\mathcal{A}$, the advantage for the* learning with errors problem $\mathsf{LWE}_{n,m,q,\chi}$ *of $\mathcal{A}$ relative to a quantum random oracle is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{QRO}_{a,b}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda) = \left| \Pr\left[\mathcal{A}^{|\mathsf{H}\rangle}(\mathbf{A}, \mathbf{A}^{\top}\mathbf{s} + \mathbf{z}) = 1\right] - \Pr\left[\mathcal{A}^{|\mathsf{H}\rangle}(\mathbf{A}, \mathbf{w} + \mathbf{z}) = 1\right] \right|$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{w} \leftarrow \mathbb{Z}_q^m$, $\mathbf{z} \leftarrow \chi^m$, $\mathsf{H} \overset{\$}{\leftarrow} \mathsf{Func}(\{0,1\}^a, \{0,1\}^b)$. We say that the LWE *assumption relative to an (a,b)-quantum random oracle holds if $\mathsf{Adv}_{\mathcal{A},\mathsf{QRO}_{a,b}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda)$ is negligible for all PPT $\mathcal{A}$.*

It is easy to see that the LWE assumption relative to a quantum random oracle can be reduced to the LWE assumption with a certain loss of the time for the reduction by Lemma 3.1. Alternatively, if we assume the existence of a quantumly-accessible PRF, then the reduction loss can be made smaller. Namely, we have the following lemmas.

Lemma 3.4. *For any n, m, q, χ, a, b, and an algorithm $\mathcal{A}$ making at most Q oracle queries, there exists an algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{QRO}_{a,b}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda) = \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + Q \cdot T_{a,b}^{2Q\text{-wise}}$ where $T_{a,b}^{2Q\text{-wise}}$ denotes the time to evaluate a $2Q$-wise independent hash function from $\{0,1\}^a$ to $\{0,1\}^b$.

Lemma 3.5. *Let $F : \mathcal{K} \times \{0,1\}^a \to \{0,1\}^b$ be a quantumly-accessible PRF. For any n, m, q, χ, a, b and an algorithm $\mathcal{A}$ making at most Q oracle queries, there exist algorithms $\mathcal{B}$ and $\mathcal{C}$ such that*

$$\mathsf{Adv}_{\mathcal{A},\mathsf{QRO}_{a,b}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda) \le \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m,q,\chi}}(\lambda) + \mathsf{Adv}_{\mathcal{C},F}^{\mathsf{PRF}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A}) + Q \cdot T_F$ and $\mathsf{Time}(\mathcal{C}) \approx \mathsf{Time}(\mathcal{A})$ where T_F denotes the time to evaluate F.

In this chapter, we give reductions from the security of IBE schemes to the LWE assumption relative to a quantum random oracle. Given such reductions, we can also reduce them to the LWE assumption or to the LWE assumption plus the security of quantumly-accessible PRFs by Lemma 3.4 or 3.5, respectively. The latter is tighter than the former at the cost of assuming the existence of quantumly-accessible PRFs.

Remark 3.2. *A keen reader may wonder why we have to require the extra assumption on the existence of PRFs when we're working in the QROM, since as we mentioned earlier in Remark 3.3.2, it seems that we can use a QRO as a PRF. The point here is that during the security reduction, the simulator (which is given the classical LWE instance) must simulate the QRO query to the adversary against the LWE problem relative to a quantum random oracle query, hence, the simulator is not in possession of the QRO. Note that the reason why we are able to use the QRO as a PRF as mentioned in Remark 3.1 is because the simulator is aiming to reduce the LWE problem relative to a quantum random oracle query to the IBE scheme. Specifically, in this case the simulator can use the QRO provided by its challenge to simulate a PRF.*

3.4 Tightly Secure Single Challenge GPV-IBE

In this section, we show that we can give a tight security proof for the original GPV-IBE [GPV08] in the single-challenge setting if we set the parameters appropriately. Such proofs can be given in both the classical ROM and QROM settings.

3.4.1 Construction

Let the identity space $\mathcal{ID}$ of the scheme be $\mathcal{ID} = \{0,1\}^{\ell_{\mathsf{ID}}}$, where $\ell_{\mathsf{ID}}(\lambda)$ denotes the identity-length. Let also $\mathsf{H} : \{0,1\}^{\ell_{\mathsf{ID}}} \to \mathbb{Z}_q^n$ be a hash function treated as a random oracle during security analysis. The IBE scheme GPV is given as follows. For simplicity, we describe the scheme as a stateful one. As remarked in Remark 3.1, we can make the scheme stateless without any additional assumption in the QROM.

$\mathsf{Setup}(1^\lambda)$: On input 1^λ, it first chooses a prime q, positive integers n, m, and Gaussian parameters α', σ, where all these values are implicitly a function of the security parameter λ. The precise parameter selection is specified in the following section. It then runs $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ to generate a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ with a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ such that $\|\mathbf{T_A}\|_{\text{GS}} \le O(n \log q)$. Then it outputs

$$\mathsf{mpk} = \mathbf{A} \quad \text{and} \quad \mathsf{msk} = \mathbf{T_A}$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: If $\mathsf{sk}_{\mathsf{ID}}$ is already generated, then this algorithm returns it. Otherwise it computes $\mathbf{u}_{\mathsf{ID}} = \mathsf{H}(\mathsf{ID})$ and samples $\mathbf{e}_{\mathsf{ID}} \in \mathbb{Z}^m$ such that

$$\mathbf{A}\mathbf{e}_{\mathsf{ID}} = \mathbf{u}_{\mathsf{ID}} \mod q$$

using $\mathbf{e}_{\mathsf{ID}} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}}, \sigma)$. It returns $\mathsf{sk}_{\mathsf{ID}} = \mathbf{e}_{\mathsf{ID}}$ as the secret key.

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}$, it first samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{x} \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ and $x \leftarrow D_{\mathbb{Z},\alpha' q}$. Then it sets $\mathbf{u}_{\mathsf{ID}} = \mathsf{H}(\mathsf{ID})$ and computes

$$c_0 = \mathbf{u}_{\mathsf{ID}}^\top \mathbf{s} + x + \mathsf{M}\lfloor q/2 \rceil, \quad \mathbf{c}_1 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

Finally, it outputs the ciphertext $C = (c_0, \mathbf{c}_1) \in \mathbb{Z}_q \times \mathbb{Z}_q^m$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C)$: To decrypt a ciphertext $C = (c_0, \mathbf{c}_1)$ with a secret key $\mathsf{sk}_{\mathsf{ID}}$, it computes $w = c_0 - \mathbf{c}_1^\top \mathbf{e}_{\mathsf{ID}} \in \mathbb{Z}_q$ and outputs 0 if w is closer to 0 than to $\lfloor q/2 \rceil$ modulo q. Otherwise it outputs 1.

3.4.2 Correctness and Prameter Selection

The following shows correctness of the above IBE scheme.

Lemma 3.6 (Correctness). *Suppose the parameters q, σ, and α' are such that*

$$\sigma > \|\mathbf{T_A}\|_{\text{GS}} \cdot \sqrt{\log(2m+4)/\pi}, \qquad \alpha' < 1/8\sigma m.$$

Let $\mathbf{e}_{\mathsf{ID}} \leftarrow \mathsf{KeyGen}(\mathbf{A}, \mathbf{T_A}, \mathsf{ID}), C \leftarrow \mathsf{Enc}(\mathbf{A}, \mathsf{ID}', \mathsf{M} \in \{0,1\})$ and $\mathsf{M}' \leftarrow \mathsf{Dec}(\mathbf{A}, \mathbf{e}_{\mathsf{ID}}, C)$. If $\mathsf{ID} = \mathsf{ID}'$, then with overwhelming probability we have $\mathsf{M}' = \mathsf{M}$.

Proof. When the Dec algorithm operates as specified, we have

$$w = c_0 - \mathbf{e}_{\mathsf{ID}}^\top \mathbf{c}_1 = \mathsf{M}\lfloor q/2 \rceil + \underbrace{x + \mathbf{e}_{\mathsf{ID}}^\top \mathbf{x}}_{\text{error term}}.$$

By Lemma 2.12 and the condition posed on the choice of σ, we have that the distribution of $\mathbf{e}_{\mathsf{ID}}$ is $2^{-\Omega(n)}$ close to $D_{\Lambda_{\mathbf{u}}^\perp(\mathbf{A}),\sigma}$. Therefore, by Lemma 2.4, we have $x \le \alpha' q\sqrt{m}$, $\|\mathbf{x}\| \le \alpha' q\sqrt{m}$, and $\|\mathbf{e}_{\mathsf{ID}}\| \le \sigma \cdot \sqrt{m}$ except for $2^{-\Omega(n)}$ probability. Then, the error term is bounded by

$$|\mathbf{h}^\top \mathbf{x} - \mathbf{e}_{\mathsf{ID}}^\top \mathbf{x}| \le x + |\mathbf{e}_{\mathsf{ID}}^\top \mathbf{x}| \le 2\alpha' q \sigma m.$$

Hence, for the error term to have absolute value less than $q/4$, it suffices to choose q and α' as in the statement of the lemma. □

Parameter Selection. For the system to satisfy correctness and make the security proof work, we need the following restrictions. Note that we will prove the security of the scheme under the LWE assumption whose noise rate is α, which is lower than α' that is used in the encryption algorithm.

- The error term is less than $q/4$ (i.e., $\alpha' < 1/8m\sigma$ by Lemma 3.6)
- $\mathsf{TrapGen}$ operates properly (i.e., $m > 3n \log q$ by Lemma 2.12)
- Samplable from $D_{\Lambda_{\mathbf{u}}^\perp(\mathbf{A}),\sigma}$ (i.e., $\sigma > \|\mathbf{T_A}\|_{\text{GS}} \cdot \sqrt{\log(2m+4)/\pi} = O(\sqrt{n \log m \log q})$ by Lemma 2.12),
- σ is sufficiently large so that we can apply Lemma 2.1 and 2.5 (i.e., $\sigma > \sqrt{n + \log m}$, $16\sqrt{\log 2m/\pi}$),
- We can apply Lemma 2.6 (i.e., $\alpha'/2\alpha > \sqrt{n(\sigma^2 m + 1)}$),
- $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ is hard (i.e., $\alpha q > 2\sqrt{n}$).

To satisfy these requirements, for example, we can set the parameters $m, q, \sigma, \alpha, \alpha'$ as follows:

$$m = n^{1+\kappa}, \qquad q = 10n^{3.5+4\kappa}, \qquad \sigma = n^{0.5+\kappa},$$
$$\alpha' q = n^{2+2\kappa}, \qquad \alpha q = 2\sqrt{n},$$

where $\kappa > 0$ is a constant that can be set arbitrarily small. To withstand attacks running in time 2^λ, we may set $n = \tilde{\Omega}(\lambda)$. In the above, we round up m to the nearest integer and q to the nearest largest prime. We remark that though the above parameter is worse compared to the original GPV-IBE scheme, this is due to our conservative choice of making the statistical error terms appearing in the reduction cost $2^{-\Omega(n)}$ rather than the standard negligible notion $2^{-\omega(\log \lambda)}$. The latter choice of parameters will lead to better parameters, which may be as efficient as the original GPV-IBE.

3.4.3 Security Proof in ROM

The following theorem addresses the security of GPV in the classical ROM setting. Our analysis departs from the original one [GPV08] and as a consequence much tighter.

Theorem 3.1. *The IBE scheme* GPV *is adaptively-anonymous single-challenge secure in the random oracle model assuming the hardness of* $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$. *Namely, for any classical adversary*

$\mathcal{A}$ making at most Q_{H} random oracle queries to H and Q_{ID} secret key queries, there exists an algorithm $\mathcal{B}$ such that

$$\mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}}(\lambda) \leq \mathsf{Adv}^{\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B}}(\lambda) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}}) \cdot 2^{-\Omega(n)}$$

and

$$\mathsf{Time}(\mathcal{B}) = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}}) \cdot \mathsf{poly}(\lambda).$$

Proof of Theorem 3.1. Let $\mathsf{CTSam}(\mathsf{mpk})$ be an algorithm that outputs a random element from $\mathbb{Z}_q \times \mathbb{Z}_q^m$ and $\mathcal{A}$ be a classical PPT adversary that attacks the adaptively-anonymous security of the IBE scheme. Without loss of generality, we make some simplifying assumptions on $\mathcal{A}$. First, we assume that whenever $\mathcal{A}$ queries a secret key or asks for a challenge ciphertext, the corresponding ID has already been queried to the random oracle H. Second, we assume that $\mathcal{A}$ makes the same query for the same random oracle at most once. Third, we assume that $\mathcal{A}$ does not repeat secret key queries for the same identity more than once. We show the security of the scheme via the following games. In each game, we define X_i as the event that the adversary $\mathcal{A}$ wins in Game_i.

Game_0 : This is the real security game. At the beginning of the game, $(\mathbf{A}, \mathbf{T_A}) \overset{\$}{\leftarrow} \mathsf{TrapGen}(1^n, 1^m, q)$ is run and the adversary $\mathcal{A}$ is given $\mathbf{A}$. The challenger then samples $\mathsf{coin} \overset{\$}{\leftarrow} \{0,1\}$ and keeps it secret. During the game, $\mathcal{A}$ may make random oracle queries, secret key queries, and the challenge query. These queries are handled as follows:

- When $\mathcal{A}$ makes a random oracle query to H on ID, the challenger chooses a random vector $\mathbf{u}_{\mathsf{ID}} \leftarrow \mathbb{Z}_q^n$ and locally stores the tuple $(\mathsf{ID}, \mathbf{u}_{\mathsf{ID}}, \perp)$, and returns $\mathbf{u}_{\mathsf{ID}}$ to $\mathcal{A}$.
- When the adversary $\mathcal{A}$ queries a secret key for ID, the challenger computes $\mathbf{e}_{\mathsf{ID}} = \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}}, \sigma)$ and returns $\mathbf{e}_{\mathsf{ID}}$ to $\mathcal{A}$.
- When the adversary makes the challenge query for ID^* and a message M^*, the challenger returns $(c_0, \mathbf{c}_1) \overset{\$}{\leftarrow} \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$ if $\mathsf{coin} = 0$ and $(c_0, \mathbf{c}_1) \overset{\$}{\leftarrow} \mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$.

At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger outputs $\widehat{\mathsf{coin}}$. By definition, we have $\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} - \mathsf{coin}] - \frac{1}{2}\right| = \mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}}(\lambda)$.

Game_1 : In this game, we change the way the random oracle queries to H are answered. When $\mathcal{A}$ queries the random oracle H on ID, the challenger generates a pair $(\mathbf{u}_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}})$ by first sampling $\mathbf{e}_{\mathsf{ID}} \leftarrow \mathcal{D}_{\mathbb{Z}^m,\sigma}$ and setting $\mathbf{u}_{\mathsf{ID}} = \mathbf{A}\mathbf{e}_{\mathsf{ID}}$. Then it locally stores the tuple $(\mathsf{ID}, \mathbf{u}_{\mathsf{ID}}, \perp)$, and returns $\mathbf{u}_{\mathsf{ID}}$ to $\mathcal{A}$. Here, we remark that when $\mathcal{A}$ makes a secret key query for ID, the challenger returns $\mathbf{e}'_{\mathsf{ID}} \overset{\$}{\leftarrow} \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}}, \sigma)$, which is independent from $\mathbf{e}_{\mathsf{ID}}$ that was generated in the simulation of the random oracle H on input ID. Note that in this game, we only change the distribution of $\mathbf{u}_{\mathsf{ID}}$ for each identity. Due to Lemma 2.1, the distribution of $\mathbf{u}_{\mathsf{ID}}$ in Game_2 is $2^{-\Omega(n)}$-close to that of Game_1 except for $2^{-\Omega(n)}$ fraction of $\mathbf{A}$ since we choose $\sigma > \sqrt{n + \log m}$. Therefore, the statistical distance between the view of $\mathcal{A}$ in Game_1 and Game_2 is $2^{-\Omega(n)} + Q_{\mathsf{H}} \cdot 2^{-\Omega(n)} < Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$. Therefore, we have $\left|\Pr[X_1] - \Pr[X_2]\right| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Game_2 : In this game, we change the way secret key queries are answered. By the end of this game, the challenger will no longer require the trapdoor $\mathbf{T_A}$ to generate the secret keys. When $\mathcal{A}$ queries the random oracle on ID, the challenger generates a pair $(\mathbf{u}_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}})$ as in the previous game. Then it locally stores the tuple $(\mathsf{ID}, \mathbf{u}_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}})$ and returns $\mathbf{u}_{\mathsf{ID}}$ to $\mathcal{A}$. When $\mathcal{A}$ queries a secret key for ID, the challenger retrieves the unique tuple $(\mathsf{ID}, \mathbf{u}_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}})$ from local storage and returns $\mathbf{e}_{\mathsf{ID}}$. For any fixed $\mathbf{u}_{\mathsf{ID}} \in \mathbb{Z}_q^n$, let $\mathbf{e}^{(1)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}$ and $\mathbf{e}^{(2)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}$ be random variables that are distributed according to the distributions of $\mathsf{sk}_{\mathsf{ID}}$ conditioning on $H(\mathsf{ID}) = \mathbf{u}_{\mathsf{ID}}$ in Game_1 and Game_2, respectively. Due to Lemma 2.12, we have $\Delta(\mathbf{e}^{(1)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. On the other hand, due to Lemma 2.1, we have $\Delta(\mathbf{e}^{(2)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. Since $\mathcal{A}$ obtains at most Q_{ID} user secret keys $\mathsf{sk}_{\mathsf{ID}}$, we have $\left|\Pr[X_1] - \Pr[X_2]\right| = Q_{\mathsf{ID}} \cdot 2^{-\Omega(n)}$.

Game_3 : In this game, we change the way the matrix $\mathbf{A}$ is generated. Concretely, the challenger chooses $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ without generating the associated trapdoor $\mathbf{T}_{\mathbf{A}}$. By Lemma 2.12, this makes only $2^{-\Omega(n)}$-statistical difference. Since the challenger can answer all the secret key queries without the trapdoor due to the change we made in the previous game, the view of $\mathcal{A}$ is altered only negligibly. Therefore, we have $\left|\Pr[X_2] - \Pr[X_3]\right| = 2^{-\Omega(n)}$.

Game_4 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. Recall in the previous games when $\mathsf{coin} = 0$, the challenger created a valid challenge ciphertext as in the real scheme. In this game, to create the challenge ciphertext for identity ID^* and message bit M^*, the challenger first retrieves the unique tuple $(\mathsf{ID}^*, \mathbf{u}_{\mathsf{ID}^*}, \mathbf{e}_{\mathsf{ID}^*})$ from local storage. Then the challenger picks $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $\mathbf{v} = \mathbf{A}^{\top}\mathbf{s} + \bar{\mathbf{x}} \in \mathbb{Z}_q^m$. It then runs

$$\mathsf{ReRand}([\mathbf{e}_{\mathsf{ID}^*}|\mathbf{I}_m], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}) \to \mathbf{c}' \in \mathbb{Z}_q^{m+1}$$

from Lemma 2.6, where $\mathbf{I}_m$ is the identity matrix with size m. Let $c'_0 \in \mathbb{Z}_q$ denote the first entry of $\mathbf{c}'$ and $\mathbf{c}_1 \in \mathbb{Z}_q^m$ denote the remaining entries of $\mathbf{c}'$. Finally, the challenger outputs the challenge ciphertext as

$$C^* = (c_0 = c'_0 + \mathsf{M}^*\lfloor q/2\rceil, \quad \mathbf{c}_1). \tag{3.1}$$

We now proceed to bound $|\Pr[X_3] - \Pr[X_4]|$. We apply the noise rerandomization lemma (Lemma 2.6) with $\mathbf{V} = [\mathbf{e}_{\mathsf{ID}^*}|\mathbf{I}_m]$, $\mathbf{b} = \mathbf{A}^{\top}\mathbf{s}$ and $\mathbf{z} = \bar{\mathbf{x}}$ to see that the distribution of $\mathbf{c}'$ is negligibly close to the following:

$$\mathbf{c}' = \mathbf{V}^{\top}\mathbf{b} + \mathbf{x}' = \Big(\mathbf{A} \cdot [\mathbf{e}_{\mathsf{ID}^*}|\mathbf{I}_m]\Big)^{\top}\mathbf{s} + \mathbf{x}' = [\mathbf{u}_{\mathsf{ID}^*}|\mathbf{A}]^{\top}\mathbf{s} + \mathbf{x}'$$

where the distribution of $\mathbf{x}'$ is $2^{-\Omega(n)}$-close to $D_{\mathbb{Z}^{m+1},\alpha' q}$. Here, the last equality follows from $\mathbf{A}\mathbf{e}_{\mathsf{ID}^*} = \mathbf{u}_{\mathsf{ID}^*}$ and we can appropriately apply the noise rerandomization lemma since we have the following for our parameter selection:

$$\alpha'/2\alpha > \sqrt{n(\sigma^2 m + 1)} \geq \sqrt{n(\|\mathbf{e}_{\mathsf{ID}^*}\|^2 + 1)} \geq \sqrt{n} \cdot s_1([\mathbf{e}_{\mathsf{ID}^*}|\mathbf{I}_m]),$$

where the second inequality holds with $1 - 2^{-\Omega(n)}$ probability. It can be seen that the challenge ciphertext is distributed statistically close to that in Game_3. Therefore, we may conclude that $\left|\Pr[X_3] - \Pr[X_4]\right| = 2^{-\Omega(n)}$.

Game_5 : In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext the challenger first picks $\mathbf{b} \leftarrow \mathbb{Z}_q^m$, $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $\mathbf{v} = \mathbf{b} + \bar{\mathbf{x}} \in \mathbb{Z}_q^m$. It then sets $\mathbf{V} = [\mathbf{e}_{\mathsf{ID}^*}|\mathbf{I}_m]$ and runs the ReRand algorithm as in Game_3. Finally, it sets the challenge ciphertext as in Eq. (3.1). We claim that $\left|\Pr[X_4] - \Pr[X_5]\right|$ is negligible assuming the hardness of the $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ problem. To show this, we use $\mathcal{A}$ to construct an LWE adversary $\mathcal{B}$ as follows:

$\mathcal{B}$ is given a problem instance of LWE as $(\mathbf{A}, \mathbf{v} = \mathbf{b} + \bar{\mathbf{x}}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m$ where $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m,\alpha q}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{b} = \mathbf{A}^\top \mathbf{s}$ for some $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ or $\mathbf{b} \leftarrow \mathbb{Z}_q^m$. $\mathbf{B}$ sets the master public key mpk to be the LWE matrix $\mathbf{A}$. Note that unlike the real IBE scheme, $\mathbf{B}$ does not require the master secret key $\mathbf{T}_\mathbf{A}$ due to the modification we made in Game_3. To generate the challenge ciphertext, $\mathcal{B}$ first picks $\mathsf{coin} \leftarrow \{0, 1\}$. If $\mathsf{coin} = 0$, it generates the challenge ciphertext as in Eq. (3.1) using $\mathbf{v}$, and returns it to $\mathbf{A}$. We emphasize that all $\mathcal{B}$ needs to do to generate the ciphertext is to run the ReRand algorithm, which it can do without the knowledge of the secret randomness $\mathbf{s}$ and $\bar{\mathbf{x}}$. If $\mathsf{coin} = 1$, $\mathcal{B}$ returns a random ciphertext using $\mathsf{CTSam}(\mathsf{mpk})$. At the end of the game, $\mathcal{A}$ outputs $\widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\widehat{\mathsf{coin}} = \mathsf{coin}$ and 0 otherwise. It can be seen that if $\mathbf{A}, \mathbf{v}$ is a valid LWE sample (i.e., $\mathbf{v} = \mathbf{A}^\top \mathbf{s}$), the view of the adversary corresponds to Game_4. Otherwise (i.e., $\mathbf{v} \leftarrow \mathbb{Z}_q^m$), it corresponds to Game_5. We therefore conclude that assuming the hardness of $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ problem we have $\big|\Pr[X_4] - \Pr[X_5]\big| = \mathsf{negl}$.

Game_6 : In this game, we change the way the challenge ciphertext is created once more. If $\mathsf{coin} = 0$, to create the challenge ciphertext the challenger first picks $\mathbf{b} \leftarrow \mathbb{Z}_q^m$, $\bar{\mathbf{x}}' \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ and computes

$$\mathbf{c}' = [\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m]^\top \mathbf{b} + \mathbf{x}'.$$

It then parses $\mathbf{c}'$ into c_0' and $\mathbf{c}_1$ (as in Game_4) and sets the challenge ciphertext as Eq. (3.1). Similarly to the change from Game_3 to Game_4, we have $\big|\Pr[X_5] - \Pr[X_6]\big| = 2^{-\Omega(n)}$ by Lemma 2.6.

It remains to show that no adversary has negligible chance in winning Game_6. Notice that when $\mathsf{coin} = 0$, the challenge ciphertext can be written as

$$c_0 = \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b} + x_0' + \mathsf{M}\lfloor q/2 \rceil, \quad \mathbf{c}_1 = \mathbf{b} + \mathbf{x}_1',$$

where x_0' is the first entry of $\mathbf{x}'$ and $\mathbf{x}_1'$ is the remaining entries. It suffices to show that the joint distribution of $(\mathbf{b}, \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b})$ is negligibly close to the uniform distribution over $\mathbb{Z}_q^m \times \mathbb{Z}_q$, conditioned on $\mathbf{u}_{\mathsf{ID}^*}$. From the view of $\mathcal{A}$, $\mathbf{e}_{\mathsf{ID}^*}$ is distribute as $D_{\Lambda^\perp_{\mathbf{u}_{\mathsf{ID}^*}}(\mathbf{A}),\sigma}$. By Lemma 2.5, we have

$$\mathbf{H}_\infty(\mathbf{e}_{\mathsf{ID}^*}) \geq m - 1$$

for all but $2^{-\Omega(n)}$ fraction of $\mathbf{A}$. Now we can apply the leftover hash lemma since $\mathbf{b}$ is distributed uniformly at random over $\mathbb{Z}_q^m$ and conclude that $(\mathbf{b}, \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b})$ is $\sqrt{q/2^{m-1}}$-close to the uniform distribution. Hence, we have $\Pr[X_6] \leq 2^{-\Omega(n)} + \sqrt{q/2^{m-1}} < 2^{-\Omega(n)}$.

Therefore, combining everything together, the theorem is proven. $\square$

3.4.4 Security Proof in QROM

As we explained in the introduction, our analysis in the ROM can be easily be extended to the QROM setting. We can prove the following theorem that addresses the security of the GPV-IBE scheme in the QROM setting. The analysis here is different from that by Zhandry [Zha12b], who gave the first security proof for the GPV-IBE scheme in the QROM setting and our analysis here is much tighter.

Theorem 3.2. *The IBE scheme* GPV *is adaptively-anonymous single-challenge secure assuming the hardness of* $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ *in the quantum random oracle model. Namely, for any quantum adversary* $\mathcal{A}$ *making at most* Q_H *queries to* $|\mathsf{H}\rangle$ *and* Q_{ID} *secret key queries, there exists a quantum algorithm* $\mathcal{B}$ *making* $Q_\mathsf{H} + Q_{\mathsf{ID}}$ *quantum random oracle queries such that*

$$\mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}}(\lambda) \leq \mathsf{Adv}^{\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B},\mathsf{QRO}_{\ell_{\mathsf{ID}},\ell r}}(\lambda) + (Q_\mathsf{H}^2 + Q_{\mathsf{ID}}) \cdot 2^{-\Omega(n)}$$

and

$$\mathsf{Time}(\mathcal{B}) = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}}) \cdot \mathsf{poly}(\lambda)$$

where ℓ_r denotes the length of the randomness for $\mathsf{Sample}\mathbb{Z}$.

Proof of Theorem 3.2. Let $\mathsf{CTSam}(\mathsf{mpk})$ be an algorithm that outputs a random element from $\mathbb{Z}_q \times \mathbb{Z}_q^m$ and $\mathcal{A}$ be a quantum adversary that attacks the adaptively-anonymous security of the IBE scheme. Without loss of generality, we can assume that $\mathcal{A}$ makes secret key queries on the same identity at most once. We show the security of the scheme via the following games. In each game, we define X_i as the event that the adversary $\mathcal{A}$ wins in Game_i.

Game_0 : This is the real security game for the adaptively-anonymous security. At the beginning of the game, the challenger chooses a random function $\mathsf{H} : \{0,1\}^{\ell_{\mathsf{ID}}} \to \mathbb{Z}_q^n$. Then it generates $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$ and gives $\mathbf{A}$ to $\mathcal{A}$. Then it samples $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and keeps it secret. During the game, $\mathcal{A}$ may make (quantum) random oracle queries, secret key queries, and a challenge query. These queries are handled as follows:

- When $\mathcal{A}$ makes a random oracle query on a quantum state $\sum_{\mathsf{ID},y} \alpha_{\mathsf{ID},y} |\mathsf{ID}\rangle |y\rangle$, the challenger returns $\sum_{\mathsf{ID},y} \alpha_{\mathsf{ID},y} |\mathsf{ID}\rangle |\mathsf{H}(\mathsf{ID}) \oplus y\rangle$.
- When $\mathcal{A}$ makes a secret key query on ID, the challenger samples $\mathbf{e}_{\mathsf{ID}} = \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}_{\mathsf{ID}}, \sigma)$ and returns $\mathbf{e}_{\mathsf{ID}}$ to $\mathcal{A}$.
- When $\mathcal{A}$ makes a challenge query for ID^* and a message M^*, the challenger returns $(c_0, \mathbf{c}_1) \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$ if $\mathsf{coin} = 0$ and $(c_0, \mathbf{c}_1) \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$.

At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger outputs $\widehat{\mathsf{coin}}$. By definition, we have $\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} - \mathsf{coin}] - \frac{1}{2}\right| = \mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}}(\lambda)$.

Game_1 : In this game, we change the way the random oracle H is simulated. Namely, the challenger first chooses another random function $\widehat{\mathsf{H}} \xleftarrow{\$} \mathsf{Func}(\{0,1\}^{\ell_{\mathsf{ID}}}, \{0,1\}^{\ell_r})$. Then we define $\mathsf{H}(\mathsf{ID}) := \mathbf{A}\mathbf{e}_{\mathsf{ID}}$ where $\mathbf{e}_{\mathsf{ID}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}))$, and use this H throughout the game. For any fixed ID, the distribution of $\mathsf{H}(\mathsf{ID})$ is identical and its statistical distance from the uniform distribution is $2^{-\Omega(n)}$ for all but $2^{-\Omega(n)}$ fraction of $\mathbf{A}$ due to Lemma 2.1 since we choose $\sigma > \sqrt{n + \log m}$. Note that in this game, we only change the distribution of $\mathbf{u}_{\mathsf{ID}}$ for each identity, and the way we create secret keys are unchanged. Then due to Lemma 3.2, we have $\left|\Pr[X_0] - \Pr[X_1]\right| = 2^{-\Omega(n)} + 4Q_{\mathsf{H}}^2\sqrt{2^{-\Omega(n)}} = Q_{\mathsf{H}}^2 \cdot 2^{-\Omega(n)}$.

Game_2 : In this game, we change the way secret key queries are answered. By the end of this game, the challenger will no longer require the trapdoor $\mathbf{T}_{\mathbf{A}}$ to generate the secret keys. When $\mathcal{A}$ queries a secret key for ID, the challenger returns $\mathbf{e}_{\mathsf{ID}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}))$. For any fixed $\mathbf{u}_{\mathsf{ID}} \in \mathbb{Z}_q^n$, let $\mathbf{e}^{(1)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}$ and $\mathbf{e}^{(2)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}$ be random variables that are distributed according to the distributions of $\mathbf{e}_{\mathsf{ID}}$ conditioning on $\mathsf{H}(\mathsf{ID}) = \mathbf{u}_{\mathsf{ID}}$ in Game_1 and Game_2, respectively. Due to Lemma 2.12, we have $\Delta(\mathbf{e}^{(1)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. On the other hand, due to Lemma 2.1, we have $\Delta(\mathbf{e}^{(2)}_{\mathsf{ID},\mathbf{u}_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. Since $\mathcal{A}$ obtains at most Q_{ID} user secret keys $\mathbf{e}_{\mathsf{ID}}$, we have $\left|\Pr[X_1] - \Pr[X_2]\right| = Q_{\mathsf{ID}} \cdot 2^{-\Omega(n)}$.

Game_3 : In this game, we change the way the matrix $\mathbf{A}$ is generated. Concretely, the challenger chooses $\mathbf{A} \leftarrow \mathbb{Z}_q^{n \times m}$ without generating the associated trapdoor $\mathbf{T}_{\mathbf{A}}$. By Lemma 2.12, the distribution of $\mathbf{A}$ differs at most by $2^{-\Omega(n)}$. Since the challenger can answer all the secret key queries without the trapdoor due to the change we made in the previous game, the view of $\mathcal{A}$ is altered only by $2^{-\Omega(n)}$. Therefore, we have $\left|\Pr[X_2] - \Pr[X_3]\right| = 2^{-\Omega(n)}$.

Game_4 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. Recall in the previous games when $\mathsf{coin} = 0$, the challenger created a valid challenge ciphertext as in the real scheme. In this game, to create the challenge ciphertext for identity ID^* and message bit M^*, the challenger first computes $\mathbf{e}_{\mathsf{ID}^*} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}^*))$ and $\mathbf{u}_{\mathsf{ID}^*} := \mathbf{A}\mathbf{e}_{\mathsf{ID}^*}$. Then the challenger picks $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m, \alpha q}$ and computes $\mathbf{v} = \mathbf{A}^\top \mathbf{s} + \bar{\mathbf{x}} \in \mathbb{Z}_q^m$. It then runs

$$\mathsf{ReRand}([\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}) \rightarrow \mathbf{c}' \in \mathbb{Z}_q^{m+1}$$

from Lemma 2.6, where $\mathbf{I}_m$ is the identity matrix with size m. Let $c_0' \in \mathbb{Z}_q$ denote the first entry of $\mathbf{c}'$ and $\mathbf{c}_1 \in \mathbb{Z}_q^m$ denote the remaining entries of $\mathbf{c}'$. Finally, the challenger outputs the challenge ciphertext as

$$C^* = (c_0 = c_0' + \mathsf{M}^* \lfloor q/2 \rceil, \quad \mathbf{c}_1). \tag{3.2}$$

We now proceed to bound $|\Pr[X_3] - \Pr[X_4]|$. We apply the noise rerandomization lemma (Lemma 2.6) with $\mathbf{V} = [\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m]$, $\mathbf{b} = \mathbf{A}^\top \mathbf{s}$ and $\mathbf{z} = \bar{\mathbf{x}}$ to see that the following equation holds:

$$\mathbf{c}' = \mathbf{V}^\top \mathbf{b} + \mathbf{x}' = \Big(\mathbf{A} \cdot [\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m]\Big)^\top \mathbf{s} + \mathbf{x}' = [\mathbf{u}_{\mathsf{ID}^*} | \mathbf{A}]^\top \mathbf{s} + \mathbf{x}'$$

where $\mathbf{x}'$ is distributed according to a distribution whose statistical distance is at most $2^{-\Omega(n)}$ from $D_{\mathbb{Z}^{m+1}, \alpha' q}$. Here, the last equality follows from $\mathbf{A}\mathbf{e}_{\mathsf{ID}^*} = \mathbf{u}_{\mathsf{ID}^*}$ and we can appropriately apply the noise rerandomization lemma since we have the following for our parameter selection:

$$\alpha'/2\alpha > \sqrt{n(\sigma^2 m + 1)} \geq \sqrt{n(\|\mathbf{e}_{\mathsf{ID}^*}\|^2 + 1)} \geq \sqrt{n} \cdot s_1([\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m]),$$

where the second inequality holds with $1 - 2^{-\Omega(n)}$ probability. It therefore follows that the statistical distance between the distributions of the challenge ciphertext in Game_3 and Game_4 is at most $2^{-\Omega(n)}$. Therefore, we may conclude that $\left|\Pr[X_3] - \Pr[X_4]\right| = 2^{-\Omega(n)}$.

Game_5 : In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext the challenger first picks $\mathbf{b} \leftarrow \mathbb{Z}_q^m$, $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m, \alpha q}$ and computes $\mathbf{v} = \mathbf{b} + \bar{\mathbf{x}} \in \mathbb{Z}_q^m$. It then runs the ReRand algorithm as in Game_4. Finally, it sets the challenge ciphertext as in Eq. (3.2). We claim that $\left|\Pr[X_4] - \Pr[X_5]\right|$ is negligible assuming the hardness of the $\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}$ problem relative to a quantum random oracle $|\widehat{\mathsf{H}}\rangle : \{0,1\}^{\ell_{\mathsf{ID}}} \rightarrow \{0,1\}^{\ell_r}$. To show this, we use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that breaks the LWE assumption relative to $|\widehat{\mathsf{H}}\rangle$.

$\mathcal{B}$ is given a problem instance of LWE as $(\mathbf{A}, \mathbf{v} = \mathbf{b} + \bar{\mathbf{x}}) \in \mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^m$ where $\bar{\mathbf{x}} \leftarrow D_{\mathbb{Z}^m, \alpha q}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{b} = \mathbf{A}^\top \mathbf{s}$ for some $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ or $\mathbf{b} \leftarrow \mathbb{Z}_q^m$. First, we remark that $\mathcal{B}$ can simulate the quantum random oracle $|\mathsf{H}\rangle$ for $\mathcal{A}$ by using its own random oracle $|\widehat{\mathsf{H}}\rangle$ because H is programmed as $\mathsf{H}(\mathsf{ID}) := \mathbf{A}\mathbf{e}_{\mathsf{ID}}$ where $\mathbf{e}_{\mathsf{ID}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}))$ by the modification we made in Game_1. $\mathcal{B}$ sets the master public key mpk to be the LWE matrix $\mathbf{A}$. Note that unlike the real IBE scheme, $\mathcal{B}$ does not require the master secret key $\mathbf{T}_{\mathbf{A}}$ due to the modification we made in Game_3. Namely, when $\mathcal{A}$ queries ID for the key oracle, $\mathcal{B}$ just returns $\mathbf{e}_{\mathsf{ID}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}))$. To generate the challenge ciphertext, $\mathcal{B}$ first picks $\mathsf{coin} \leftarrow \{0,1\}$. If $\mathsf{coin} = 0$, it generates the challenge ciphertext as in Eq. (3.2) using $\mathbf{v}$, and returns it to $\mathcal{A}$. We emphasize that all $\mathcal{B}$ needs to do to generate the ciphertext is to run the ReRand algorithm, which it can do without the knowledge of the secret randomness $\mathbf{s}$ and $\bar{\mathbf{x}}$. If $\mathsf{coin} = 1$, $\mathcal{B}$ returns a random ciphertext using $\mathsf{CTSam}(\mathsf{mpk})$. At the end of the game, $\mathcal{A}$ outputs $\widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\widehat{\mathsf{coin}} = \mathsf{coin}$ and 0 otherwise.

It can be seen that if $\mathbf{A}, \mathbf{v}$ is a valid LWE sample (i.e., $\mathbf{v} = \mathbf{A}^\top \mathbf{s}$), the view of the adversary corresponds to Game_4. Otherwise (i.e., $\mathbf{v} \leftarrow \mathbb{Z}_q^m$), it corresponds to Game_5. Therefore we have $\big|\Pr[X_4] - \Pr[X_5]\big| = \mathsf{Adv}_{\mathcal{B},\mathsf{QRO}_{\ell_{\mathsf{ID}},\ell_r}}^{\mathsf{LWE}_{n,m,q,D_{\mathbb{Z},\alpha q}}}(\lambda)$. As for the running time, we have $\mathsf{Time}(\mathcal{B}) = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}}) \cdot \mathsf{poly}(\lambda)$ since all $\mathcal{B}$ has to do is to run $\mathcal{A}$ once plus to compute some additional computations that can be done in a fixed polynomial time whenever $\mathcal{A}$ makes a quantum random oracle or secret key query.

Game_6 : In this game, we further change the way the challenge ciphertext is created. If $\mathsf{coin} = 0$, to create the challenge ciphertext the challenger first picks $\mathbf{b} \leftarrow \mathbb{Z}_q^m$, $\mathbf{x}' \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ and computes

$$\mathbf{c}' = [\mathbf{e}_{\mathsf{ID}^*} | \mathbf{I}_m]^\top \mathbf{b} + \mathbf{x}'.$$

It then parses $\mathbf{c}'$ into c_0' and $\mathbf{c}_1$ (as in Game_4) and sets the challenge ciphertext as Eq. (3.2). Similarly to the change from Game_3 to Game_4, we have $\big|\Pr[X_5] - \Pr[X_6]\big| = 2^{-\Omega(n)}$ by Lemma 2.6.

It remains to show that no adversary has non-negligible chance in winning Game_6. Notice that when $\mathsf{coin} = 0$, the challenge ciphertext can be written as

$$c_0 = \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b} + x_0' + \mathsf{M}\lfloor q/2 \rceil, \quad \mathbf{c}_1 = \mathbf{b} + \mathbf{x}_1',$$

where x_0' is the first entry of $\mathbf{x}'$ and $\mathbf{x}_1'$ is the remaining entries. It suffices to show that the joint distribution of $(\mathbf{b}, \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b})$ is statistically close to the uniform distribution over $\mathbb{Z}_q^m \times \mathbb{Z}_q$, conditioned on $\mathbf{u}_{\mathsf{ID}^*}$. From the view of $\mathcal{A}$, $\mathbf{e}_{\mathsf{ID}^*}$ is distribute as $D_{\Lambda^\perp_{\mathbf{u}(\mathsf{ID}^*)}(\mathbf{A}),\sigma}$ because all information of $\mathbf{e}_{\mathsf{ID}^*}$ revealed to $\mathcal{A}$ is $\mathsf{H}(\mathsf{ID}^*) = \mathbf{A}\mathbf{e}_{\mathsf{ID}^*}$ where $\mathbf{e}_{\mathsf{ID}^*} = \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}(\mathsf{ID}^*))$ and $\widehat{\mathsf{H}}(\mathsf{ID}^*)$ is completely random from the view of $\mathcal{A}$. (Remark that $\widehat{\mathsf{H}}(\mathsf{ID}^*)$ is used in the game only when $\mathcal{A}$ queries ID^* to the key generation oracle, which is prohibited in the adaptively-anonymous security game.) By Lemma 2.5, we have

$$\mathbf{H}_\infty(\mathbf{e}_{\mathsf{ID}^*}) \geq m - 1$$

for all but $2^{-\Omega(n)}$ fraction of $\mathbf{A}$. Now we can apply the leftover hash lemma since $\mathbf{b}$ is distributed uniformly at random over $\mathbb{Z}_q^m$ and conclude that $(\mathbf{b}, \mathbf{e}_{\mathsf{ID}^*}^\top \mathbf{b})$ is $\sqrt{q/2^{m-1}}$-close to the uniform distribution by the leftover hash lemma. Hence, we have $\Pr[X_6] \leq 2^{-\Omega(n)} + \sqrt{q/2^{m-1}} < 2^{-\Omega(n)}$.

Therefore, combining everything together, the theorem is proven. □

3.5 (Almost) Tightly Secure Multi-Challenge IBE

In this section, we propose an IBE scheme that is (almost) tightly secure in the multi-challenge setting. The security of the scheme is proven both in the classical ROM and QROM settings. Our construction is obtained by applying the Katz-Wang [KW03] technique to the original GPV-IBE scheme. Since the proofs require some previous results on random extractions and lossy mode LWE, we first review them below.

3.5.1 Randomness Extraction

We recap some definitions and results on randomness extraction. As we have already introduced in the preliminaries, the min-entropy of a random variable X was defined as $\mathbf{H}_\infty(X) = -\log(\max_x \Pr[X = x])$. A similar notion called the *average min-entropy*, as introduced by Dodis et al. [DORS04], is defined as follows:

$$\tilde{\mathbf{H}}_\infty(X|I) = -\log(\mathbb{E}_{i\leftarrow I}[2^{-H_\infty(X|I=i)}]).$$

The average min-entropy corresponds to the optimal probability of guessing X, given knowledge of I. Min-entropy is a rather fragile notion, since a single high-probability element can ruin the min-entropy of an otherwise good distribution. Therefore, it is often more beneficial to work with the *ϵ-smooth min-entropy* introduced by Renner and Wolf [RW04], which considers all distributions that are ϵ-close to X, but which has higher entropy:

$$\mathbf{H}^\epsilon_\infty(X) = \max_{Y:\ \Delta(X,Y)\le\epsilon} \mathbf{H}_\infty(Y).$$

Similarly, a smooth version of average min-entropy can be defined as follows:

$$\tilde{\mathbf{H}}^\epsilon_\infty(X|I) = \max_{(Y,J):\ \Delta((X,I),(Y,J))\le\epsilon} \tilde{\mathbf{H}}_\infty(Y|J).$$

We recall the definition of universal hash functions and provide an elementary construction of them that will be used in our construction of multi-insatnce secure IBE schemes.

Definition 3.3 (Universal Hash Functions)**.** *A family of functions $\mathcal{H} = \{h : \mathcal{X} \to \mathcal{D}\}_h$ is called a family of universal hash functions, if for all $x, x' \in \mathcal{X}$ with $x \neq x'$, we have $\Pr_{h\leftarrow\mathcal{H}}[h(x) = h(x')] \le \frac{1}{|\mathcal{D}|}$.*

Fact 3.1. *Let $q > 2$. Let $\mathcal{H} = \{\mathbf{u} : \mathbb{Z}_q^n \to \mathbb{Z}_q\}_{\mathbf{u}\in\mathbb{Z}_q^n}$ be a family of hash functions, where $\mathbf{u}(\mathbf{s})$ is defined as $\mathbf{u}(\mathbf{s}) = \mathbf{u}^\top\mathbf{s} \mod q$. Then, $\mathcal{H}$ is a family of universal hash functions.*

The following lemma gives a lower bound for the smooth average min-entropy of some random variable when partial related information is leaked.

Lemma 3.7 ([AKPW13], Lemma 2.4)**.** *Let X, Y, and Z be correlated random variables and $\mathcal{Z}$ be some set such that $\Pr[Z \in \mathcal{Z}] \le \epsilon$ and $|\mathcal{Z}| \le 2^z$. Then, for any $\epsilon' > 0$, $\tilde{\mathbf{H}}^{\epsilon+\epsilon'}_\infty(X|(Y,Z)) \ge \tilde{\mathbf{H}}^{\epsilon'}_\infty(X|Y) - z$.*

The following is a generalization of the leftover hash lemma due to [DORS04]. Here, we provide the smoothed-variant of the lemma. Roughly, this relates to the number of extractable bits that look nearly uniform to the adversary who knows some value that is ϵ-close to the random variable I.

Lemma 3.8 (Generalized Leftover Hash Lemma)**.** *Let $\mathcal{H} = \{h : \mathcal{X} \to \mathcal{D}\}$ be a family of universal hash functions. Let X be an independent random variable with values in $\mathcal{X}$, let I be any random variable. Then, for any $\epsilon \ge 0$, we have*

$$\Delta\big((h, h(X), I), (h, U(\mathcal{D}), I\big) \le 2\epsilon + \frac{1}{2}\cdot\sqrt{2^{-\tilde{\mathbf{H}}^\epsilon_\infty(X|I)}\cdot|\mathcal{D}|}.$$

Proof. Let (Y, J) be random variables such that

$$(Y, J) = \arg \max_{(Y,J):\ \Delta((X,I),(Y,J)) \le \epsilon} \tilde{\mathbf{H}}_\infty(Y|J).$$

Here, note that we have $\Delta(I, J) \le \epsilon$. Then,

$$\begin{aligned}
&\Delta\big((h, h(X), I), (h, U(\mathcal{D}), I\big) \\
\le &\Delta\big((h, h(X), I), (h, h(Y), J)\big) + \Delta\big((h, h(Y), J), (h, U(\mathcal{D}), J)\big) \\
&\quad + \Delta\big((h, U(\mathcal{D}), J), (h, U(\mathcal{D}), I)\big) \\
\le &2\epsilon + \frac{1}{2} \cdot \sqrt{2^{-\tilde{\mathbf{H}}_\infty(Y|J)} \cdot |\mathcal{D}|} \qquad (3.3)
\end{aligned}$$

In the above derivation, Eq. (3.3) follows from the standard generalized leftover hash lemma of [DORS04] and the definition of smooth average min-entropy. This completes the proof. □

Lossy Mode for LWE. It is well known that the secret vector $\mathbf{s} \in \mathbb{Z}_q^n$ is uniquely defined (with all but negligible probability) given an LWE instance $(\mathbf{A}, \mathbf{A}^\top\mathbf{s}+\mathbf{x}) \in \mathbb{Z}_q^{n\times m} \times \mathbb{Z}_q^m$ when $\mathbf{A}$ is uniformly chosen from $\mathbb{Z}_q^{n\times m}$ for sufficiently large m. On the other hand, if we sample $\mathbf{A}$ from a special distribution which is computationally indistinguishable from the uniform distribution over $\mathbb{Z}_q^{n\times m}$, then the pair $(\mathbf{A}, \mathbf{A}^\top\mathbf{s} + \mathbf{x})$ leaks almost no information of the secret $\mathbf{s}$. Since the LWE problem of this version does not reveal much information about the secret $\mathbf{s}$, this instance is often referred to as the "*lossy mode*". A series of works, e.g., [GKPV10, BKPW12, AKPW13, LSSS17] have investigated the lossy nature of the problem. We first formally describe the procedure $\mathsf{SampleLossy}$ to sample $\mathbf{A}$ in the lossy mode. Let n, m, ℓ be positive integers, and χ be a distribution over $\mathbb{Z}_q$.

$\mathsf{SampleLossy}(n, m, \ell, \chi)$: It samples $\mathbf{C} \xleftarrow{\$} U(\mathbb{Z}_q^{n\times\ell})$, $\mathbf{B} \xleftarrow{\$} U(\mathbb{Z}_q^{\ell\times m})$, and $\mathbf{F} \xleftarrow{\$} \chi^{n\times m}$, and outputs $\mathbf{A} = \mathbf{CB} + \mathbf{F}$.

It is easy to see that $\mathbf{A}$ in the lossy mode is indistinguishable from random under the LWE assumption by a standard hybrid argument.

Lemma 3.9. *For any PPT algorithm $\mathcal{A}$, there exists a PPT algorithm $\mathcal{B}$ such that*

$$\big|\Pr[\mathcal{A}(\mathbf{A}_0) = 1] - \Pr[\mathcal{A}(\mathbf{A}_1) = 1]\big| \le n \cdot \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{\ell,m,q,\chi^m}}(\lambda)$$

and $\mathsf{Time}(\mathcal{B}) \approx \mathsf{Time}(\mathcal{A})$ *where* $\mathbf{A}_0 \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ *and* $\mathbf{A}_1 \xleftarrow{\$} \mathsf{SampleLossy}(n, m, \ell, \chi)$. *If $\mathcal{A}$ has access to a quantum random oracle from $\{0,1\}^a$ to $\{0,1\}^b$, then the right hand side is replaced by* $n \cdot \mathsf{Adv}_{\mathcal{B},\mathsf{QRO}_{a,b}}^{\mathsf{LWE}_{\ell,n,q,\chi}}(\lambda)$, *and the number of quantum random oracle queries by $\mathcal{A}$ and $\mathcal{B}$ are the same.*

We will be using the following lemma slightly adapted from the works of [AKPW13]. In particular, we consider the smooth *average* min-entropy rather than the smooth min-entropy.

Lemma 3.10 (Adapted from [AKPW13], Lemma B.4)**.** *Let n, ℓ, m, q, β be positive integer parameters, α, β be some Gaussian parameters, and χ be a distribution (all parameterized by the security parameter λ, such that* $\Pr_{x\leftarrow\chi}[|x| \ge \beta q] \le \mathsf{negl}(\lambda)$ *and $\alpha \ge \beta\gamma nm$. Let* $\mathbf{s}$ *and* $\mathbf{e}$ *be random variables distributed according to $U([-\gamma, \gamma]^n)$ and $D_{\mathbb{Z}^m,\alpha q}$, respectively. Furthermore, let $\mathbf{A}$ be a matrix sampled by* $\mathsf{SampleLossy}(n, m, \ell, \chi)$. *Then, for any $\epsilon \ge 2^{-\lambda}$, we have the following:.*

$$\tilde{\mathbf{H}}_\infty^\epsilon(\mathbf{s}|\mathbf{A}, \mathbf{A}^\top\mathbf{s} + \mathbf{e}) \ge \mathbf{H}_\infty(\mathbf{s}) - (\ell + 2\lambda)\log q - \mathsf{negl}(\lambda).$$

Remark 3.3. *In [AKPW13], they do not consider the average over $(\mathbf{A}, \mathbf{A}^\top\mathbf{s} + \mathbf{e})$, i.e., they only prove the above lemma for $\mathbf{H}_\infty^\epsilon(\mathbf{s}|\mathbf{A}, \mathbf{A}^\top\mathbf{s} + \mathbf{e})$. However, their proof actually works for the above statement as well. This change will be useful during the security proof when we apply the (smoothed) generalized leftover hash lemma, which works for* smooth-average *min-entropies.*

3.5.2 Construction

Let the identity space $\mathcal{ID}$ of the scheme be $\mathcal{ID} = \{0,1\}^{\ell_{\mathsf{ID}}}$, where $\ell_{\mathsf{ID}}(\lambda)$ denotes the identity-length. Let also $\mathsf{H} : \{0,1\}^{\ell_{\mathsf{ID}}+1} \to \mathbb{Z}_q^n$ be a hash function treated as a random oracle during the security analysis where ℓ_{ID} denotes the identity-length. The IBE scheme $\mathsf{GPV}_{\mathsf{mult}}$ is given as follows. For simplicity, we describe the scheme as a stateful one. As remarked in Remark 3.1, we can make the scheme stateless without any additional assumption in the QROM.

$\mathsf{Setup}(1^\lambda)$: On input 1^λ, it first chooses a prime q, positive integers n, m, γ, and Gaussian parameters α, σ, where all these values are implicitly a function of the security parameter λ. The precise parameter selection is specified in the following section. It then runs $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ to generate a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ with a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ such that $\|\mathbf{T_A}\|_{\text{GS}} \leq O(n \log q)$. Then it outputs

$$\mathsf{mpk} = \mathbf{A} \quad \text{and} \quad \mathsf{msk} = \mathbf{T_A}$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: If $\mathsf{sk}_{\mathsf{ID}}$ is already generated, then this algorithm returns it. Otherwise it picks $b_{\mathsf{ID}} \xleftarrow{\$} \{0,1\}$, computes $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathsf{H}(\mathsf{ID}\|b_{\mathsf{ID}})$, and samples $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} \in \mathbb{Z}^m$ such that

$$\mathbf{A}\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} \mod q$$

as $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$. It returns $\mathsf{sk}_{\mathsf{ID}} = (b_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ as the secret key.

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}$, it first samples $\mathbf{s} \xleftarrow{\$} U([-\gamma, \gamma])$, $\mathbf{x} \leftarrow D_{\mathbb{Z}^m, \alpha q}$. Then it computes $\mathbf{u}_{\mathsf{ID}\|0} = \mathsf{H}(\mathsf{ID}\|0)$ and $\mathbf{u}_{\mathsf{ID}\|1} = \mathsf{H}(\mathsf{ID}\|1)$ and sets the ciphertext as

$$c_0 = \mathbf{u}_{\mathsf{ID}\|0}^\top \mathbf{s} + \mathsf{M}\lfloor q/2 \rceil, \quad c_1 = \mathbf{u}_{\mathsf{ID}\|1}^\top \mathbf{s} + \mathsf{M}\lfloor q/2 \rceil, \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

Finally, it outputs the ciphertext $C = (c_0, c_1, \mathbf{c}_2) \in \mathbb{Z}_q \times \mathbb{Z}_q \times \mathbb{Z}_q^m$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C)$: To decrypt a ciphertext $C = (c_0, c_1, \mathbf{c}_2)$ with a secret key $\mathsf{sk}_{\mathsf{ID}}$, it computes $w = c_{b_{\mathsf{ID}}} - \mathbf{c}_2^\top \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} \in \mathbb{Z}_q$ and outputs 0 if w is closer to 0 than to $\lfloor q/2 \rceil$ modulo q. Otherwise it outputs 1.

3.5.3 Correctness and Prameter Selection

The following shows correctness of the above IBE scheme.

Lemma 3.11 (Correctness). *Suppose the parameters q, σ, and α are such that*

$$\sigma > \|\mathbf{T_A}\|_{\text{GS}} \cdot \sqrt{\log(2m+4)/\pi}, \qquad \alpha < 1/4\sigma m.$$

Let $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} \leftarrow \mathsf{KeyGen}(\mathbf{A}, \mathbf{T_A}, \mathsf{ID}), C \leftarrow \mathsf{Enc}(\mathbf{A}, \mathsf{ID}', \mathsf{M} \in \{0,1\})$ and $\mathsf{M}' \leftarrow \mathsf{Dec}(\mathbf{A}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}, C)$. If $\mathsf{ID} = \mathsf{ID}'$, then with overwhelming probability we have $\mathsf{M}' = \mathsf{M}$.

Proof. When the Dec algorithm operates as specified, we have

$$w = c_{b_{\mathsf{ID}}} - \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}^\top \mathbf{c}_2 = \mathsf{M}\lfloor q/2 \rceil + \underbrace{\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}^\top \mathbf{x}}_{\text{error term}}.$$

By Lemma 2.12 and the condition posed on the choice of σ, we have that the distribution of $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ is $2^{-\Omega(n)}$ close to $D_{\Lambda^\perp_{\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}}(\mathbf{A}),\sigma}$. Therefore, by Lemma 2.4, we have $\|\mathbf{x}\| \leq \alpha q \sqrt{m}$, and $\|\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}\| \leq \sigma \cdot \sqrt{m}$ except for $2^{-\Omega(n)}$ probability. Then, the error term is bounded by

$$|\mathbf{h}^\top \mathbf{x} - \mathbf{e}_{\mathsf{ID}}^\top \mathbf{x}| \leq |\mathbf{e}_{\mathsf{ID}}^\top \mathbf{x}| \leq \alpha q \sigma m.$$

Hence, for the error term to have absolute value less than $q/4$, it suffices to choose q and α as in the statement of the lemma. $\square$

Parameter Selection. For the system to satisfy correctness and make the security proof work, we need the following restrictions. Note that we will prove the security of the scheme under $\mathsf{LWE}_{\ell,m,q,\chi^m}$, where ℓ is specified in the following.

- The error term is less than $q/4$ (i.e., $\alpha < 1/4m\sigma$ by Lemma 3.11)
- $\mathsf{TrapGen}$ operates properly (i.e., $m > 3n \log q$ by Lemma 2.12)
- Samplable from $D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}}\|b_{\mathsf{ID}}}(\mathbf{A}),\sigma}$ (i.e., $\sigma > \|\mathbf{T_A}\|_{\text{GS}} \cdot \sqrt{\log(2m+4)/\pi} = O(\sqrt{n \log m \log q})$ by Lemma 2.12),
- $\mathsf{LWE}_{\ell,n,q,\chi}$ is hard so that we can use the lossy mode in the proof by Lemma 3.9 (i.e., $\chi = D_{\mathbb{Z},2\sqrt{\ell}}$),
- σ is sufficiently large so that we can apply Lemma 2.1 (i.e., $\sigma > \sqrt{n + \log m}$),
- we can apply to Lemma 3.10 in the proof (i.e., $\alpha > \beta\gamma n m$ for β such that $\Pr_{x\leftarrow\chi}[|x| \geq \beta q] \leq \mathsf{negl}(\lambda)$),
- we can apply the generalized leftover hash lemma (Lemma 3.8) in the proof (i.e., $n\log(2\gamma) - (\ell + 3\lambda)\log q \geq \log q + \Omega(n)$).

To satisfy these requirements, for example, we can set the parameters $\ell, n, m, q, \sigma, \alpha, \beta, \gamma$ as follows:

$$\begin{array}{llll} n = 25\ell, & m = n^{1+\kappa}, & \sigma = n^{0.5+\kappa}, & q = 5n^{5.5+3\kappa}, \\ \alpha q = n^{4+\kappa}, & \beta q = n, & \gamma = n, & \end{array}$$

where $\kappa > 0$ is a constant that can be set arbitrarily small. To withstand attacks running in time 2^{λ}, we may set $\ell = \tilde{\Omega}(\lambda)$. In the above, we round up m to the nearest integer and q to the nearest largest prime. As the case with the single-challenge setting, if we make the more aggressive choice of using the negligible notion $2^{-\omega(\log\lambda)}$, we will be able to obtain better parameter selections.

3.5.4 Security Proof in ROM

We can (almost) tightly prove the security of our IBE scheme $\mathsf{GPV}_{\mathsf{mult}}$ both in the classical ROM and QROM settings. The following theorem addresses the security of $\mathsf{GPV}_{\mathsf{mult}}$ in the classical ROM setting.

Theorem 3.3. *The IBE scheme $\mathsf{GPV}_{\mathsf{mult}}$ is adaptively-anonymous multi-challenge secure assuming the hardness of $\mathsf{LWE}_{\ell,m,q,\chi}$ in the random oracle model, where $\chi = D_{\mathbb{Z},\alpha q}$. Namely, for any classical adversary $\mathcal{A}$ making at most Q_{H} queries to H, Q_{ch} challenge queries, and Q_{ID} secret key queries, there exists an algorithm $\mathcal{B}$ such that*

$$\mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}_{\mathsf{mult}}}(\lambda) \leq 3n \cdot \mathsf{Adv}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B}}(\lambda) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot 2^{-\Omega(n)}$$

and

$$\mathsf{Time}(\mathcal{B}) = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot \mathsf{poly}(\lambda).$$

Proof of Theorem 3.3. Let $\mathsf{CTSam}(\mathsf{mpk})$ be an algorithm that outputs $(c_0, c_1, \mathbf{c}_2)$ such that $c_0 \xleftarrow{\$} \mathbb{Z}_q$, $c_1 \xleftarrow{\$} \mathbb{Z}_q$, and $\mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}$ for $\mathbf{s} \xleftarrow{\$} U([-\gamma, \gamma])$ and $\mathbf{x} \leftarrow D_{\mathbb{Z}^m, \alpha q}$. Let also $\mathcal{A}$ be a classical PPT adversary that attacks the (multi-challenge) adaptively-anonymous security of the IBE scheme. Without loss of generality, we make some simplifying assumptions on $\mathcal{A}$. First, we assume that whenever $\mathcal{A}$ queries a secret key or asks for a challenge ciphertext, $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$ for the corresponding ID has already been queried to the random oracle H. We also assume that whenever $\mathcal{A}$ queries H on input $\mathsf{ID}\|b$, it also queries H on input $\mathsf{ID}\|\bar{b}$ as well. Furthermore, we assume that $\mathcal{A}$ makes the same query for the random oracle or secret key oracle at most once. We show the security of the scheme via the following games. In each game, we define X_i as the event that the adversary $\mathcal{A}$ wins in Game_i.

Game_0 : This is the real security game. At the beginning of the game, $(\mathbf{A}, \mathbf{T_A}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$ is run and the adversary $\mathcal{A}$ is given $\mathbf{A}$. The challenger then samples $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and keeps it secret. During the game, $\mathcal{A}$ may make random oracle queries, secret key queries, and a challenge query. These queries are handled as follows:

- When $\mathcal{A}$ queries the random oracle H on $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$, the challenger samples $\mathbf{u}_{\mathsf{ID}\|0} \xleftarrow{\$} \mathbb{Z}_q^n$ and $\mathbf{u}_{\mathsf{ID}\|1} \xleftarrow{\$} \mathbb{Z}_q^n$ and returns $\mathbf{u}_{\mathsf{ID}\|0}$ and $\mathbf{u}_{\mathsf{ID}\|1}$.

- When $\mathcal{A}$ queries a secret key for ID, the challenger first computes $b_{\mathsf{ID}} \xleftarrow{\$} \{0,1\}$ and returns $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$ to $\mathcal{A}$ where $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ is the vector chosen during the simulation of the random oracle.

- When the adversary makes the challenge query for ID^* and a message M^*, the challenger returns $(c_0, c_1, \mathbf{c}_2) \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^*, \mathsf{M})$ if $\mathsf{coin} = 0$ and $(c_0, c_1, \mathbf{c}_2) \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$. It then returns $C^* = (c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$.

At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger outputs $\widehat{\mathsf{coin}}$. By definition, we have $\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} - \mathsf{coin}] - \frac{1}{2}\right| = \mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A}, \mathsf{GPV}_{\mathsf{mult}}}(\lambda)$.

Game_1 : In this game, we change the game so that b_{ID} is chosen when $\mathcal{A}$ queries the random oracle H on input $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$ rather than when $\mathcal{A}$ queries a secret key for ID. It is clear that this is only a conceptual change and we have $\Pr[X_0] = \Pr[X_1]$.

Game_2 : In this game, we change the way the random oracle queries to H are answered. When $\mathcal{A}$ queries H on inputs $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$, the challenger first samples $b_{\mathsf{ID}} \xleftarrow{\$} \{0,1\}$ as specified in the previous game. Then, it generates a pair $(\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ by first sampling $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} \leftarrow \mathcal{D}_{\mathbb{Z}^m, \sigma}$ and setting $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathbf{A}\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$. It also samples $\mathbf{u}_{\mathsf{ID}\|\bar{b}_{\mathsf{ID}}} \xleftarrow{\$} \mathbb{Z}_q^n$. Then it locally stores the tuples $(\mathsf{ID}\|b_{\mathsf{ID}}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \perp)$ and $(\mathsf{ID}\|\bar{b}_{\mathsf{ID}}, \mathbf{u}_{\mathsf{ID}\|\bar{b}_{\mathsf{ID}}}, \perp)$, and returns $\mathbf{u}_{\mathsf{ID}\|0}$ and $\mathbf{u}_{\mathsf{ID}\|1}$ to $\mathcal{A}$. Here, we remark that when $\mathcal{A}$ makes a secret key query for ID, the challenger returns $\mathbf{e}'_{\mathsf{ID}\|b_{\mathsf{ID}}} \xleftarrow{\$} \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$, which is independent from $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ that was generated in the simulation of the random oracle H on input $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$. Note that in this game, we only change the distribution of $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ for each identity. Due to Lemma 2.1, the distribution of $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ in Game_2 is $2^{-\Omega(n)}$-close to that of Game_1 except for $2^{-\Omega(n)}$ fraction of $\mathbf{A}$ since we choose $\sigma > \sqrt{n + \log m}$. Therefore, the statistical distance between the view of $\mathcal{A}$ in Game_1 and Game_2 is $2^{-\Omega(n)} + Q_{\mathsf{H}} \cdot 2^{-\Omega(n)} < Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$. Therefore, we have $\left|\Pr[X_1] - \Pr[X_2]\right| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Game_3 : In this game, we change the way secret key queries are answered. By the end of this game, the challenger will no longer require the trapdoor $\mathbf{T_A}$ to generate the secret keys. When $\mathcal{A}$ queries

H on $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$, the challenger generates a pair $(\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ as in the previous game. Then it locally stores the tuple $(\mathsf{ID}\|b_{\mathsf{ID}}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ and returns $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ to $\mathcal{A}$. When $\mathcal{A}$ queries a secret key for ID, the challenger retrieves the unique tuple $(\mathsf{ID}\|b_{\mathsf{ID}}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ from the local storage and returns $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$. For any fixed $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} \in \mathbb{Z}_q^n$, let $\mathbf{e}^{(2)}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ and $\mathbf{e}^{(3)}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ be random variables that are distributed according to the distributions of $\mathsf{sk}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ conditioning on $\mathsf{H}(\mathsf{ID}) = \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ in Game_2 and Game_3, respectively. Due to Lemma 2.12, we have $\Delta(\mathbf{e}^{(2)}_{\mathsf{ID}\|b_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}}(\mathbf{A}),\sigma}) \le 2^{-\Omega(n)}$. On the other hand, due to Lemma 2.1, we have $\Delta(\mathbf{e}^{(3)}_{\mathsf{ID}\|b_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}}(\mathbf{A}),\sigma}) \le 2^{-\Omega(n)}$. Since $\mathcal{A}$ obtains at most Q_{ID} user secret keys, we have $\left|\Pr[X_2] - \Pr[X_3]\right| = Q_{\mathsf{ID}} \cdot 2^{-\Omega(n)}$.

Game_4 : In this game, we change the way the matrix $\mathbf{A}$ is generated. Concretely, the challenger chooses $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ without generating the associated trapdoor $\mathbf{T}_{\mathbf{A}}$. By Lemma 2.12, this makes only $2^{-\Omega(n)}$-statistical difference. Since the challenger can answer all the secret key queries without the trapdoor due to the change we made in the previous game, the view of $\mathcal{A}$ is altered only negligibly. Therefore, we have $\left|\Pr[X_3] - \Pr[X_4]\right| = 2^{-\Omega(n)}$.

Game_5 : In this game, we change $\mathbf{A}$ to lossy mode (See Lemma 3.10). We claim that $\left|\Pr[X_4] - \Pr[X_5]\right|$ is negligible assuming the hardness of the $\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}$ problem. To show this, we use $\mathcal{A}$ to construct an adversary $\mathcal{B}'$ that distinguishes random $\mathbf{A}$ from that in lossy mode. This can be done by a straightforward reduction since $\mathcal{B}'$ does not require the master secret key $\mathbf{T}_{\mathbf{A}}$ to simulate the game due to the modification we made in Game_3. We therefore by Lemma 3.9 conclude that there exists $\mathcal{B}$ such that we have $\left|\Pr[X_4] - \Pr[X_5]\right| = n \cdot \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}(\lambda)$.

Game_6 : In this game, we change the way the challenge ciphertexts are generated. Recall that by our assumption, $\mathcal{A}$ makes queries for H on inputs $\mathsf{ID}^*\|0$ and $\mathsf{ID}^*\|1$ before making a challenge query for identity ID^*. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$ it samples a ciphertext as $C^* \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ and returns C^* to $\mathcal{A}$ if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} = \mathbf{u}^{\top}_{\mathsf{ID}^*\|b_{\mathsf{ID}^*}}\mathbf{s} + \mathsf{M}^*\lfloor q/2\rceil, \quad c_{\bar{b}_{\mathsf{ID}^*}} \xleftarrow{\$} \mathbb{Z}_q, \quad \mathbf{c}_2 = \mathbf{A}^{\top}\mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \xleftarrow{\$} U([-\gamma,\gamma])$, where b_{ID^*} is generated when the hash queries for $\mathsf{ID}^*\|0$ and $\mathsf{ID}^*\|1$ were made. It then rearranges the terms if necessary and returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$.

We argue that the view of $\mathcal{A}$ is statistically close to that in the previous game. To prove, we do a hybrid argument over all the challenge ciphertexts and change $c_{\bar{b}_{\mathsf{ID}^*}}$ to be random one-by-one (when $\mathsf{coin} = 0$). To conclude, it suffices to show that $c_{\bar{b}_{\mathsf{ID}^*}}$ is distributed $2^{-\Omega(n)}$ close to the uniform distribution over $\mathbb{Z}_q$. For each ciphertext, for any $\epsilon' = 2^{-\lambda}$, we have

$$\begin{aligned}
\tilde{\mathbf{H}}^{\epsilon'}_{\infty}(\mathbf{s}|\mathbf{A}, c_{b_{\mathsf{ID}^*}}, \mathbf{c}_2) &\ge \tilde{\mathbf{H}}^{\epsilon'}_{\infty}(\mathbf{s}|\mathbf{A}, \mathbf{c}_2) - \log q \\
&\ge \mathbf{H}_{\infty}(\mathbf{s}) - (\ell + 2\lambda + 1)\log q - \mathsf{negl}(\lambda) \\
&= n\log(2\gamma) - (\ell + 3\lambda)\log q \\
&\ge \log q + \Omega(n)
\end{aligned}$$

where the first inequality follows by applying Lemma 3.7 with $\mathcal{Z} = \mathbb{Z}_q$ and $\epsilon = 0$, the second inequality follows from Lemma 3.10, and the last inequality follows from our parameter choice. This implies that $\Delta(\mathbf{u}_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}}, \mathbf{u}^{\top}_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}}\mathbf{s}, |\mathbf{A}, c_{b_{\mathsf{ID}^*}}, \mathbf{c}_2) \le 2^{-\Omega(n)}$ for $\mathbf{u}_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}} \leftarrow \mathbb{Z}_q^n$ by Lemma 3.8 together with Fact 3.1. Therefore, we have $\left|\Pr[X_5] - \Pr[X_6]\right| = Q_{\mathsf{ch}} \cdot 2^{-\Omega(n)}$.

From Game_7 to Game_{10} in the following, we undo the changes we added from Game_2 to Game_5.

Game_7 : In this game, $\mathbf{A}$ is sampled as $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$. Similarly to the change from Game_4 to Game_5, there exists $\mathcal{B}$ such that we have $\left|\Pr[X_6] - \Pr[X_7]\right| = n \cdot \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}(\lambda)$.

Game_8 : In this game, $\mathbf{A}$ is sampled with a trapdoor as $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. Similarly to the change from Game_3 to Game_4, we have $\left|\Pr[X_7] - \Pr[X_8]\right| = 2^{-\Omega(n)}$.

Game_9 : In this game, we change the way secret key queries are answered. When $\mathcal{A}$ makes a secret key query for ID, the challenger returns $\mathbf{e}'_{\mathsf{ID}\|b_{\mathsf{ID}}} \overset{\$}{\leftarrow} \mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$, where b_{ID} and $\mathbf{u}_{\mathsf{ID}}$ are chosen when random oracle queries on $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$ are made. Similarly to the change from Game_2 to Game_3, we have $\left|\Pr[X_8] - \Pr[X_9]\right| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Game_{10} : In this game, we change the way the random oracle queries to H are answered. When $\mathcal{A}$ queries the random oracle on $\mathsf{ID}\|0$ and $\mathsf{ID}\|1$, the challenger samples $\mathbf{u}_{\mathsf{ID}\|0}, \mathbf{u}_{\mathsf{ID}\|1} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$ and locally stores the tuples $(\mathsf{ID}\|0, \mathbf{u}_{\mathsf{ID}\|0}, \perp)$ and $(\mathsf{ID}\|1, \mathbf{u}_{\mathsf{ID}\|1}, \perp)$, and returns $\mathbf{u}_{\mathsf{ID}\|0}$ and $\mathbf{u}_{\mathsf{ID}\|1}$ to $\mathcal{A}$. These $\mathbf{u}_{\mathsf{ID}\|0}$ and $\mathbf{u}_{\mathsf{ID}\|1}$ are also used when answering the secret key queries. Similarly to the change from Game_1 to Game_2, we have $\left|\Pr[X_{10}] - \Pr[X_9]\right| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Because of the changes we introduced in Game_7 to Game_{10}, we can add the following change.

Game_{11} : In this game, we change the way the challenge ciphertexts are generated. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it returns a random ciphertext if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad c_{\bar{b}_{\mathsf{ID}^*}} = \mathbf{u}_{\mathsf{ID}^*\|b_{\mathsf{ID}^*}}^\top \mathbf{s} + \mathsf{M}^* \lfloor q/2 \rceil, \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, where b_{ID^*} is generated when the hash queries for $\mathsf{ID}^*\|0$ and $\mathsf{ID}^*\|1$ were made. It then rearranges the terms if necessary and returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$. We claim that this change is only conceptual. Note that the distribution of the ciphertexts in this game corresponds to that in previous game, if we flip the value of b_{ID^*} for every challenge identity ID^*. However, since $\mathcal{A}$ never makes the secret key query for ID^* and $\mathbf{u}_{\mathsf{ID}^*\|0}$ and $\mathbf{u}_{\mathsf{ID}^*\|1}$ are sampled from exactly the same distribution, the value of b_{ID^*} is information theoretically hidden from $\mathcal{A}$. This implies that the distributions are the same in this and the previous game. Therefore, we have $\Pr[X_{11}] = \Pr[X_{10}]$.

Game_{12} : In this game, we further change the way the challenge ciphertexts are generated. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it returns a ciphertext sampled from $\mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad c_{\bar{b}_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$ and $\mathbf{x} \leftarrow D_{\mathbb{Z}^m,\alpha q}$. It then returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$. We claim that the change is unnoticed by $\mathcal{A}$ assuming the LWE assumption. This can be shown by adding changes to Game_{11} that are almost the same as those we introduced from Game_2 to Game_6. The only difference is that $c_{b_{\mathsf{ID}^*}}$ is always sampled as $c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q$ here. By the similar analysis, there exists $\mathcal{B}$ such that we have $|\Pr[X_{11}] - \Pr[X_{12}]| \le n \cdot \mathsf{Adv}_{\mathcal{B}'}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}(\lambda) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot 2^{-n}$.

Finally, we observe that the challenge ciphertexts are sampled from $\mathsf{CTSam}(\mathsf{mpk})$ regardless of whether $\mathsf{coin} = 0$ or 1. Therefore, we have $\Pr[X_{12}] = 1/2$. Putting things together, the theorem readily follows. $\square$

3.5.5 Security Proof in QROM

As we explained in the introduction, our analysis in the ROM can be easily extended to the QROM setting. We can prove the following theorem that addresses the security of $\mathsf{GPV}_{\mathsf{mult}}$ in the QROM.

Theorem 3.4. *The IBE scheme $\mathsf{GPV}_{\mathsf{mult}}$ is adaptively-anonymous multi-challenge secure assuming the hardness of $\mathsf{LWE}_{\ell,m,q,\chi}$ in the quantum random oracle model, where $\chi = D_{\mathbb{Z},\alpha q}$. Namely, for any classical adversary $\mathcal{A}$ making at most Q_{H} quantum random oracle queries, Q_{ch} challenge queries, and Q_{ID} secret key queries, there exists an algorithm $\mathcal{B}$ making at most $3Q_{\mathsf{H}}+2Q_{\mathsf{ID}}+6Q_{\mathsf{ch}}$ quantum random oracle queries such that*

$$\mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}_{\mathsf{mult}}}(\lambda) \le 3n \cdot \mathsf{Adv}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B},\mathsf{QRO}_{\ell_{\mathsf{ID}}+2,\max\{\ell_r,(\lfloor \log q\rfloor+2\lambda)\times n\}}}(\lambda) + (Q_{\mathsf{H}}+Q_{\mathsf{ID}}+Q_{\mathsf{ch}})\cdot 2^{-\Omega(n)}$$

and

$$\mathsf{Time}(\mathcal{B}) = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}}+Q_{\mathsf{ID}}+Q_{\mathsf{ch}})\cdot \mathsf{poly}(\lambda)$$

where ℓ_r denotes the length of the randomness for $\mathsf{Sample}\mathbb{Z}$.

Proof of Theorem 3.4. Let $\mathsf{CTSam}(\mathsf{mpk})$ be an algorithm that outputs $(c_0, c_1, \mathbf{c}_2)$ such that $c_0 \xleftarrow{\$} \mathbb{Z}_q$, $c_1 \xleftarrow{\$} \mathbb{Z}_q$, and $\mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}$ for $\mathbf{s} \xleftarrow{\$} U([-\gamma,\gamma])$ and $\mathbf{x} \leftarrow D_{\mathbb{Z}^m,\alpha q}$. Let also $\mathcal{A}$ be a quantum adversary that attacks the (multi-challenge) adaptively-anonymous security of the IBE scheme. Without loss of generality, we can assume that $\mathcal{A}$ makes secret key queries on the same identity at most once.

Game_0 : This is the real security game. At the beginning of the game, the challenger chooses random functions $\mathsf{H} \xleftarrow{\$} \mathsf{Func}(\{0,1\}^{\ell_{\mathsf{ID}}+1}, \mathbb{Z}_q^n)$, which is used to simulate the random oracle. The challenger generates $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^m, q)$ and the adversary $\mathcal{A}$ is given $\mathbf{A}$. The challenger then samples $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and keeps it secret. During the game, $\mathcal{A}$ may make (quantum) random oracle queries, secret key queries, and challenge queries. These queries are handled as follows:

- When $\mathcal{A}$ makes a random oracle query on a quantum state $\sum_{\mathsf{ID},b,y} \alpha_{\mathsf{ID},b,y} |\mathsf{ID}\|b\rangle |y\rangle$, then the challenger returns $\sum_{\mathsf{ID},b,y} \alpha_{\mathsf{ID},b,y} |\mathsf{ID}\|b\rangle |\mathsf{H}(\mathsf{ID}\|b) \oplus y\rangle$.
- When $\mathcal{A}$ makes a secret key query on ID, the challenger first chooses $b_{\mathsf{ID}} \xleftarrow{\$} \{0,1\}$, computes $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathsf{H}(\mathsf{ID}\|b_{\mathsf{ID}})$, and returns $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$ to $\mathcal{A}$.
- When $\mathcal{A}$ makes the challenge query for ID^* and a message M^*, the challenger returns $(c_0, c_1, \mathbf{c}_2) \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^*, \mathsf{M})$ if $\mathsf{coin} = 0$ and $(c_0, c_1, \mathbf{c}_2) \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$. It then returns $C^* = (c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$.

At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger outputs $\widehat{\mathsf{coin}}$. By definition, we have $\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} - \mathsf{coin}] - \frac{1}{2}\right| = \mathsf{Adv}^{\mathsf{IBE}}_{\mathcal{A},\mathsf{GPV}_{\mathsf{mult}}}(\lambda)$.

Game_1 : In this game, we change the way b_{ID} is chosen in simulations of secret key queries. Namely, in this game, the challenger first picks a random function $\mathsf{H}' \xleftarrow{\$} \mathsf{Func}(\{0,1\}^{\ell_{\mathsf{ID}}}, \{0,1\})$. When $\mathcal{A}$ makes a secret key query ID, the challenger uses $b_{\mathsf{ID}} = \mathsf{H}'(\mathsf{ID})$ instead of randomly choosing it. We remark that an oracle access to H' is not given to $\mathcal{A}$. Since $\mathsf{H}'(\mathsf{ID})$ is an independently and uniformly random bit for all ID, we have $\Pr[X_0] = \Pr[X_1]$.

Game_2 : In this game, we change the way the random oracle H is simulated. Namely, the challenger first chooses additional random functions $\widehat{\mathsf{H}}_0 \xleftarrow{\$} \mathsf{Func}(\{0,1\}^{\ell_{\mathsf{ID}}}, \{0,1\}^{\ell_r})$ and $\widehat{\mathsf{H}}_1 \xleftarrow{\$} \mathsf{Func}(\{0,1\}^{\ell_{\mathsf{ID}}}, \{0,1\}^{(\lfloor \log q \rfloor + 2n)\times n})$. Let $\iota : \{0,1\}^{(\lfloor \log q \rfloor + 2n)\times n} \to \mathbb{Z}_q^n$ denote a natural embedding function. More precisely, it is given $(a_1, ..., a_n) \in \{0,1\}^{(\lfloor \log q \rfloor + 2n)\times n}$ as an input, and outputs $(\tilde{a}_1 \mod q, ..., \tilde{a}_n \mod q)^T$ where $\tilde{a}_i$ denotes a positive integer whose binary representation is a_i. It is easy to see that if we sample $(a_1, ..., a_n) \xleftarrow{\$} \{0,1\}^{(\lfloor \log q \rfloor + 2n)\times n}$, then the statistical distance between the distribution of $\iota(a_1, ..., a_n)$ and the uniform distribution over $\mathbb{Z}_q^n$ is $2^{-\Omega(n)}$. Then the challenger defines H as follows:

$$\mathsf{H}(\mathsf{ID}\|b) := \begin{cases} \mathbf{A}\mathbf{e}_{\mathsf{ID}\|b} & \text{If } b = \mathsf{H}'(\mathsf{ID}) \\ \iota(\widehat{\mathsf{H}}_1(\mathsf{ID})) & \text{Otherwise} \end{cases} \tag{3.4}$$

where $\mathbf{e}_{\mathsf{ID}\|b} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}_0(\mathsf{ID}))$. We remark that oracle accesses to $\widehat{\mathsf{H}}_0$ and $\widehat{\mathsf{H}}_1$ are not given to $\mathcal{A}$. For any fixed ID, for the case of $b \neq \mathsf{H}'(\mathsf{ID})$, the distribution of $\mathsf{H}(\mathsf{ID}\|b)$ is identical for all ID and its statistical distance from the uniform distribution is $2^{-\Omega(n)}$ as remarked above. For the case of $b = \mathsf{H}'(\mathsf{ID})$, the distribution of $\mathsf{H}(\mathsf{ID}\|b)$ is identical for all ID and its statistical distance from the uniform distribution is $2^{-\Omega(n)}$ for all but $2^{-\Omega(n)}$ fraction of $\mathbf{A}$ due to Lemma 2.1 since we choose $\sigma > \sqrt{n + \log m}$. Then due to Lemma 3.2, we have $\left|\Pr[X_1] - \Pr[X_2]\right| = 2^{-\Omega(n)} + 4Q_{\mathsf{H}}^2\sqrt{2^{-\Omega(n)}} + 4Q_{\mathsf{H}}^2\sqrt{2^{-\Omega(n)}} = Q_{\mathsf{H}}^2 \cdot 2^{-\Omega(n)}$.

Game_3 : In this game, we change the way secret key queries are answered. By the end of this game, the challenger will no longer require the trapdoor $\mathbf{T}_{\mathbf{A}}$ to generate the secret keys. When $\mathcal{A}$ queries a secret key for ID, the challenger returns $(b_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}})$ where $b_{\mathsf{ID}} = \mathsf{H}'(\mathsf{ID})$ and $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}_0(\mathsf{ID}))$. Since b_{ID} is unchanged from the previous game, we only have to prove that the distribution of $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ differs negligibly from that in the previous game. For any fixed $\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}} \in \mathbb{Z}_q^n$, let $\mathbf{e}^{(2)}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ and $\mathbf{e}^{(3)}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ be random variables that are distributed according to the distributions of $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ conditioning on $\mathsf{H}(\mathsf{ID}\|b_{\mathsf{ID}}) = \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}$ in Game_2 and Game_3, respectively. Due to Lemma 2.12, we have $\Delta(\mathbf{e}^{(2)}_{\mathsf{ID}\|b_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. On the other hand, due to Lemma 2.1, we have $\Delta(\mathbf{e}^{(3)}_{\mathsf{ID}\|b_{\mathsf{ID}}}, D_{\Lambda^{\perp}_{\mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}}(\mathbf{A}),\sigma}) \leq 2^{-\Omega(n)}$. Since $\mathcal{A}$ obtains at most Q_{ID} user secret keys $\mathbf{e}_{\mathsf{ID}}$, we have $\left|\Pr[X_2] - \Pr[X_3]\right| = Q_{\mathsf{ID}} \cdot 2^{-\Omega(n)}$.

Game_4 : In this game, we change the way the matrix $\mathbf{A}$ is generated. Concretely, the challenger chooses $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$ without generating the associated trapdoor $\mathbf{T}_{\mathbf{A}}$. By Lemma 2.12, the distribution of $\mathbf{A}$ differs at most by $2^{-\Omega(n)}$. Since the challenger can answer all the secret key queries without the trapdoor due to the change we made in the previous game, the view of $\mathcal{A}$ is altered only by $2^{-\Omega(n)}$. Therefore, we have $\left|\Pr[X_3] - \Pr[X_4]\right| = 2^{-\Omega(n)}$.

Game_5 : In this game, we change $\mathbf{A}$ to the lossy mode (See Lemma 3.10). We claim that $\left|\Pr[X_4] - \Pr[X_5]\right|$ is negligible assuming the hardness of the $\mathsf{LWE}_{\ell,n,q,D_{\mathbb{Z},\alpha q}}$ problem relative to a quantum random oracle $|\tilde{\mathsf{H}}\rangle : \{0,1\}^{\ell_{\mathsf{ID}}+2} \to \{0,1\}^{\max\{\ell_r, (\lfloor \log q \rfloor + 2\lambda)\times n\}}$. For this purpose, we construct an algorithm $\mathcal{B}'$ that distinguishes uniform $\mathbf{A}$ and that in the lossy mode relative to $|\tilde{\mathsf{H}}\rangle$. As remarked in Section 3.3.1, we can implement three independent quantum random oracles $|\mathsf{H}'\rangle : \{0,1\}^{\ell_{\mathsf{ID}}} \to \{0,1\}$, $|\widehat{\mathsf{H}}_0\rangle : \{0,1\}^{\ell_{\mathsf{ID}}} \to \{0,1\}^{\ell_r}$, and $|\widehat{\mathsf{H}}_1\rangle : \{0,1\}^{\ell_{\mathsf{ID}}} \to \{0,1\}^{(\lfloor \log q \rfloor + 2\lambda)\times n}$ by using $|\tilde{\mathsf{H}}\rangle$. Therefore we assume that $\mathcal{B}'$ can access to three random oracles $|\mathsf{H}'\rangle$, $|\widehat{\mathsf{H}}_0\rangle$ and $|\widehat{\mathsf{H}}_1\rangle$.

$\mathcal{B}'$ is given a matrix $\mathbf{A}$, and its task is to distinguish whether $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^n$ or $\mathbf{A} \xleftarrow{\$} \mathsf{SampleLossy}(n, m, \ell, \chi)$. First, we remark that $\mathcal{B}'$ can simulate the quantum random oracle $|\mathsf{H}\rangle$ by using its own random oracles $|\mathsf{H}'\rangle$, $|\widehat{\mathsf{H}}_0\rangle$ and $|\widehat{\mathsf{H}}_1\rangle$ because $|\mathsf{H}\rangle$ is programmed based on these three oracles by the modification made in Game_2. $\mathcal{B}'$ sets the master public key to be the LWE matrix $\mathbf{A}$. Note that unlike the real IBE scheme, $\mathcal{B}'$ does not require the master secret key $\mathbf{T_A}$ due to the modification we made in Game_4. Namely, when $\mathcal{A}$ queries ID for the secret key oracle, $\mathcal{B}'$ just returns $(b_{\mathsf{ID}}, \mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}})$ where $b_{\mathsf{ID}} = \mathsf{H}'(\mathsf{ID})$ and $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} := \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}_0(\mathsf{ID}))$. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it samples a ciphertext as $C^* \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$, and honestly generates a ciphertext as $C^* \xleftarrow{\$} \mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}^*, \mathsf{M}^*)$ if $\mathsf{coin} = 0$, and returns C^* to $\mathcal{A}$. At the end of the game, $\mathcal{A}$ outputs $\widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\widehat{\mathsf{coin}} = \mathsf{coin}$ and 0 otherwise.

It can be seen that if $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ the view of the adversary corresponds to Game_5, and if $\mathbf{A} \xleftarrow{\$} \mathsf{SampleLossy}(n, m, \ell, \chi)$ the view of the adversary corresponds to Game_6. Therefore we can bound $\left|\Pr[X_5] - \Pr[X_6]\right|$ by the distinguishing advantage of $\mathcal{B}'$, which makes at most $3Q_{\mathsf{H}} + 2Q_{\mathsf{ID}} + 6Q_{\mathsf{ch}}$ quantum random oracle queries. As for the running time, we have $\mathsf{Time}(\mathcal{B}') = \mathsf{Time}(\mathcal{A}) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot \mathsf{poly}(\lambda)$ since all $\mathcal{B}'$ has to do is to run $\mathcal{A}$ once plus to compute some additional computations that can be done in a fixed polynomial time whenever $\mathcal{A}$ makes any query. Then by Lemma 3.9 we conclude that there exists $\mathcal{B}$ such that we have $\left|\Pr[X_4] - \Pr[X_5]\right| = n \cdot \mathsf{Adv}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B},\mathsf{QRO}_{\ell_{\mathsf{ID}}+2,\max\{\ell_r,(\lfloor \log q\rfloor+2\lambda)\times n\}}}(\lambda)$ and the running time of $\mathcal{B}$ is almost the same as that of $\mathcal{B}'$.

Game_6 : In this game, we change the way the challenge ciphertexts are generated. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it samples a ciphertext as $C^* \xleftarrow{\$} \mathsf{CTSam}(\mathsf{mpk})$ and returns C^* to $\mathcal{A}$ if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} = \mathbf{u}^\top_{\mathsf{ID}^*\|b_{\mathsf{ID}^*}}\mathbf{s} + \mathsf{M}^*\lfloor q/2\rceil, \quad c_{\bar{b}_{\mathsf{ID}^*}} \xleftarrow{\$} \mathbb{Z}_q, \quad \mathbf{c}_2 = \mathbf{A}^\top\mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \xleftarrow{\$} U([-\gamma, \gamma])$, where $b_{\mathsf{ID}^*} = \mathsf{H}'(\mathsf{ID}^*)$. It then rearranges the terms if necessary and returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$.

We argue that the view of $\mathcal{A}$ is statistically close to that in the previous game. To prove, we do a hybrid argument over all the challenge ciphertexts and change $c_{\bar{b}_{\mathsf{ID}^*}}$ to be random one-by-one (when $\mathsf{coin} = 0$). To conclude, it suffices to show that $c_{\bar{b}_{\mathsf{ID}^*}}$ is distributed $2^{-\Omega(n)}$ close to the uniform distribution over $\mathbb{Z}_q$. For each ciphertext, for any $\epsilon' = 2^{-\lambda}$, we have

$$\begin{aligned} \tilde{\mathbf{H}}^{\epsilon'}_\infty(\mathbf{s}|\mathbf{A}, c_{b_{\mathsf{ID}^*}}, \mathbf{c}_2) &\geq \tilde{\mathbf{H}}^{\epsilon'}_\infty(\mathbf{s}|\mathbf{A}, \mathbf{c}_2) - \log q \\ &\geq \mathbf{H}_\infty(\mathbf{s}) - (\ell + 2\lambda + 1)\log q - \mathsf{negl}(\lambda) \\ &= n\log(2\gamma) - (\ell + 3\lambda)\log q \\ &\geq \log q + \Omega(n) \end{aligned}$$

where the first inequality follows by applying Lemma 3.7 with $\mathcal{Z} = \mathbb{Z}_q$ and $\epsilon = 0$, the second inequality follows from Lemma 3.10, and the last inequality follows from our parameter choice. This implies that $\Delta(\mathbf{u}_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}}, \mathbf{u}^\top_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}}\mathbf{s}, |\mathbf{A}, c_{b_{\mathsf{ID}^*}}, \mathbf{c}_2) \leq 2^{-\Omega(n)}$ for $\mathbf{u}_{\mathsf{ID}^*\|\bar{b}_{\mathsf{ID}^*}} \leftarrow \mathbb{Z}_q^n$ by Lemma 3.8 together with Fact 3.1. Therefore, we have $\left|\Pr[X_5] - \Pr[X_6]\right| = Q_{\mathsf{ch}} \cdot 2^{-\Omega(n)}$.

From Game_7 to Game_{10} in the following, we undo the changes we added from Game_2 to Game_5.

Game_7 : In this game, $\mathbf{A}$ is sampled as $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$. Similarly to the change from Game_4 to Game_5, there exists $\mathcal{B}'$ such that we have $\left|\Pr[X_6] - \Pr[X_7]\right| = n \cdot \mathsf{Adv}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}_{\mathcal{B}',\mathsf{QRO}_{\ell_{\mathsf{ID}}+2,\max\{\ell_r,(\lfloor \log q\rfloor+2\lambda)\times n\}}}(\lambda)$.

Game_8 : In this game, $\mathbf{A}$ is sampled with a trapdoor as $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$. Similarly to the change from Game_3 to Game_4, we have $\left|\Pr[X_7] - \Pr[X_8]\right| = 2^{-\Omega(n)}$.

Game_9 : In this game, we change the way secret key queries are answered. When $\mathcal{A}$ makes a secret key query for ID, the challenger returns $\mathbf{e}'_{\mathsf{ID}\|b_{\mathsf{ID}}} \overset{\$}{\leftarrow} \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{u}_{\mathsf{ID}\|b_{\mathsf{ID}}}, \sigma)$, where $b_{\mathsf{ID}} = \mathsf{H}'(\mathsf{ID})$, $\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}} = \mathsf{Sample}\mathbb{Z}(\sigma; \widehat{\mathsf{H}}_0(\mathsf{ID}))$ and $\mathbf{u}_{\mathsf{ID}} = \mathbf{A}\mathbf{e}_{\mathsf{ID}\|b_{\mathsf{ID}}}$. Similarly to the change from Game_2 to Game_3, we have $\big|\Pr[X_8] - \Pr[X_9]\big| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Game_{10} : In this game, we change the way the random oracle queries to H are answered. Namely, the challenger simply uses random function $\mathsf{H} : \{0,1\}^{\ell_{\mathsf{ID}}+1} \to \mathbb{Z}_q^n$ to simulate $|\mathsf{H}\rangle$ instead of programming it as in Eq. (3.4). Similarly to the change from Game_1 to Game_2, we have $\big|\Pr[X_{10}] - \Pr[X_9]\big| = Q_{\mathsf{H}} \cdot 2^{-\Omega(n)}$.

Because of the changes we introduced in Game_7 to Game_{10}, we can add the following change.

Game_{11} : In this game, we change the way the challenge ciphertexts are generated. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it returns a ciphertext sampled from $\mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad c_{\bar{b}_{\mathsf{ID}^*}} = \mathbf{u}_{\mathsf{ID}^*\|b_{\mathsf{ID}^*}}^\top \mathbf{s} + \mathsf{M}\lfloor q/2 \rceil, \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, where $b_{\mathsf{ID}^*} = \mathsf{H}'(\mathsf{ID}^*)$. It then rearranges the terms if necessary and returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$. We claim that this change is only conceptual. Note that the distribution of the ciphertexts in this game corresponds to that in the previous game if we flip the value of $\mathsf{H}'(\mathsf{ID}^*)$ for all challenge identities ID^*. Since $\mathcal{A}$ never makes a secret key query for ID^*, the value of $\mathsf{H}'(\mathsf{ID}^*)$ is information theoretically hidden from $\mathcal{A}$. This implies that even if we flip values of $\mathsf{H}'(\mathsf{ID}^*)$ for all challenge identities ID^*, $\mathcal{A}$ cannot notice it at all. Therefore, we have $\Pr[X_{11}] = \Pr[X_{10}]$.

Game_{12} : In this game, we further change the way the challenge ciphertexts are generated. When $\mathcal{A}$ makes a challenge query for $(\mathsf{ID}^*, \mathsf{M}^*)$, it returns a random ciphertext sampled from $\mathsf{CTSam}(\mathsf{mpk})$ if $\mathsf{coin} = 1$. If $\mathsf{coin} = 0$, it generates the ciphertext as

$$c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad c_{\bar{b}_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q, \quad \mathbf{c}_2 = \mathbf{A}^\top \mathbf{s} + \mathbf{x}.$$

for $\mathbf{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$. It then returns $(c_0, c_1, \mathbf{c}_2)$ to $\mathcal{A}$. We claim that the change is unnoticed by $\mathcal{A}$ assuming the LWE assumption. This can be shown by adding changes to Game_{11} that are almost the same as those we introduced from Game_2 to Game_6. The only difference is that $c_{b_{\mathsf{ID}^*}}$ is always sampled as $c_{b_{\mathsf{ID}^*}} \overset{\$}{\leftarrow} \mathbb{Z}_q$ here. By the similar analysis, there exists $\mathcal{B}$ such that we have $|\Pr[X_{11}] - \Pr[X_{12}]| \le n \cdot \mathsf{Adv}_{\mathcal{B},\mathsf{QRO}_{\ell_{\mathsf{ID}}+2,\max\{\ell_r,(\lfloor \log q \rfloor + 2\lambda)\times n\}}}^{\mathsf{LWE}_{\ell,m,q,D_{\mathbb{Z},\alpha q}}}(\lambda) + (Q_{\mathsf{H}} + Q_{\mathsf{ID}} + Q_{\mathsf{ch}}) \cdot 2^{-n}$.

Finally, we observe that the challenge ciphertexts are sampled from $\mathsf{CTSam}(\mathsf{mpk})$ regardless of whether $\mathsf{coin} = 0$ or 1. Therefore, we have $\Pr[X_{12}] = 1/2$. Putting things together, the theorem readily follows. $\square$

Chapter 4

Partitioning via Non-Linear Polynomial Functions

4.1 Introduction

Identity-based encryption (IBE) is a generalization of public key encryption (PKE) where the public key of a user can be any arbitrary string such as an e-mail address. The concept of IBE was first proposed by Shamir [Sha85] in 1984, but it took nearly two decades for the first realizations of IBE [SOK00, BF01, Coc01] to appear. Since then, the construction of IBE has been one of the central topics in cryptography. Nowadays, we have constructions of IBEs from assumptions on bilinear maps [BF01, BB04a, BB04b, Wat05, Gen06, Wat09], the quadratic residue assumption [Coc01, BGH07], and from the learning with error (LWE) assumption [GPV08, CHKP10, ABB10] whose hardness is implied by the worst case reductions to certain lattice problems [Reg05].

One of the most standard security definitions for IBE is the adaptive security, or often called full security. While it is not quite hard to obtain the adaptive security for an IBE in the random oracle model [BF01, Coc01, GPV08], the realization in the standard model is much harder. Roughly speaking, currently there are two general techniques in achieving adaptive security in the standard model: the partitioning technique [BB04b, Wat05] and the dual system encryption methodology [Wat09, LW10]. The latter is very attractive, because it allows us to construct very efficient IBE schemes [CW13, JR13] and even more advanced cryptosystems such as attribute-based encryptions [LOS$^+$10] with adaptive security. However, it inherently relies on *decisional assumptions* on bilinear maps (e.g., SXDH and DLIN) and cannot be extended to the proofs based on *computational assumptions* on bilinear maps (e.g., computational bilinear Diffie-Hellman (CBDH) assumption) or assumptions on lattices. On the other hand, the application of the former technique is wider. We can construct adaptively secure IBE from the CBDH assumption (by the straightforward combination of the Goldreich-Levin bit [GL89] and Waters IBE [Wat05]) and from the LWE assumption [CHKP10, ABB10, Boy10]. However, IBE schemes constructed from the former approach typically requires larger parameters due to the use of the Waters' hash [Wat05] or the admissible hash [BB04b, CHKP10].

Very recently, Yamada [Yam16] constructed IBE schemes from lattices based on the partitioning technique with novel ideas that are different from the Waters' hash or the admissible hash. His schemes achieve asymptotically shorter public parameters than previous works. One of

[0]The contents of this chapter is based on the work presented at Asiacrypt 2016 under the title "Partitioning via Non-Linear Polynomial Functions: More Compact IBEs from Ideal Lattices and Bilinear Maps" [KY16].

the drawbacks of the schemes is that they require super-polynomial size modulus for LWE. As a result, their ciphertexts are longer than those of previous works by a rather large super-constant factor. In addition, they have to assume the hardness of the LWE problem for *all polynomial* (i.e., $O(n^c)$ for *all* $c \in \mathbb{N}$) or the more aggressive *super-polynomial* approximation factor. Though their assumption is plausible, it is much stronger than those used in the previous works where the hardness of the LWE problem for some *fixed polynomial* approximation factor (i.e., $O(n^c)$ for *some* $c \in \mathbb{N}$) is assumed. Furthermore, since he used fully homomorphic computations of trapdoors [BGG+14b], a technique unique to the lattice setting, it is a highly non-trivial task to construct analogous schemes in other settings such as bilinear maps.

4.1.1 Our Contribution

In this chapter, we focus on the constructions of adaptively secure IBE in these settings where dual system encryption methodology is unavailable. In particular, we propose IBE schemes with shorter public parameters from ring/ideal lattices and from a certain computational assumption (rather than a decisional assumption) on bilinear groups, by extending and adding twists to the techniques of [Yam16]. Specifically, we obtain the following results. See Table 4.1 and 4.2 for the overview.

- We propose an anonymous and adaptively secure IBE scheme from the ring LWE (RLWE) assumption with *fixed polynomial* approximation factors, which is further reduced to certain worst case problems on ideal lattices. Note that simply instantiating Yamada's scheme using ideal lattices[1] will still require the RLWE assumption for *all polynomial* approximation factors, which is a much stronger assumption than what we use. As for the efficiency, the size of the public parameters, private keys, and ciphertexts in our scheme are $O(n\kappa^{1/d} \log n)$, $O(n \log n)$, and $O(n \log n)$, respectively. Here, n is the dimension of the ring elements, κ is the length of the identities, and d is a flexible constant that can be set arbitrary, but will affect the reduction cost exponentially. We note that all of them achieve the best efficiency among the other adaptively secure IBE from the RLWE assumption in an asymptotic sense. Compared to the ring version of Yamada's scheme, we managed to reduce the poly-logarithmic factors contained in the public parameters, private keys, and ciphertexts.

- We propose a (non anonymous and) adaptively secure IBE scheme from the 3-computational bilinear Diffie-Hellman exponent (3-CBDHE) assumption. The 3-CBDHE assumption is a weaker variant of the n-decisional bilinear Diffie-Hellman exponent (n-DBDHE) assumption [BBG05, BGW05, BH08]. The former seems to be much a weaker assumption than the latter in two aspects. First, the former is a computational assumption whereas the latter is a decisional assumption. Second, the former is not a parameterized assumption, in the sense that the size of the problem instance only depends on the security parameter. As for the efficiency, the public parameters, private keys, and ciphertexts in our scheme require $O(\sqrt{\kappa})$ group elements. Here, κ is the length of the identities. This is the first adaptively secure IBE scheme from a computational assumption on bilinear groups with public parameters consisting of sub-linear number of group elements in the length of the identities. However, we note that the sizes of the ciphertexts and private keys of our scheme are larger than the previous schemes.

[1]Note that he does not describe nor mention the ring variant of the scheme. However, we can convert his scheme into a ring variant in a straightforward manner as is the case in most previous works [CHKP10, ABB10, Boy10].

We emphasize that our result for the lattice based construction cannot be obtained through the simple switch to the ring setting in Yamada's scheme. Their proof will still require a super-polynomial-size modulus to work, whereas our new technique allows for a polynomial-size modulus. In addition, the security proof of our scheme requires new ideas that did not appear in [Yam16]. It exploits the commutative properties of the underlying ring elements in an essential way, involves a more generalized partitioning argument, and a careful analysis of the Gaussian error. Refer Section 4.2 for the technical overview. We note that the public parameter of our second scheme could be further reduced to $O(\kappa^{1/d})$ assuming the $d+1$-CBDHE assumption. However, it would come at the cost of even longer ciphertexts and complicated description of the scheme. This is beyond the scope of our work. We finally remark that the reduction costs for both of our schemes are inadmissible as was in the case of [Yam16]. In fact, the reduction loss for the first scheme is worse than [Yam16]. Improving them is left as an open problem.

Related Works. One way to reduce the size of the public parameters in Waters' hash and its analogue is to use Naccache's approach [Nac07, SPB12]. However, with this approach, we are only allowed to reduce the size of public parameters up to logarithmic factor. Ducas et al. [DLP14] constructed efficient IBE over NTRU lattices in the random oracle model. Gentry [Gen06] proposed adaptively secure IBE with compact parameters from a parameterized (or q-type) assumption on bilinear maps. Galindo [Gal10] and Chen et al. [CCZ11] proposed selectively secure CCA-secure IBE schemes from the CBDH assumption.

Note on Recent Works. Here, we mention two important recent related works.

Apon et al. [AFL16] proposed an adaptively secure IBE scheme from lattices whose parameters are very compact, using collision resistant hash function with output-length $\kappa = \omega(\log \lambda)$. Here, λ is the security parameter. While their scheme is more efficient than our scheme, we clarify that they implicitly assume exponential security on the collision resistant hash function, which is a stronger assumption than what we use. To demonstrate this, let us set $\kappa = \log^2 \lambda$. If there is no better attack than the birthday attack against the hash function, no PPT adversary can find a collision with more than negligible probability. On the other hand, the existence of even a sub-exponential time attack would compromise the security of the IBE. For example, assume that there exists an attack that finds a collision in time $2^{\sqrt{\kappa}}$. Then, the collision for the hash can be found in linear time in λ, since $2^{\sqrt{\kappa}} = 2^{\log \lambda} = \lambda$.

In their very recent work, Zhang et al. [ZCZ16] constructed an IBE scheme with poly-logarithmic public parameters. While their scheme achieves better asymptotic space efficiency than our scheme, their scheme is Q-bounded, in the sense that the security of the scheme is not guaranteed any more if the adversary obtains more than Q private keys. This restriction cannot be removed by just making Q super-polynomial, because the running time of the encryption algorithm in their scheme is at least linear in Q. We note that our scheme is secure against an unbounded collusion.

4.2 Technical Overview

4.2.1 Construction from Ring and Ideal Lattices

The Yamada IBE. We briefly review the Yamada IBE [Yam16], for our proposed IBE scheme follows the framework of theirs and overcomes some of the major problems posed by their construction. Their construction follows the general framework of constructing lattice-based IBE schemes that associates to each identity ID the matrix $[\mathbf{A}|\mathsf{H}(\mathsf{ID})] \in \mathbb{Z}_q^{n\times 2m}$. In previous IBE constructions [ABB10, CHKP10], the function $\mathsf{H}(\mathsf{ID})$ was computed by using the rather long κ

public matrices $\{\mathbf{B}_i\}_{i\in[\kappa]}$, where $\kappa = O(n)$ is the length of the identities. The main technical contribution of the Yamada IBE was in reducing the size of the public matrices to $\kappa^{1/d}$ for any constant d and hence reducing the size of the public parameters by incorporating a primitive called fully homomorphic trapdoor functions. Hereafter, we consider the case $d = 2$ for simplicity. In detail, they used an injective map $S : \{0,1\}^\kappa \to 2^{[\ell]\times[\ell]}$ that maps an identity to a subset of the set $[\ell] \times [\ell]$ where $\ell = \lceil \kappa^{1/2} \rceil$, and computed the function $\mathsf{H}(\mathsf{ID})$ as

$$\mathsf{H}(\mathsf{ID}) = \mathbf{B}_0 + \sum_{(i,j)\in S(\mathsf{ID})} \mathbf{B}_{1,i} \cdot \mathbf{G}^{-1}(\mathbf{B}_{2,j}) \tag{4.1}$$

where the number of public matrices $\mathbf{B}_0, \{\mathbf{B}_{i,j}\}_{(i,j)\in[2]\times[\ell]}$ are now reduced to $O(\kappa^{1/2})$. Here, $\mathbf{G}$ is a special gadget matrix whose trapdoor is publicly known [MP12] and $\mathbf{G}^{-1}$ is viewed as a deterministic function rather than a matrix, that maps a matrix $\mathbf{V} \in \mathbb{Z}_q^{n\times m}$ to a matrix $\mathbf{U} \in \{0,1\}^{m\times m}$ such that $\mathbf{GU} = \mathbf{V} \mod q$.

During the security proof, the reduction algorithm first prepares random integers $y_0, \{y_{i,j}\}_{(i,j)\in[2]\times[\ell]} \in \mathbb{Z}_q$ from certain domains whose size grows linear in the number of key extraction query Q of the adversary. Then after sampling $\mathbf{R}_0, \{\mathbf{R}_{i,j}\}_{i\in[2],j\in[\ell]} \in \mathbb{Z}^{m\times m}$ with small spectral norm, the reduction algorithm prepares the public parameters as

$$\mathbf{B}_0 = \mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}, \quad \mathbf{B}_{i,j} = \mathbf{A}\mathbf{R}_{i,j} + y_{i,j}\mathbf{G} \tag{4.2}$$

for $(i,j) \in [2] \times [\ell]$. Then during the security reduction the hash value for identity ID Eq.(4.1) is computed as

$$\begin{aligned}
\mathsf{H}(\mathsf{ID}) &= (\mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}) + \sum_{(i,j)\in S(\mathsf{ID})} (\mathbf{A}\mathbf{R}_{1,i} + y_{1,i}\mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{B}_{2,j}) \\
&= (\mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}) + \sum_{(i,j)\in S(\mathsf{ID})} (\mathbf{A}\mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + y_{1,i}\mathbf{B}_{2,j}) \\
&= (\mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}) + \sum_{(i,j)\in S(\mathsf{ID})} \big(\mathbf{A}\mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + y_{1,i}(\mathbf{A}\mathbf{R}_{2,j} + y_{2,j}\mathbf{G})\big) \\
&= (\mathbf{A}\mathbf{R}_0 + y_0\mathbf{G}) + \sum_{(i,j)\in S(\mathsf{ID})} \big(\mathbf{A}\mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + \mathbf{A}(y_{1,i}\mathbf{R}_{2,j}) + y_{1,i}y_{2,j}\mathbf{G}\big) \\
&= \underbrace{\mathbf{A}\left(\mathbf{R}_0 + \sum_{(i,j)\in S(\mathsf{ID})} \big(\mathbf{R}_{1,i}\mathbf{G}^{-1}(\mathbf{B}_{2,j}) + y_{1,i}\mathbf{R}_{2,j}\big)\right)}_{:=\mathbf{R}_{\mathsf{ID}},\text{ which is ``small''}} + \underbrace{\left(y_0 + \sum_{(i,j)\in S(\mathsf{ID})} y_{1,i}y_{2,j}\right)}_{:=\mathsf{F}_{\mathbf{y}}(\mathsf{ID})} \cdot \mathbf{G} \\
&= \mathbf{A}\mathbf{R}_{\mathsf{ID}} + \mathsf{F}_{\mathbf{y}}(\mathsf{ID})\mathbf{G}.
\end{aligned} \tag{4.3}$$

Observe that we implicitly relied on the fact that $\mathbf{A}$ and $y_{1,i}$ commutes. Therefore, the reduction algorithm is able to sample a secret key for ID using the trapdoor of $\mathbf{G}$ if and only if $\mathsf{F}_{\mathbf{y}}(\mathsf{ID}) \neq 0 \mod q$. Hence, the simulation succeeds when the adversary queries on secret keys for ID satisfying $\mathsf{F}_{\mathbf{y}}(\mathsf{ID}) \neq 0 \mod q$, and queries for a challenge ciphertext for $\mathsf{ID}^\star$ satisfying $\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star) = 0 \mod q$ in which case the reduction algorithm can embed its LWE challenge.

Overview of the Construction and Security Proof. The major drawback of the Yamada IBE is that they require the modulus size q to be super-polynomial. This stems from the fact that the size of $y_0, y_{i,j} \in \mathbb{Z}_q$ must grow linearly in the number of adversarial key extraction query

Q for the security proof to be meaningful, i.e., $\Pr_{\mathbf{y}}[\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star) = 0 \wedge \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_1) \neq 0 \wedge \cdots \wedge \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_Q) \neq 0]$ is noticeable in n. However, since the size of the $\mathbf{G}$-trapdoor $\mathbf{R}_{\mathsf{ID}}$ used during simulation grows proportionally to the size of $y_{1,i}$ (check above Eq.(4.3) to see how $\mathbf{R}_{\mathsf{ID}}$ was created), thereby growing proportional to $Q = \mathsf{poly}(n)$, we need to set the modulus size q to be at least super-polynomial in n for the trapdoor to operate properly. Therefore, if we try to restrict ourselves to a polynomial sized modulus q, it seems the best we can achieve is a scheme where we have to set a bound on the number of adversarial key extraction queries before instantiation, i.e., a Q-bounded scheme.

In our work, we combine several ideas in a novel way to circumvent the above seemingly inevitable problem. The first idea is to extend the elements $y_0, y_{i,j} \in \mathbb{Z}_q$ to matrices $\mathbf{Y}_0, \mathbf{Y}_{i,j} \in \mathbb{Z}_q^{n \times n}$ so that instead of increasing the size of the element $y \in \mathbb{Z}_q$, we can "pack" small elements in the entries of the matrix $\mathbf{Y} \in \mathbb{Z}_q^{n \times n}$. Namely, since the matrix has n^2 entries, if the number of key extraction query is $Q = n^c$ for some constant c, we can always set up the matrix so that c of the entries are packed by elements of size $O(n)$. Since there are n^2 entries in total, this allows us to pack the matrix with small entries (e.g., $O(n)$) for arbitrary $Q = \mathsf{poly}(n)$ without the need of increasing the modulus size q. However, this simple idea alone does not work, since during the security proof to obtain Eq.(4.3), we crucially relied on the fact that $\mathbf{A}$ and $y_{1,i}$ commutes. For our idea to work we need the two matrices $\mathbf{A}$ and $\mathbf{Y}_{1,i}$ to commute, however, in general this does not hold.

To overcome this problem, we introduce our second idea of using the ring structure of ideal lattices. Concretely, we use the special polynomial ring $R = \mathbb{Z}[X]/(X^n + 1)$ to construct our scheme for n a power of 2. The construction itself is exactly the same as the ring analogue of the Yamada IBE, however, our new security proof relies crucially on the underlying ring structure. In detail, the reduction algorithm prepares the public parameters as

$$\boldsymbol{b}_0 = \boldsymbol{a}\boldsymbol{R}_0 + y_0\boldsymbol{g}, \quad \boldsymbol{b}_{i,j} = \boldsymbol{a}\boldsymbol{R}_{i,j} + y_{i,j}\boldsymbol{g} \tag{4.4}$$

for $(i,j) \in [2] \times [\ell]$, where $\boldsymbol{a}, \boldsymbol{b}_0, \boldsymbol{b}_{i,j} \in R_q^k$, $\boldsymbol{R} \in R_q^{k \times k}$, $y_0, y_{i,j} \in R_q$ and $\boldsymbol{g} \in R_q^k$ is the ring analogue of the $\mathbf{G}$-trapdoor. Observe that $y_0, y_{i,j}$ are now elements in R_q instead of $\mathbb{Z}_q$. Although this y is not quite a matrix, this is actually more than enough for us to use the packing technique described above. This can be seen by first noticing the natural isomorphism between $R_q \cong \mathbb{Z}_q^n$ induced by the coefficient embedding and viewing $y \in R_q$ as a vector in $\mathbb{Z}_q^n$. Since y has n entries when viewed as vectors, it can support up to n^n queries by packing each entry with small elements of size $O(n)$. Furthermore, the second part of the problem addressed above is naturally resolved, since now that we are working in a ring we get the commutativity of $\boldsymbol{a}$ and $y_{1,j}$ for free. This key role in the commutativity for rings is somewhat reminiscent to the signature scheme of [DM14]. We note that the technique used by [AS15] (which has also been used in [Xag13]) to extend the results of [DM14] to matrices seems to be inapplicable in our setting. This is because in our setting we need to commute the LWE challenge matrix $\mathbf{A}$ instead of the gadget matrix $\mathbf{G}$ whose associating trapdoor is known. To summarize, by incorporating our second idea, we obtain the ring variant of Eq.(4.3) and the trapdoor operates as specified. We note that one might be tempted to pack the entries of y with constant size elements, since 2^n is still exponential in n and hence $Q(n) < 2^n$. However, the security proof relies heavily on the fact that the density (i.e., the number of entries that are packed) of y is bounded by some constant. Therefore, we must choose the size of the packed elements with care to make the overall scheme secure.

The final idea is carefully crafting a properly distributed challenge ciphertext. To be precise, the main issue is in the difficulty of creating a ciphertext that has errors that are properly distributed. This problem of generating a properly distributed challenge ciphertext was addressed in [Yam16] as well, however, they used the standard technique called the "smudging" or "noise flooding" technique which came at the cost of making the modulus size q super-polynomial in n. This was not a problem for them, since as we pointed out earlier, their scheme inherently needed a super-polynomial sized modulus to work. However, this tactic is inapplicable to our setting since we want to restrict ourselves to the polynomial sized modulus. To overcome this we devise a way to carefully craft the error term; a technique reminiscent of [GPV08, ACPS09]. First, assume we have $\mathsf{F}(\mathsf{ID}^\star) = 0$ for the challenge identity $\mathsf{ID}^\star$ and thus $\mathsf{H}(\mathsf{ID}) = \mathbf{A}\mathbf{R}_{\mathsf{ID}^\star}$. Note that for ease of understanding we explain the technique in the matrix form instead of the ring form. To prove security, we have to embed the LWE challenge $\mathbf{A}$ and $\mathbf{v}$ into the challenge ciphertext, where $\mathbf{v} = \mathbf{s}\mathbf{A} + \mathbf{x}$ or $\mathbf{v}$ a random vector. One natural way is to set

$$\mathbf{x}_1 = \mathbf{x}, \quad \mathbf{x}_2 = \mathbf{x}\mathbf{R}_{\mathsf{ID}^\star} \tag{4.5}$$

and compute the challenge ciphertext as

$$\mathbf{s}[\mathbf{A}|\mathsf{H}(\mathsf{ID}^\star)] + [\mathbf{x}_1|\mathbf{x}_2] = [\mathbf{v}|\mathbf{v}\mathbf{R}_{\mathsf{ID}^\star}].$$

However, one can not simply use the standard generalized leftover hash lemma for lattices presented in [ABB10]; a technique often used in proving such forms. This is because $\mathbf{R}_{\mathsf{ID}^\star}$ is not uniformly sampled as in the case of [ABB10], but instead highly correlated to the values of y, $\{y_{i,j}\}$ used during the simulation. Alternatively, we present a noise rerandomization technique and add a small extra noise to Eq.(4.5) and statistically hide $\mathbf{R}_{\mathsf{ID}}$. Namely, we sample noises $\mathbf{e}_1$ and $\mathbf{e}_2$ from a particular Gaussian distribution with variance computed from $\mathbf{R}_{\mathsf{ID}^\star}$ and set

$$\mathbf{x}_1 = \mathbf{x} + \mathbf{e}_1, \quad \mathbf{x}_2 = \mathbf{x}\mathbf{R}_{\mathsf{ID}^\star} + \mathbf{e}_2. \tag{4.6}$$

Thus the challenge ciphertext is created as above by further adding the new noise terms. Although the general idea of this technique has been around since [Reg05, GPV08] and has been used in contexts elsewhere, as far as we know, we believe this is a nice application for rerandomizing the noise without the need of adding a huge (super-polynomial sized) noise.

An Additional Idea. Working in the ring setting introduces some subtle yet crucial obstacles, which we did not have to address before. Namely, for q a prime and n a power of 2, the domain $R_q = \mathbb{Z}[X]/(q, X^n + 1)$ we work in is no longer a field as in the case of $\mathbb{Z}_q$. Additionally, if we use a modulus q such that $q \equiv 1 \mod 2n$ as in [LPR10, LPR13], the ring R_q completely splits into n fields. In such a ring, each field only contains $q = \mathsf{poly}(n)$ elements so the Schwartz-Zippel lemma during our security proof can not be applied. We get around this by using a modulus q such that $q \equiv 3 \mod 8$ where it is known to split into only two fields. Then, since each field now contains $q^{n/2}$ elements and R_q acts roughly as a field, we are able to apply our proof techniques. We finally note that we also obtain a nice regularity lemma over such rings which helps us attain better parameters for the scheme.

We also employ some ideas to further optimize the sizes of the public parameters, secret keys and ciphertexts. Namely, we use the (ring version of the) $\mathbf{G}$-trapdoor where the base is set as n^η for some positive constant η. We use $\eta = \frac{1}{4}$ for our concrete parameter selection. By incorporating this idea, we can further reduce the size of the parameters by a factor of $\log n$. However, this comes at the cost of making the scheme less efficient, since the function $\mathbf{G}^{-1}(\cdot)$ has a slower running time for a larger base.

4.2.2 Construction from Bilinear Maps

Here, we explain our IBE scheme from bilinear maps. We start with a slightly modified version of Waters IBE [Wat05] and gradually modify it to obtain our scheme. Let us consider a group

$\mathbb{G}$ with prime order p whose generator is g. The group is equipped with a efficiently computable bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. The public parameters of the scheme contains rather long $\kappa + 3$ group elements $\{g^{w_i}\}_{i\in[0,\kappa]}$, g^{α}, g^{β}, and a randomness $\mathsf{rand} \in \{0,1\}^{|\mathbb{G}_T|}$ that is used to derive the Goldreich-Levin hardcore bit function $\mathsf{GL} : \{0,1\}^{|\mathbb{G}_T|} \times \{0,1\}^{|\mathbb{G}_T|} \to \{0,1\}$. The form of the ciphertexts and private keys in the scheme are as follows:

$$C \;=\; \left(g^{s},\ g^{s\mathsf{H}(\mathsf{ID})},\ \mathsf{GL}\left(e(g^{\alpha},g^{\beta})^{s},\mathsf{rand}\right) \oplus \mathsf{M} \right), \quad \mathsf{sk}_{\mathsf{ID}} = \left(g^{\alpha\beta}\cdot g^{r\mathsf{H}(\mathsf{ID})},\ g^{-r} \right)$$

where $\mathsf{M} \in \{0,1\}$ is the message to be encrypted, and s and r are random elements in $\mathbb{Z}_p$ that are picked during the encryption and key generation algorithms, respectively.

Here, $\mathsf{H} : \{0,1\}^{\kappa} \to \mathbb{Z}_p$ is defined as $\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{\mathsf{ID}_i=1} w_i$ where ID_i is the i-th bit of ID. The reason why we use the hardcore bit function is to base the security of the scheme on the *computational* bilinear Diffie-Hellman (CBDH) assumption, rather than the stronger *decisional* bilinear Diffie-Hellman (DBDH) assumption which was used to prove the security of the original Waters IBE.

Next, we try to reduce the size of the public parameters using the idea of the Yamada IBE. A natural way to do this would be to introduce the injective map $S : \{0,1\}^{\kappa} \to 2^{[\ell]\times[\ell]}$ with $\ell = \lceil \kappa^{1/2} \rceil$, change the public parameters to be $g^{w_0}, \{g^{w_{i,j}}\}_{(i,j)\in[2]\times[\ell]}$, and modify the function H as

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j)\in S(\mathsf{ID})} w_{1,i}w_{2,j}. \tag{4.7}$$

Through this change, we can reduce the size of the public parameters from $O(\kappa)$ group elements to $O(\sqrt{\kappa})$, just in as [Yam16]. However, we come across an immediate problem: We cannot efficiently compute $g^{s\mathsf{H}(\mathsf{ID})}$ from the public parameters! A straightforward solution to this problem is to put "helper" terms $\{g^{w_{1,i}w_{2,j}}\}$ into the public parameters. However, this makes the size of the public parameters large again.

Our solution to this problem is to rely on the Boneh-Boyen technique [BB04a] to compute something similar to the problematic term. Namely, we compute

$$g^{s\mathsf{H}(\mathsf{ID})+\sum_{j\in S(\mathsf{ID})} \tilde{t}_j w_{2,j}}, \qquad \{\ g^{\tilde{t}_j}\ \}_{j\in[\ell]} \tag{4.8}$$

instead of computing only $g^{s\mathsf{H}(\mathsf{ID})}$. Here, $\{\tilde{t}_j\}$ are additional randomness introduced by the encryption algorithm. Accordingly, we change the form of the ciphertexts and private keys of our scheme as follows:

$$\begin{aligned} C \;&=\; \left(g^{s},\ g^{s\mathsf{H}(\mathsf{ID})+\sum_{j\in[\ell]} \tilde{t}_j w_{2,j}},\ \{g^{\tilde{t}_j}\}_{j\in[\ell]},\ \mathsf{GL}\left(e(g^{\alpha},g^{\beta})^{s},\mathsf{rand}\right)\oplus \mathsf{M} \right), \\ \mathsf{sk}_{\mathsf{ID}} \;&=\; \left(g^{\alpha\beta}\cdot g^{r\mathsf{H}(\mathsf{ID})},\ g^{-r},\ \{g^{rw_{2,j}}\}_{j\in[\ell]} \right). \end{aligned} \tag{4.9}$$

Note that although the size of the public parameters is smaller than the original scheme, the sizes of the ciphertexts and private keys are larger due to the additional terms. We now show that one can efficiently compute the ciphertext. In particular, we show that it is possible to generate the terms in Eq.(4.8). To see this, let us introduce the variables $\{t_j\}$ such that

$$\tilde{t}_j := t_j - s\left(\sum_{i\in\{i\in[1,\ell] \mid (i,j)\in S(\mathsf{ID})\}} w_{1,i} \right). \tag{4.10}$$

Then, we have

$$\begin{aligned}
& s\mathsf{H}(\mathsf{ID}) + \sum_{j\in[\ell]} \tilde{t}_j w_{2,j} \\
= \;& s\mathsf{H}(\mathsf{ID}) + \sum_{j\in[\ell]} w_{2,j}\left(t_j - s\left(\sum_{i\in\{i\in[1,\ell] \mid (i,j)\in S(\mathsf{ID})\}} w_{1,i}\right)\right) \\
= \;& s\mathsf{H}(\mathsf{ID}) + \sum_{j\in[\ell]} w_{2,j}t_j - s\sum_{j\in[\ell]}\left(\sum_{i\in\{i\in[1,\ell] \mid (i,j)\in S(\mathsf{ID})\}} w_{1,i}w_{2,j}\right) \\
= \;& sw_0 + s\cancel{\sum_{(i,j)\in S(\mathsf{ID})} w_{1,i}w_{2,j}} + \sum_{j\in[\ell]} w_{2,j}t_j - s\cancel{\sum_{(i,j)\in S(\mathsf{ID})} w_{1,i}w_{2,j}} \\
= \;& sw_0 + \sum_{j\in[\ell]} w_{2,j}t_j. \qquad (4.11)
\end{aligned}$$

Since Eq.(4.10) and (4.11) are linear in w_0, $w_{i,j}$, it can be seen that the terms in Eq.(4.8) can be computed efficiently, as desired.

By substituting $\tilde{t}_j$ in Eq.(4.9) with the right-hand side of Eq.(4.8), we obtain our final scheme. As for the security, we can prove the adaptive security of the scheme from the 3-computational bilinear Diffie-Hellman exponent (3-CBDHE) assumption. We need to rely on this stronger assumption than the standard CBDH assumption, because of the different algebraic structure incorporated by the modified Waters IBE.

4.3 Preparation

4.3.1 Homomorphic Computation

Let d be a natural number. We introduce the function $\mathsf{PubEval}_d : (R_q^k)^d \to R_q^k$ as in [Yam16], which takes a set of vectors $\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_d \in R_q^k$ as inputs and outputs a vector in R_q^k. This function wil be used to hash identities to R_q^k in our lattice-based IBE construction. The function is defined recursively as follows:

$$\mathsf{PubEval}_d(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d) = \begin{cases} \boldsymbol{b}_1 & \text{if } d = 1 \\ \boldsymbol{b}_1 \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) & \text{if } d \geq 2. \end{cases}$$

Lemma 4.1. *Let $y_1, \ldots, y_d$ be elements in R, $\boldsymbol{a}, \boldsymbol{b}_1, \ldots, \boldsymbol{b}_d$ be vectors in R_q^k and $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_d$ be matrices in $R^{k\times k}$ such that $\boldsymbol{b}_i = \boldsymbol{a}\boldsymbol{R}_i + y_i\boldsymbol{g}_b$ for $i \in [d]$. Furthermore, we assume that $s_1(\boldsymbol{R}_i) \leq B, \|\phi(y_i)\|_1 \leq \delta$ for $i \in [d]$. Then, there exists an efficient algorithm $\mathsf{TrapEval}_d$ that takes $\boldsymbol{R}_1, \ldots, \boldsymbol{R}_d$, $y_1, \ldots, y_d$ as inputs and outputs $\boldsymbol{R}' \in R^{k\times k}$ such that*

$$\mathsf{PubEval}_d(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d) = \boldsymbol{a}\boldsymbol{R}' + y_1 \cdots y_d \boldsymbol{g}_b \in R_q^k \qquad (4.12)$$

and $s_1(\boldsymbol{R}') \leq B\delta^{d-1} + Bbnk\big(\frac{\delta^{d-1}-1}{\delta-1}\big)$.

Proof. Before starting the actual proof for the above lemma, we state the following lemma that provides us with a useful bound for the singular value of a single element in R.

Lemma 4.2 ([DM14], Lemma 5)**.** *For any ring element $a \in R$, we have $s_1(a) \leq \|\phi(a)\|_1$.*

Then, the proof of Lemma 4.1 is given as follows. We prove it by induction. The base case (the case of $d = 1$) is trivial. Therefore, let us assume the hypothesis for $d-1$ where $d \geq 2$. Then, we have

$$s_1(\boldsymbol{R}_0) \leq B\delta^{d-2} + Bbnk\Big(\frac{\delta^{d-2}-1}{\delta-1}\Big) \quad \text{and} \quad \mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d) = \boldsymbol{a}\boldsymbol{R}_0 + y_2 \cdots y_d \boldsymbol{g}_b$$

for efficiently computable $\boldsymbol{R}_0$. Therefore, by the definition of $\mathsf{PubEval}_d$, we have

$$\begin{aligned}
& \mathsf{PubEval}_d(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_d) \\
= \; & (\boldsymbol{a}\boldsymbol{R}_1 + y_1\boldsymbol{g}_b) \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) \\
= \; & \boldsymbol{a}\boldsymbol{R}_1 \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) + y_1 \cdot \mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d) \\
= \; & \boldsymbol{a}\boldsymbol{R}_1 \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) + y_1(\boldsymbol{a}\boldsymbol{R}_0 + y_2 \cdots y_d \boldsymbol{g}_b) \\
= \; & \boldsymbol{a}(\boldsymbol{R}_1 \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) + y_1\boldsymbol{R}_0) + y_1 y_2 \cdots y_d \boldsymbol{g}_b.
\end{aligned}$$

It can be seen that Eq.(4.12) holds by setting

$$\boldsymbol{R}' = \boldsymbol{R}_1 \cdot \boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big) + y_1\boldsymbol{R}_0.$$

It is clear that it can be efficiently computable. Furthermore, we have

$$\begin{aligned}
s_1(\boldsymbol{R}') & \leq s_1(\boldsymbol{R}_1) \cdot s_1\Big(\boldsymbol{g}_b^{-1}\big(\mathsf{PubEval}_{d-1}(\boldsymbol{b}_2, \ldots, \boldsymbol{b}_d)\big)\Big) + s_1(y_1) \cdot s_1(\boldsymbol{R}_0) \\
& \leq B \cdot bnk + \|\phi(a)\|_1 \cdot s_1(\boldsymbol{R}_0) \\
& \leq Bbnk + \delta\Big(B\delta^{d-2} + Bbnk\Big(\frac{\delta^{d-2}-1}{\delta-1}\Big)\Big) \\
& = B\delta^{d-1} + Bbnk\Big(\frac{\delta^{d-1}-1}{\delta-1}\Big).
\end{aligned}$$

The second inequality follows from Lemma 4.2 and the fact that $s_1(\boldsymbol{g}_b^{-1}(\boldsymbol{u})) \leq bnk$ holds for any $\boldsymbol{u} \in R_q^k$. □

Lemma 4.3 (Expansion of Coefficients). *Let $c_1, c_2, B_1, B_2 \in \mathbb{N}$. Let also $u = u_0 + u_1X + \cdots u_{c_1-1}X^{c_1-1} \in R$ and $v = v_0 + v_1X + \cdots v_{c_2-1}X^{c_2-1} \in R$ be ring elements. We further assume that $c_1 + c_2 < n$ and $\|\phi(u)\|_\infty < B_1$ and $\|\phi(v)\|_\infty < B_2$. Then we have $\|\phi(uv)\|_\infty \leq \min\{c_1, c_2\} \cdot B_1B_2$.*

Proof. We have

$$\begin{aligned}
\|\phi(uv)\|_\infty & = \left\| \phi\left(\sum_{j=0}^{c_1+c_2-2} \left(\sum_{i=\max\{0,j+1-c_2\}}^{\min\{c_1-1,j\}} u_i v_{j-i} \right) X^j \right) \right\|_\infty \\
& = \max_{j \in [0, c_1+c_2-2]} \left\{ \sum_{i=\max\{0,j+1-c_2\}}^{\min\{c_1-1,j\}} u_i v_{j-i} \right\} \\
& \leq \min\{c_1, c_2\} B_1 B_2
\end{aligned}$$

where the last equation follows from $\|\phi(u)\|_\infty \leq B_1$, $\|\phi(v)\|_\infty \leq B_2$, and $\min\{c_1-1, j\} + 1 - \max\{0, j+1-c_2\} \leq \min\{(c_1-1)+1-0, j+1-(j+1-c_2)\} = \min\{c_1, c_2\}$. □

4.3.2 Lower Bounds for the Advantage of Adversaries

The following Lemma addresses a general statement for bounding the success probability of an adversary engaging with the security game of IBE. In more detail, when the partitioning technique is used to prove security, the guess returned by the adversary is correlated with the key extraction queries it has made. Therefore, we need to argue with care to obtain a meaningful bound on the success probability that holds for arbitrary key extraction queries.

Lemma 4.4 (Implicit in [BR09, Yam16]). *Let us consider an IBE scheme and an adversary $\mathcal{A}$ that breaks adaptive security (adaptively-anonymous security) with advantage ϵ. Let us also consider a map γ that maps a sequence of identities to a value in $[0,1]$. We consider the following experiment. We first execute the security game for $\mathcal{A}$. Let $\mathsf{ID}^\star$ be the challenge identity and $\mathsf{ID}_1, \ldots, \mathsf{ID}_Q$ be the identities for which key extraction queries were made. We denote $\mathbb{ID} = (\mathsf{ID}^\star, \mathsf{ID}_1, \ldots, \mathsf{ID}_Q)$. At the end of the game, we set $\mathsf{coin}' \in \{0,1\}$ as $\mathsf{coin}' = \widehat{\mathsf{coin}}$ with probability $\gamma(\mathbb{ID})$ and $\mathsf{coin}' \stackrel{\$}{\leftarrow} \{0,1\}$ with probability $1-\gamma(\mathbb{ID})$. Then, the following holds.*

$$\left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| \geq \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$

where $\gamma_{\min}$ (resp. $\gamma_{\max}$) is the maximum (resp. minimum) of $\gamma(\mathbb{ID})$ taken over all possible $\mathbb{ID}$.

Proof. For $\mathbb{ID} = (\mathsf{ID}^\star, \mathsf{ID}_1, \ldots, \mathsf{ID}_Q)$, we define $\mathsf{Q}(\mathbb{ID})$ as the event that $\mathcal{A}$ chooses $\mathsf{ID}^\star$ as its challenge identity and it makes key extraction queries for $\mathsf{ID}_1, \ldots, \mathsf{ID}_Q$. We also define $\mathsf{Replace}$ as the event that coin' is set as $\mathsf{coin}' \stackrel{\$}{\leftarrow} \{0,1\}$. Then, we have

$$\begin{aligned}
&\left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| \\
=\ &\left|\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Pr[\mathsf{coin}' = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\right| && (4.13)\\
=\ &\left|\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\mathsf{coin}' = \mathsf{coin} \wedge \neg\mathsf{Replace} | \mathsf{Q}(\mathbb{ID})] \right.\\
&\qquad\qquad \left. + \Pr[\mathsf{coin}' = \mathsf{coin} \wedge \mathsf{Replace} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\Big)\right| && (4.14)\\
=\ &\left|\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] \cdot \gamma(\mathbb{ID}) + \frac{1}{2} \cdot (1 - \gamma(\mathbb{ID})) - \frac{1}{2}\Big)\right| && (4.15)\\
=\ &\left|\sum_{\mathbb{ID}} \gamma(\mathbb{ID}) \cdot \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\Big)\right| && (4.16)\\
\geq\ &\left|\sum_{\mathbb{ID}} \gamma_{\min} \cdot \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\Big)\right| \\
&\qquad - \left|\sum_{\mathbb{ID}} (\gamma(\mathbb{ID}) - \gamma_{\min}) \cdot \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\Big)\right| && (4.17)\\
\geq\ &\gamma_{\min}\left|\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] \cdot \Big(\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathbb{ID})] - \frac{1}{2}\Big)\right| - \frac{\gamma_{\max} - \gamma_{\min}}{2}\left|\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})]\right| && (4.18)
\end{aligned}$$

$$= \gamma_{\min} \left| \Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2} \right| - \frac{\gamma_{\max} - \gamma_{\min}}{2} \tag{4.19}$$

$$= \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2} \tag{4.20}$$

where the sum is taken over all possible $\mathbb{ID}$ (i.e., $\mathbb{ID}$ with $\mathsf{Q}(\mathbb{ID}) > 0$). In the above, Eq.(4.13) follows by the law of total probability, Eq.(4.14) follows from the law of total probability and $\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] = 1$, Eq.(4.15) follows from the fact that the probability of $\mathsf{Replace}$ is $\gamma(\mathbb{ID})$, when conditioned on $\mathsf{Q}(\mathbb{ID})$ (regardless of the value of $\widehat{\mathsf{coin}}$), Eq.(4.16) is trivial, Eq.(4.17) follows from the triangle inequality, Eq.(4.18) holds since $\gamma(\mathbb{ID}) \le \gamma_{\max}$ and $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin} | \mathsf{Q}(\mathsf{ID})] - 1/2| \le 1/2$, Eq.(4.19) follows again from $\sum_{\mathbb{ID}} \Pr[\mathsf{Q}(\mathbb{ID})] = 1$, and Eq.(4.20) is by the definition of ϵ. $\square$

Injective map. Let d and κ be some integers. Furthermore, let ℓ be $\ell = \lceil \kappa^{1/d} \rceil$. Then, an element of $[1, \kappa]$ can be written as an element of $[1, \ell]^d$ using some canonical map. Furthermore, it is also possible to write a subset of $[1, \kappa]$ as a subset of $[1, \ell]^d$ by naturally extending the canonical map. By identifying a bit string in $\{0,1\}^\kappa$ with a subset of $[1, \kappa]$ (for example, by regarding the former as the indicator vector of a subset of $[1, \kappa]$), we can define an efficiently computable injective map S that maps a bit string $\mathsf{ID} \in \{0,1\}^\kappa$ to a subset $S(\mathsf{ID})$ of $[1, \ell]^d$.

4.3.3 Core Lemma for Our Partitioning

We make a general statement concerning the partitioning technique for IBEs, which we use during the security analysis for both our lattice and bilinear map based constructions. Namely, we use the following Lemma in order to argue that the probability of the hash value for identities corresponding to the key extraction queries being invertible and the hash value for the challenge identity being zero is non-negligible.

Lemma 4.5. *Let $\nu, \mu, d, Q \ge 1$ be any integers. Let Φ be a ring and $\Omega_1, \ldots, \Omega_\nu$ be a set of fields equipped with homomorphisms $\pi_j : \Phi \to \Omega_j$ for $j \in [\nu]$. Assume that the map Π defined as $\Pi : \Phi \ni y \mapsto (\pi_1(y), \ldots, \pi_\nu(y)) \in \Omega_1 \times \cdots \times \Omega_\nu$ is an isomorphism. Let S_0 and S_1 be subsets of Φ with finite cardinality. Let us consider a set of multivariate polynomials $f_i(Y_1, \ldots, Y_\mu) \in \Phi[Y_1, \ldots, Y_\mu]$ for $i \in [0, Q]$ We further assume the following properties:*

1. *The map π_j is injective on S_1 for all $j \in [\nu]$.*
2. *We have $\pi_j(f_0) - \pi_j(f_i)$ is a non-zero polynomial with degree d for all $i \in [Q]$ and $j \in [\nu]$. Here π_j is extended to $\pi_j : \Phi[X] \to \Omega_j[X]$ in a natural way.*
3. *We have $S_0 \supseteq \cup_{i \in [0,Q]} \{-f_i(y_1, \ldots, y_\mu) | y_1, \ldots, y_\mu \in S_1\}$.*

Then, for $y_0 \overset{\$}{\leftarrow} S_0$ and $y_1, \ldots, y_\mu \overset{\$}{\leftarrow} S_1$, we have

$$\frac{1}{|S_0|}\left(1 - \frac{d\nu Q}{|S_1|}\right) \le \Pr_{y_0, \boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0\ \wedge\ y_0 + f_1(\boldsymbol{y}') \in \Phi^*\ \wedge \cdots \wedge\ y_0 + f_Q(\boldsymbol{y}') \in \Phi^*] \le \frac{1}{|S_0|}$$

where we denote $\boldsymbol{y}' = (y_1, \ldots, y_\mu)$ and $\Phi^* = \Pi^{-1}(\Omega_1^* \times \cdots \times \Omega_\nu^*)$.

Proof. Let us denote $\gamma := \Pr_{y_0, \boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0\ \wedge\ y_0 + f_1(\boldsymbol{y}') \in \Phi^*\ \wedge \cdots \wedge\ y_0 + f_Q(\boldsymbol{y}') \in \Phi^*]$ where the probability is taken over $y_0 \overset{\$}{\leftarrow} S_0$ and $y_1, \ldots, y_\mu \overset{\$}{\leftarrow} S_1$. We first show the upper bound. We have

$$\gamma \le \Pr_{y_0, \boldsymbol{y}'}[y_0 + f_0(\boldsymbol{y}') = 0] = \Pr_{y_0, \boldsymbol{y}'}[y_0 = -f_0(\boldsymbol{y}')] = \frac{1}{|S_0|}.$$

The last equation follows since there exists unique $y_0 \in S_0$ such that $y_0 = -f_0(\boldsymbol{y}')$, for any $\boldsymbol{y}' \in S_1^\mu$ from our third assumption. We then proceed to show the lower bound. We have

$$\begin{aligned}
\gamma &= \Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0 \ \wedge \ y_0 + f_1(\boldsymbol{y}') \in \Phi^* \ \wedge \cdots \wedge \ y_0 + f_Q(\boldsymbol{y}') \in \Phi^*] \\
&= \Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0\ ] \\
&\quad - \Pr_{y_0,\boldsymbol{y}'}[y_0 + f_0(\boldsymbol{y}') = 0 \wedge \neg \left(y_0 + f_1(\boldsymbol{y}') \in \Phi^* \wedge \cdots \wedge y_0 + f_Q(\boldsymbol{y}') \in \Phi^*\right)] \qquad (4.21) \\
&= \Pr_{y_0,\boldsymbol{y}'}[y_0 + f_0(\boldsymbol{y}') = 0] - \Pr_{y_0,\boldsymbol{y}'}[\bigvee_{i=1}^{Q} \left(y_0 + f_0(\boldsymbol{y}') = 0 \wedge y_0 + f_i(\boldsymbol{y}') \notin \Phi^*\right)] \qquad (4.22) \\
&\geq \Pr_{y_0,\boldsymbol{y}'}[y_0 + f_0(\boldsymbol{y}') = 0] - \sum_{i\in[Q]} \Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0 \ \wedge \ y_0 + f_i(\boldsymbol{y}') \notin \Phi^*\ ] \qquad (4.23) \\
&= \frac{1}{|S_0|} - \sum_{i\in[Q]} \underbrace{\Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0 \ \wedge \ y_0 + f_i(\boldsymbol{y}') \notin \Phi^*\ ]}_{:=\gamma_i'} \qquad (4.24)
\end{aligned}$$

where Eq.(4.21) is a general equation that holds for any event, Eq.(4.22) follows from De morgan's laws and the distributive property, Eq.(4.23) follows from the union bound, Eq.(4.24) is again from our third assumption. We then have to show an upper bound for γ_i'.

$$\begin{aligned}
\gamma_i' &= \Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0 \ \wedge \ y_0 + f_i(\boldsymbol{y}') \notin \Phi^*\ ] \\
&= \Pr_{y_0,\boldsymbol{y}'}[\ y_0 + f_0(\boldsymbol{y}') = 0 \ \wedge \ f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ ] \qquad (4.25) \\
&= \Pr_{y_0,\boldsymbol{y}'}[\ y_0 = -f_0(\boldsymbol{y}') \mid f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ ] \cdot \Pr_{y_0,\boldsymbol{y}'}[\ f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ ] \qquad (4.26) \\
&= \frac{1}{|S_0|} \cdot \Pr_{y_0,\boldsymbol{y}'}[\ f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ ] \qquad (4.27) \\
&= \frac{1}{|S_0|} \cdot \underbrace{\Pr_{\boldsymbol{y}'}[\ f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ ]}_{:=\gamma_i''} \qquad (4.28)
\end{aligned}$$

where Eq.(4.25) is just an equivalent expression, Eq.(4.26) is trivial, Eq.(4.27) is from the fact that for any $\boldsymbol{y}' \in S_1^\mu$ there exists unique $y_0 \in S_0$ such that $y_0 = -f_0(\boldsymbol{y}')$ (from our third assumption), and in Eq.(4.28) we omit y_0 since it is independent of the probability. It suffices to show an upper bound for γ_i''. We have

$$\begin{aligned}
\gamma_i'' &= \Pr_{\boldsymbol{y}' \xleftarrow{\$} S_1^\mu} \left[\ f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}') \notin \Phi^*\ \right] \qquad (4.29) \\
&= \Pr_{\boldsymbol{y}' \xleftarrow{\$} S_1^\mu} \left[\bigvee_{j=1}^{\nu} \Pi(f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}')) \in \Omega_1 \times \cdots \times \Omega_{j-1} \times \{0\} \times \Omega_{j+1} \times \cdots \times \Omega_\nu\right] \qquad (4.30) \\
&\leq \sum_{j=1}^{\nu} \Pr_{\boldsymbol{y}' \xleftarrow{\$} S_1^\mu} [\Pi(f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}')) \in \Omega_1 \times \cdots \times \Omega_{j-1} \times \{0\} \times \Omega_{j+1} \times \cdots \times \Omega_\nu] \qquad (4.31) \\
&= \sum_{j=1}^{\nu} \Pr_{\boldsymbol{y}' \xleftarrow{\$} S_1^\mu} [\pi_j(f_0(\boldsymbol{y}') - f_i(\boldsymbol{y}')) = 0] \qquad (4.32)
\end{aligned}$$

$$= \sum_{j=1}^{\nu} \Pr_{\boldsymbol{y}' \xleftarrow{\$} S_1^{\mu}} \left[\big(\pi_j(f_0 - f_i)\big)(\pi_j(\boldsymbol{y}')) = 0 \right] \tag{4.33}$$

$$= \sum_{j=1}^{\nu} \Pr_{\boldsymbol{z} \xleftarrow{\$} \pi_j(S_1)^{\mu}} \left[\big(\pi_j(f_0 - f_i)\big)(\boldsymbol{z}) = 0 \right] \tag{4.34}$$

$$\leq \sum_{j=1}^{\nu} \frac{d}{|\pi_j(S_1)|} \tag{4.35}$$

$$= \frac{d\nu}{|S_1|} \tag{4.36}$$

where in Eq.(4.29) we made the distribution of $\boldsymbol{y}'$ explicit, Eq.(4.30) is from the fact that $\Phi \backslash \Phi^* = \Pi^{-1} \left(\cup_{j=1}^{\nu} (\Omega_1 \times \cdots \times \Omega_{j-1} \times \{0\} \times \Omega_{j+1} \times \cdots \times \Omega_{\nu}) \right)$, Eq.(4.31) follows from the union bound, Eq.(4.32) is by the definition of Π, Eq.(4.33) follows since π_j is a homomorphism, Eq.(4.34) follows since π_j is injective on S_1 (our first assumption), Eq.(4.35) is from the fact that $\pi_j(f_0 - f_i) \in \Omega_j[X]$ is a non-zero polynomial with degree d (our second assumption) and the Schwartz-Zippel lemma, and Eq.(4.36) follows since π_j is injective on S_1.

□

4.4 Construction from RLWE

In this section, we show our IBE scheme from the RLWE assumption. Let d be a (flexible) constant number. In addition, let the identity space of the scheme be $\mathcal{ID} = \{0,1\}^{\kappa}$ for some $\kappa \in \mathbb{N}$ and the message space be $\{0,1\}^n \subset R$.[2] For our construction, we consider an efficiently computable injective map S that maps an identity $\mathsf{ID} \in \{0,1\}^{\kappa}$ to a subset $S(\mathsf{ID})$ of $[1,\ell]^d$, where $\ell = \lceil \kappa^{1/d} \rceil$. Such a map can be constructed easily as we explained in Section 4.3.2. Let $n := n(\lambda)$, $b := b(n)$, $\rho := \rho(n)$, $m := 2n$, $k := k(n)$, $q := q(n)$, $\ell := \ell(n)$, $\alpha := \alpha(n)$, $\alpha' := \alpha'(n)$, and $\sigma := \sigma(n)$ be parameters that are specified later. Let also $\Phi_m(X) = X^n + 1$ be the mth cyclotomic polynomial and $R = \mathbb{Z}[X]/(\Phi_m(X))$.

$\mathsf{Setup}(1^{\lambda})$: On input 1^{λ}, it first runs $(\boldsymbol{a}, \boldsymbol{T_a}) \xleftarrow{\$} \mathsf{TrapGen}(1^n, 1^k, q, \rho)$ to obtain $\boldsymbol{a} \in R_q^k$ and $\boldsymbol{T_a} \in R^{k \times k}$. It also picks $u \xleftarrow{\$} R_q$, $\boldsymbol{b}_0, \boldsymbol{b}_{i,j} \xleftarrow{\$} R_q^k$ for $(i,j) \in [d] \times [\ell]$ and outputs

$$\mathsf{mpk} = (\boldsymbol{a}, \boldsymbol{b}_0, \{\boldsymbol{b}_{i,j}\}_{(i,j)\in[d]\times[\ell]}, u) \quad \text{and} \quad \mathsf{msk} = \boldsymbol{T_a}.$$

In the following, we use a deterministic function $\mathsf{H} : \mathcal{ID} \to R_q^k$ defined as

$$\mathsf{H}(\mathsf{ID}) = \boldsymbol{b}_0 + \sum_{(j_1,\ldots,j_d)\in S(\mathsf{ID})} \mathsf{PubEval}_d(\boldsymbol{b}_{1,j_1}, \boldsymbol{b}_{2,j_2}, \ldots, \boldsymbol{b}_{d,j_d}) \in R_q^k.$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: It first computes $\mathsf{H}(\mathsf{ID})$ and picks $\boldsymbol{e} \in R^{2k}$ such that

$$[\boldsymbol{a} | \mathsf{H}(\mathsf{ID})] \cdot \boldsymbol{e}^T = u$$

using $\mathsf{SampleLeft}(\boldsymbol{a}, \mathsf{H}(\mathsf{ID}), u, \mathbf{T}_{\boldsymbol{a}}, \sigma) \to \boldsymbol{e}$. It returns $\mathsf{sk}_{\mathsf{ID}} = \boldsymbol{e}$.

[2]Note that we regard m as an elements in R via $\phi^{-1} : \mathbb{Z}^n \to R$ (the inversion of coefficient embedding).

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}^n \subset R$, it first picks $s \xleftarrow{\$} R_q$, $x_0 \xleftarrow{\$} D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$, $\boldsymbol{x}_1, \boldsymbol{x}_2 \xleftarrow{\$} \left(D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha'}\right)^k$. Then it computes

$$c_0 = su + x_0 + \lfloor q/2 \rceil \cdot \mathsf{M}, \quad \boldsymbol{c}_1 = s[\boldsymbol{a}|\mathsf{H}(\mathsf{ID})] + [\boldsymbol{x}_1|\boldsymbol{x}_2].$$

Finally, it outputs the ciphertext $C = (c_0, \boldsymbol{c}_1) \in R_q \times R_q^{2k}$.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C)$: To decrypt a ciphertext $C = (c_0, \boldsymbol{c}_1)$ using a private key $\mathsf{sk}_{\mathsf{ID}} = \boldsymbol{e}$, it computes

$$\left(\lfloor (2/q) \cdot \phi(c_0 - \boldsymbol{c}_1 \boldsymbol{e}^T) \rceil \mod 2\right) = m. \tag{4.37}$$

Here, the rounding function $\lfloor \cdot \rceil$ is applied componentwise.

4.4.1 Correctness and Parameter Selection.

The following lemma states the correctness of our above IBE scheme.

Lemma 4.6 (Correctness). *Assume $\alpha q \omega(\sqrt{\log n}) + \sqrt{nk}\alpha' \sigma \omega(\sqrt{\log nk}) \leq q/5$ holds with over whelming probability. Then the above scheme has negligible decryption error.*

Before proving Lemma 4.6, we prepare the following two lemmas.

Lemma 4.7 ([MR04], Lemma 4.4). *For any n-dimensional lattice Λ, real $\epsilon \in (0,1)$ and $s \geq \eta_\epsilon(\Lambda)$, we have $\Pr[\|\mathbf{x}\| > s\sqrt{n} \mid \mathbf{x} \leftarrow D_{\Lambda, s\omega(\sqrt{\log n})}] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}$.*

The following is an analogue of [ABB10], Lemma 12 where the error is instead chosen from the discrete Gaussian.

Lemma 4.8 (Discrete Gaussian Error Bound). *Let $\mathbf{e}$ be some vector in $\mathbb{Z}^n$ and let $\mathbf{x} \leftarrow D_{\mathbb{Z}^n,\alpha q}$ for some $\alpha q > \omega(\sqrt{\log n})$. Then the quantity $|\mathbf{e}\mathbf{x}^T|$ treated as an integer in $[0, \dots, q-1]$ satisfies $|\mathbf{e}\mathbf{x}^T| \leq \|\mathbf{e}\|_2 \alpha q \omega(\sqrt{\log n})$ with all but negligible probability in n.*

Proof. (of Lemma 4.8.) By [MP12], Lemma 2.8, each element of x_i are δ-subgaussian of parameter αq, where $\delta > 0$ is negligible in n. Then the random variable $\mathbf{e}\mathbf{x}^T$ is $n\delta$-subgaussian with parameter $\|\mathbf{e}\|_2 \alpha q$. Hence by the subgaussian distribution tail bound, we have $\Pr[|\mathbf{e}\mathbf{x}^T| > \|\mathbf{e}\|_2 \alpha q \omega(\sqrt{\log n})] \leq \mathsf{negl}(n)$, which proves the lemma. □

Then, the proof of Lemma 4.6 is given as follows.

Proof. When the $\mathsf{Decrypt}$ algorithm operates as specified for a valid encryption of message $\mathsf{M} \in \{0,1\}^n \subset R$, we have

$$\phi(c_0 - \boldsymbol{c}_1 \boldsymbol{e}^T) = \lfloor \frac{q}{2} \rceil \phi(\mathsf{M}) + \underbrace{\phi(x_0) - \phi([\boldsymbol{x}_1|\boldsymbol{x}_2])\mathrm{rot}(\boldsymbol{e}^T)}_{\text{error term}},$$

Hence, for the $\mathsf{Decrypt}$ algorithm to output M, we need to show that the error term does not exceed, say $q/5$. Since $x_0 \xleftarrow{\$} D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$, the vector $\phi(x_0)$ is a subgaussian with parameter αq, i.e., $D_{\mathbb{Z}^n,\alpha q}$. Therefore, by the standard subgaussian tail bound argument, $|\phi(x_0)_j| \leq \alpha q \omega(\sqrt{\log n})$ with all but negligible probability, where $\phi(x_0)_j$ denotes the jth entry. Furthermore, since $\boldsymbol{x}_1, \boldsymbol{x}_2 \xleftarrow{\$} (D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha'})^k$, we have that $\phi([\boldsymbol{x}_1|\boldsymbol{x}_2]) \xleftarrow{\$} D_{\mathbb{Z}^{2nk},\alpha'}$. From the definition of the map rot, we have that each column of $\mathrm{rot}(\boldsymbol{e}^T) \in \mathbb{Z}^{2nk\times n}$ is of norm $\|\phi(\boldsymbol{e})\|_2$. Hence, by Lemma 4.7, Lemma 4.8 and from the fact that $\phi(\boldsymbol{e}) \xleftarrow{\$} D_{\Lambda^{\perp}_{\phi(u)}([\mathrm{rot}(a^T)^T|\mathrm{rot}(\mathsf{H}(\mathsf{ID})^T)^T]),\sigma}$, we have $|\phi([\boldsymbol{x}_1|\boldsymbol{x}_2])\mathrm{rot}(\boldsymbol{e}^T)_j| \leq \|\phi(\boldsymbol{e})\|_2 \cdot \alpha'\omega(\sqrt{\log nk}) \leq \sqrt{nk}\alpha'\sigma\omega(\sqrt{\log nk})$ with all but negligible probability, where $\mathrm{rot}(\boldsymbol{e}^T)_j$ denotes the jth column.

Putting all the pieces together, we conclude that the jth entry of the error term is bounded as

$$\left|\left(\phi(x_0) - \phi([\boldsymbol{x}_1|\boldsymbol{x}_2])\mathrm{rot}(\boldsymbol{e}^T)\right)_j\right| \leq \alpha q\omega(\sqrt{\log n}) + \sqrt{nk}\alpha'\sigma\omega(\sqrt{\log nk}),$$

with all but negligible probability. By assumption this is smaller than $q/5$ with overwhelming probability. Hence, the error probability for the $\mathsf{Decrypt}$ algorithm is negligible. □

Parameter selection. To satisfy the correctness requirement and make the security proof follow through, we need the following:

- the error term is less than $q/5$ with overwhelming probability (i.e., $\alpha q\omega(\sqrt{\log n})+\sqrt{nk}\alpha'\sigma\omega(\sqrt{\log nk})$. See Lemma 4.6,),
- $\mathsf{TrapGen}$ can operate (i.e., $\rho < \frac{1}{2}\sqrt{q/n}$ and $k \geq 2\log_\rho q$. See Lemma 2.16.),
- the gadget matrix $\boldsymbol{g}_b$ can be defined (i.e., $k \geq \lceil \log_b q \rceil$. See Lemma 2.16.),
- the regularity lemma (Lemma 2.15) can be applied in the security proof (i.e., $\frac{k}{2}\left(\frac{q^2}{(2\rho+1)^k}\right)^{\frac{n}{2}} = \mathsf{negl}(n)$.),
- σ is sufficiently large so that $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ work (i.e., $\sigma > O(b\rho \cdot \sqrt{n\log_\rho q}) \cdot \omega(\sqrt{\log nk})$ and $\sigma > s_1(\boldsymbol{R})\sqrt{b^2+1}\cdot\omega(\sqrt{\log n})$, where $s_1(\boldsymbol{R}) \leq C''\cdot\kappa\rho\sqrt{n}(\sqrt{k}+\omega(\sqrt{\log n}))\Big((cn)^{d-1}+ bnk\frac{(cn)^{d-1}-1}{cn-1}\Big)$ for some absolute constant C''. See Eq.(4.45). The latter condition turns out to be more restrictive.),
- ReRand algorithm in the security proof works (i.e., $\alpha' > 2\alpha q(s_1(\boldsymbol{R})+1)$, $\alpha q > \omega(\sqrt{\log nk})$ where $s_1(\boldsymbol{R})$ is the same as the one defined above. See Lemma 2.6.),
- the worst case to average case reduction works (i.e., $\alpha q \geq n^{3/2}k^{1/4}\omega(\log^{9/4} n)$. See Section 2.2.3.).

Recall that d is a (flexible) constant which may be set very small (e.g., $d = 2$ or 3) in a typical setting, and $\kappa(n) = n$ is the size of the identity space ID. To satisfy the above requirements, we propose two candidate parameter selections as follows:

Type 1 IBE. For this construction we set $b = 2$ and $\rho = 1$ in order to reduce the modulus size q. Recalling that we defined $\log_1 q := \log_2 q$, we can set the parameters as follows:

$$\begin{array}{llll} k = 4(d+1)\log n, & q = n^{2d+2}, & b = 2, & \rho = 1, \\ \sigma = n^{d-\frac{1}{2}}\cdot\omega(\log n), & \alpha = n^{-2d+\frac{1}{2}}\cdot\omega(\log^{\frac{9}{2}} n)^{-1}, & \alpha' = n^{d+2\eta+2}\cdot\omega(\log^3 n)^{-1}. & \end{array}$$

We denote this specific instantiation as the Type 1 IBE scheme.

Type 2 IBE. For this construction we set $b = \rho = n^\eta$ for an arbitrary positive real η in order to reduce the size of the public parameters, private keys, and ciphertexts. Namely, one way to set the parameters is as follows:

$$\begin{aligned} &k = 4 + \frac{2d+2}{\eta}, && q = n^{2d+2+4\eta}, && b = \rho = n^\eta, \\ &\sigma = n^{d+2\eta-\frac{1}{2}} \cdot \omega(\log n), && \alpha = n^{-2d-\frac{7}{2}\eta+\frac{1}{2}} \cdot \omega(\log^2 n)^{-1}, && \alpha' = n^{d+2\eta+2} \cdot \omega(\log^{\frac{3}{4}} n)^{-1}. \end{aligned}$$

By plugging in $\eta = \frac{1}{4}$ we obtain the following concrete parameter selection:

$$\begin{aligned} &k = 8d + 12, && q = n^{2d+3}, && b = \rho = n^{\frac{1}{4}}, \\ &\sigma = n^d \cdot \omega(\log n), && \alpha = n^{-2d-\frac{3}{8}} \cdot \omega(\log^2 n)^{-1}, && \alpha' = n^{d+\frac{5}{2}} \cdot \omega(\log^{\frac{3}{4}} n)^{-1}. \end{aligned}$$

We denote this specific instantiation as the Type 2 IBE scheme.

4.4.2 Security Proof for the Scheme

The following theorem addresses the security of the scheme. The proof proceeds in a similar manner as in [Yam16], but we incorporate several novel ideas as we explained in Section 4.2.

Theorem 4.1. *The above IBE scheme is adaptively-anonymous secure assuming* $\mathsf{RLWE}_{n,k+1,q,D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}}$ *is hard, where the ciphertext space is* $\mathcal{C} = R_q \times R_q^{2k}$.

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the adaptively-anonymous security of the scheme. In addition, let $\epsilon = \epsilon(n)$ and $Q = Q(n)$ be its advantage and the upper bound of the number of key extraction queries, respectively.

Since $\mathcal{A}$ is PPT and λ and n are polynomially related (namely, $n = O(\lambda^\delta)$ for some constant δ), there exists a constant number $c_1 \in \mathbb{N}$ such that $4(dQ+1) \leq n^{c_1}$ for all n that are sufficiently large. Similarly, since $\mathcal{A}$ breaks the security of the scheme, there exists $c_2 \in \mathbb{N}$ such that $2\epsilon \geq n^{-c_2}$ holds for infinitely many n. By setting $c = c_1 + c_2$, we have that

$$4dQ \leq n^c \text{ for all } n \in \mathbb{N} \quad \text{and} \quad \frac{\epsilon}{2(dQ+1)} \geq \frac{1}{n^c} \quad \text{for infinitely many } n \in \mathbb{N}. \tag{4.38}$$

In the proof, we will assume $d(c-1) < n$. Since both c and d are constant numbers, this holds for sufficiently large n.

We show the security of the scheme via the following games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. While it is set $\mathsf{coin}' = \widehat{\mathsf{coin}}$ in the first game, these values might be different in the later games. In the following, we define X_i to be the event that $\mathsf{coin}' = \mathsf{coin}$.

Game_0 : This is the real security game. Recall that since the ciphertext space is $\mathcal{C} = R_q \times R_q^{2k}$, in the challenge phase, the challenge ciphertext is set as $C^\star = (c_0, \boldsymbol{c}_1) \xleftarrow{\$} R_q \times R_q^{2k}$ if $\mathsf{coin} = 1$. At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By definition, we have

$$\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : For integers $t_0, t_1 \in \mathbb{Z}$ such that $t_0 \le t_1$ and positive integer $c \in \mathbb{N}$, let us denote $[t_0, t_1]_{R,c}$ as

$$[t_0, t_1]_{R,c} := \left\{ \sum_{i=0}^{c-1} a_i X^i \;\middle|\; a_i \in [t_0, t_1] \text{ for all } i \in [0, c-1] \right\} \subseteq R. \tag{4.39}$$

In words, $[t_0, t_1]_{R,c}$ denotes the set of polynomials of degree less then $c - 1$ with all of its coefficients in the interval $[t_0, t_1]$. Note that c is the constant defined in Eq.(4.38). In this game, we change Game_0 so that the challenger performs the following additional step at the end of the game. First, the challenger picks $\boldsymbol{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[d,\ell]})$ as

$$y_0 \xleftarrow{\$} [-\kappa(cn)^d, -1]_{R,(c-1)d+1} \qquad \text{and} \qquad y_{i,j} \xleftarrow{\$} [1, n]_{R,c} \tag{4.40}$$

for $(i, j) \in [d] \times [\ell]$. Recall κ is the length of the identities. We then define a function $\mathsf{F}_{\boldsymbol{y}} : \mathcal{ID} \to R_q$ as follows:

$$\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}) = y_0 + \sum_{(j_1,\ldots,j_d)\in S(\mathsf{ID})} y_{1,j_1} \cdots y_{d,j_d}.$$

Then the challenger checks whether the following condition holds:

$$\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star) = 0 \;\wedge\; \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_1) \in R_q^* \;\wedge\; \cdots \;\wedge\; \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_Q) \in R_q^*, \tag{4.41}$$

where $\mathsf{ID}^\star$ is the challenge identity, and $\mathsf{ID}_1, \ldots, \mathsf{ID}_Q$ are identities for which $\mathcal{A}$ has made key extraction queries. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$, and sets $\mathsf{coin}' \xleftarrow{\$} \{0, 1\}$. In this case, we say that the challenger aborts. If condition (4.41) holds, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. As we will show in Lemma 4.9, we have

$$\left| \Pr[X_1] - \frac{1}{2} \right| \ge \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(\frac{\epsilon}{2} - \frac{dQ}{n^c} \right).$$

So as not to interrupt the proof of Theorem 4.1, we intentionally skip the proof for the time being.

Game_2 : In this game, we change the way $\boldsymbol{b}_0$ and $\boldsymbol{b}_{i,j}$ are chosen. At the beginning of the game, the challenger picks $\boldsymbol{R}_0, \boldsymbol{R}_{i,j} \xleftarrow{\$} [-\rho, \rho]_R^{k\times k}$ for $(i, j) \in [d] \times [\ell]$. It also picks $\boldsymbol{y}$ as in Game_1. Then, $\boldsymbol{a}$, $\boldsymbol{b}_0$, and $\boldsymbol{b}_{i,j}$ are defined as

$$\boldsymbol{b}_0 = \boldsymbol{a}\boldsymbol{R}_0 + y_0\boldsymbol{g}_b, \qquad \boldsymbol{b}_{i,j} = \boldsymbol{a}\boldsymbol{R}_{i,j} + y_{i,j}\boldsymbol{g}_b, \tag{4.42}$$

for $(i, j) \in [d] \times [\ell]$. The rest of the game is the same as in Game_1.

Now, we bound $|\Pr[X_2] - \Pr[X_1]|$. By Lemma 2.15, the distributions

$$\left(\, \boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0 + y_0\boldsymbol{g}_b, \{\boldsymbol{a}\boldsymbol{R}_{i,j} + y_{i,j}\boldsymbol{g}_b\}_{(i,j)\in[d]\times[\ell]} \,\right) \text{ and } \left(\, \boldsymbol{a}, \boldsymbol{b}_0, \{\boldsymbol{b}_{i,j}\}_{(i,j)\in[d]\times[\ell]} \,\right)$$

are $\mathsf{negl}(n)$-close, where $\boldsymbol{b}_0, \boldsymbol{b}_{i,j} \xleftarrow{\$} R_q^k$. Therefore, we have $|\Pr[X_1] - \Pr[X_2]| = \mathsf{negl}(n)$.

Game_3 Recall that in the previous game, the challenger aborts at the end of the game if condition (4.41) is not satisfied. In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have $\Pr[X_2] = \Pr[X_3]$.

Before describing the next game, we define $\boldsymbol{R}_{\mathsf{ID}} \in R^{k\times k}$ for an identity $\mathsf{ID} \in \mathcal{ID}$ as

$$\boldsymbol{R}_{\mathsf{ID}} = \boldsymbol{R}_0 + \sum_{(j_1,\ldots,j_d)\in S(\mathsf{ID})} \mathsf{TrapEval}_d(\boldsymbol{R}_{1,j_1},\ldots,\boldsymbol{R}_{d,j_d},y_{1,j_1},\ldots,y_{d,j_d}). \tag{4.43}$$

Note that by the definition of $\boldsymbol{R}_{\mathsf{ID}}$, $\mathsf{H}(\mathsf{ID})$, $\mathsf{PubEval}$ and $\mathsf{TrapEval}$ (Lemma 4.1) we have

$$\begin{aligned}\mathsf{H}(\mathsf{ID}) &= \boldsymbol{b}_0 + \sum_{(j_1,\ldots,j_d)\in S(\mathsf{ID})} \mathsf{PubEval}_d(\boldsymbol{b}_{1,j_1}\boldsymbol{b}_{2,j_2},\ldots,\boldsymbol{b}_{d,j_d})\\ &= \boldsymbol{a}\boldsymbol{R}_{\mathsf{ID}} + \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID})\boldsymbol{g}_b. \end{aligned}\tag{4.44}$$

Since $\boldsymbol{R}_0, \boldsymbol{R}_{i,j} \xleftarrow{\$} [-\rho,\rho]_R^{k\times k}$, from Lemma 2.13 we have $s_1(\boldsymbol{R}_0), s_1(\boldsymbol{R}_{i,j}) \le B$ with all but negligible probability where $B = C'\cdot\rho\sqrt{n}(\sqrt{k}+\omega(\sqrt{\log n}))$ for some positive absolute constant C'. Furthermore, we have $\|y_{i,j}\|_1 \le cn$ from Eq. (4.40). Therefore by Lemma 4.1, we have

$$\begin{aligned} s_1(\boldsymbol{R}_{\mathsf{ID}}) &\le s_1(\boldsymbol{R}_0) + \sum_{(j_1,\ldots,j_d)\in S(\mathsf{ID})} s_1(\mathsf{TrapEval}_d(\boldsymbol{R}_{1,j_1},\ldots,\boldsymbol{R}_{d,j_d},y_{1,j_1},\ldots,y_{d,j_d}))\\ &\le B\left(1+\kappa(cn)^{d-1}+\kappa bnk\frac{(cn)^{d-1}-1}{cn-1}\right), \end{aligned}\tag{4.45}$$

for any $\mathsf{ID} \in \mathcal{ID}$ with all but negligible probability.

Game_4 In this game, we change the way the vector $\boldsymbol{a}$ is sampled. Namely, Game_4 challenger picks $\boldsymbol{a} \xleftarrow{\$} R_q^k$ instead of generating it with a trapdoor. By Lemma 2.16, this makes only negligible difference. Furthermore, we also change the way the key extraction queries are answered. When $\mathcal{A}$ makes a key extraction query for an identity ID, the challenger first computes $\boldsymbol{R}_{\mathsf{ID}}$ as in Eq.(4.43). It aborts if $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}) \notin R_q^*$ as in the previous game and runs

$$\mathsf{SampleRight}(\boldsymbol{a},\boldsymbol{g}_b,\boldsymbol{R}_{\mathsf{ID}},\mathsf{F}_{\mathbf{y}}(\mathsf{ID}),u,\mathbf{T}_{\boldsymbol{g}_b},\sigma) \to \boldsymbol{e},$$

otherwise. Note that in the previous game the private key was sampled as

$$\mathsf{SampleLeft}(\boldsymbol{a},\mathsf{H}(\mathsf{ID}),u,\mathbf{T}_{\boldsymbol{a}},\sigma) \to \boldsymbol{e}.$$

By Eq.(4.45) and for our choice of σ, the output distribution of $\mathsf{SampleRight}$ is $\mathsf{negl}(n)$-close to $D^{\mathsf{coeff}}_{\Lambda^{\perp}_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T|\mathrm{rot}(\mathsf{H}(\mathsf{ID})^T)^T]),\sigma}$. Furthermore, by the choice of σ, this distribution is $\mathsf{negl}(n)$-close to the output distribution of $\mathsf{SampleLeft}$. Therefore, the above change alters the view of $\mathcal{A}$ only negligibly. Thus, we have $|\Pr[X_3]-\Pr[X_4]| = \mathsf{negl}(n)$.

Game_5 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. Recall in the previous games when $\mathsf{coin} = 0$, we created a valid challenge ciphertext as in the real scheme. If $\mathsf{coin} = 0$ and $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star) = 0$ (i.e., if it does not abort), to create the challenge ciphertext Game_5 challenger first picks $s \xleftarrow{\$} R_q$ and $\boldsymbol{x} \xleftarrow{\$} (D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q})^k$ and computes $\boldsymbol{v} = s\boldsymbol{a}+\boldsymbol{x} \in R^k$. It then runs the algorithm

$$\mathsf{ReRand}\left(\mathrm{rot}([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]),\phi(\boldsymbol{v}),\alpha q,\frac{\alpha'}{2\alpha q}\right) \to \mathbf{c} \in \mathbb{Z}_q^{2nk}$$

from Lemma 2.6, where $\boldsymbol{I}_k \in R^{k\times k}$ is the identity matrix of size $k\times k$. Finally, it picks $x_0 \xleftarrow{\$} D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$ and sets the challenge ciphertext as

$$C^\star = \left(\ c_0 = v_0 + \lfloor q/2 \rceil \cdot \mathsf{M},\ \boldsymbol{c}_1 = \phi^{-1}(\mathbf{c})\ \right) \in R_q \times R_q^{2k}, \tag{4.46}$$

where $v_0 = su + x_0$ and M is the message chosen by $\mathcal{A}$. We claim that this change alters the view of $\mathcal{A}$ only negligibly. To show this, observe that the input to ReRand is $\mathrm{rot}\big([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]\big) \in \mathbb{Z}_q^{nk\times 2nk}$ and

$$\phi(\boldsymbol{v}) = \phi(s\boldsymbol{a} + \boldsymbol{x}) = \phi(s)\mathrm{rot}(\boldsymbol{a}) + \phi(\boldsymbol{x}) \in \mathbb{Z}_q^{nk},$$

where $\phi(\boldsymbol{x})$ is distributed as $\phi(\boldsymbol{x}) \xleftarrow{\$} D_{\mathbb{Z}^{nk},\alpha q}$. Therefore, by the property of ReRand and our choice of α and α', the output $\mathbf{c} \in \mathbb{Z}_q^{2nk}$ is

$$\begin{aligned}\mathbf{c} &= \Big(\phi(s)\mathrm{rot}(\boldsymbol{a})\Big)\cdot \mathrm{rot}\big([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]\big) + \mathbf{x}' \\ &= \phi(s)\cdot \mathrm{rot}\big([\boldsymbol{a}|\mathsf{H}(\mathsf{ID}^\star)]\big) + \mathbf{x}' \\ &= \phi\big(s\big[\boldsymbol{a}|\mathsf{H}(\mathsf{ID}^\star)\big]\big) + \mathbf{x}',\end{aligned}$$

where the distribution of $\mathbf{x}'$ is within negligible distance from $\mathbf{x}' \xleftarrow{\$} D_{\mathbb{Z}^{2nk},\alpha'}$ due to Lemma 2.6. Here, we use the fact that $\mathsf{H}(\mathsf{ID}^\star) = \boldsymbol{a}\boldsymbol{R}_{\mathsf{ID}^\star}$ holds since $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star) = 0$. It can be readily seen that the distribution of $\boldsymbol{c}_1 = \phi^{-1}(\mathbf{c})$ in Game_5 is statistically close to that in Game_4. Therefore, we have $|\Pr[X_4] - \Pr[X_5]| = \mathsf{negl}(n)$.

Game_6 In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$ and the abort condition is not satisfied, to create the challenge ciphertext for identity $\mathsf{ID}^\star$ and message M, Game_6 challenger first picks $v_0 \xleftarrow{\$} R_q$, $\boldsymbol{v}' \xleftarrow{\$} R_q^k$ and $\boldsymbol{x} \xleftarrow{\$} (D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q})^k$, and runs

$$\mathsf{ReRand}\left(\mathrm{rot}\big([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]\big), \phi(\boldsymbol{v}), \alpha q, \frac{\alpha'}{2\alpha q}\right) \to \mathbf{c} \in \mathbb{Z}_q^{2nk}, \tag{4.47}$$

where $\boldsymbol{v} = \boldsymbol{v}' + \boldsymbol{x}$. Then, the challenge ciphertext is set as in Eq.(4.46). As we will show in Lemma 4.10, assuming $\mathsf{RLWE}_{n,k+1,q,D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}}$ is hard, we have $|\Pr[X_5] - \Pr[X_6]| = \mathsf{negl}(n)$.

Game_7 In this game, we further change the way the challenge ciphertext is created. When $\mathsf{coin} = 0$ and the abort condition is not satisfied, the challenge ciphertext for $\mathsf{ID}^\star$ is created as

$$C^\star = \big(\ c_0 = v_0 + \lfloor q/2 \rceil \cdot \mathsf{M},\ \boldsymbol{c}_1 = [\boldsymbol{v}'|\boldsymbol{v}'\boldsymbol{R}_{\mathsf{ID}^\star}] + [\boldsymbol{x}_1|\boldsymbol{x}_2]\ \big) \in R_q \times R^{2k},$$

where $v_0 \xleftarrow{\$} R_q$, $\boldsymbol{v}' \xleftarrow{\$} R_q^k$ and $\boldsymbol{x}_1, \boldsymbol{x}_2 \xleftarrow{\$} (D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha'})^k$.

We claim that this change alters the view of $\mathcal{A}$ only negligibly. This can be seen by a similar argument to that we made in the step from Game_3 to Game_4. We first observe that in Game_6 the input to ReRand is $\mathrm{rot}\big([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]\big) \in \mathbb{Z}_q^{nk\times 2nk}$ and

$$\phi(\boldsymbol{v}) = \phi(\boldsymbol{v}' + \boldsymbol{x}) = \phi(\boldsymbol{v}') + \phi(\boldsymbol{x}) \in \mathbb{Z}_q^{nk}, \tag{4.48}$$

where $\phi(\boldsymbol{x})$ is distributed as $D_{\mathbb{Z}^{nk},\alpha q}$. Therefore, the output $\mathbf{c} \in \mathbb{Z}_q^{2nk}$ of ReRand is

$$\mathbf{c} = \phi(\boldsymbol{v}')\cdot \mathrm{rot}\big([\boldsymbol{I}_k|\boldsymbol{R}_{\mathsf{ID}^\star}]\big) + \mathbf{x}' = \phi\big([\boldsymbol{v}'|\boldsymbol{v}'\boldsymbol{R}_{\mathsf{ID}^\star}]\big) + \mathbf{x}',$$

where the distribution of $\mathbf{x}'$ is within negligible distance from $\mathbf{x}' \xleftarrow{\$} D_{\mathbb{Z}^{2nk},\alpha'}$ due to Lemma 2.6. Hence, the distribution of $\boldsymbol{c}_1 = \phi^{-1}(\mathbf{c})$ in Game_6 is statistically close to that in Game_7. Therefore, we have $|\Pr[X_6] - \Pr[X_7]| = \mathsf{negl}(n)$.

Game_8 In this game, we change the way the key extraction queries are answered. Instead of running $\mathsf{SampleLeft}$ or $\mathsf{SampleRight}$, the (possibly inefficient) challenger directly picks a secret key $\mathsf{sk}_{\mathsf{ID}}$ for identity ID as $\mathsf{sk}_{\mathsf{ID}} \xleftarrow{\$} D^{\mathsf{coeff}}_{\Lambda^{\perp}_{\phi(u)}([\mathrm{rot}(\boldsymbol{a}^T)^T | \mathrm{rot}(\mathsf{H}(\mathsf{ID})^T)^T]),\sigma}$ without using $\boldsymbol{R}_{\mathsf{ID}}$. Similarly to the change from Game_3 to Game_4, by the choice of σ and Eq.(4.45), this alters the view of $\mathcal{A}$ only negligibly. Therefore, we have $|\Pr[X_7] - \Pr[X_8]| = \mathsf{negl}(n)$. Note that this is only a conceptual game in order to get rid of any (negligible) correlation between the secret key and $\boldsymbol{R}_{\mathsf{ID}}$ so as not to interfere with the statistical argument using $\boldsymbol{R}_{\mathsf{ID}^\star}$ in the following game.

Game_9 In this game, we change the challenge ciphertext to be a random vector, regardless of whether $\mathsf{coin} = 0$ or $\mathsf{coin} = 1$. Namely, Game_9 challenger generates the challenge ciphertext $C^\star = (c_0, \boldsymbol{c}_1)$ as

$$c_0 \xleftarrow{\$} R_q, \qquad \text{and} \qquad \boldsymbol{c}_1 \xleftarrow{\$} R_q^{2k}.$$

We now proceed to bound $|\Pr[X_8] - \Pr[X_9]|$. Since Game_8 and Game_9 differ only in the creation of the challenge ciphertext when $\mathsf{coin} = 0$, we focus on this case. First, it is easy to see that c_0 is uniformly random over R_q in both of Game_8 and Game_9. Therefore, we only need to show that the distribution of $\boldsymbol{c}_1$ in Game_8 is $\mathsf{negl}(n)$-close to the uniform distribution over R_q^{2k}. To see this, it suffices to show that $[\boldsymbol{v}' | \boldsymbol{v}' \boldsymbol{R}_{\mathsf{ID}^\star}]$ is distributed statistically close to the uniform distribution over R_q^{2k}. First, observe that the following distributions are $\mathsf{negl}(n)$-close:

$$(\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}'\boldsymbol{R}_0) \approx (\boldsymbol{a}, \boldsymbol{a}', \boldsymbol{v}', \boldsymbol{v}'') \approx (\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}''), \tag{4.49}$$

where $\boldsymbol{a}, \boldsymbol{a}' \xleftarrow{\$} R_q^k$, $\boldsymbol{R}_0 \xleftarrow{\$} [-\rho, \rho]_R^{k \times k}$, $\boldsymbol{v}', \boldsymbol{v}'' \xleftarrow{\$} R_q^k$. It can be seen that the first and the second distributions are $\mathsf{negl}(n)$-close, by applying Lemma 2.15 for $[\boldsymbol{a}; \boldsymbol{v}'] \in R_q^{2 \times k}$ and $\boldsymbol{R}_0$. It can also be seen that the second and the third distributions are $\mathsf{negl}(n)$-close, by applying the same lemma for $\boldsymbol{a}$ and $\boldsymbol{R}_0$. From the above, the following distributions are statistically close:

$$\begin{aligned}
& (\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}'\boldsymbol{R}_{\mathsf{ID}^\star}) \\
= & \left(\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}' \left(\boldsymbol{R}_0 + \sum_{(j_1, \ldots, j_d) \in S(\mathsf{ID})} \mathsf{TrapEval}_d(\boldsymbol{R}_{1,j_1}, \ldots, \boldsymbol{R}_{d,j_d}, y_{1,j_1}, \ldots, y_{d,j_d}) \right) \right) \\
\approx & \left(\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}'' + \boldsymbol{v}' \left(\sum_{(j_1, \ldots, j_d) \in S(\mathsf{ID})} \mathsf{TrapEval}_d(\mathbf{R}_{1,j_1}, \ldots, \mathbf{R}_{d,j_d}, y_{1,j_1}, \ldots, y_{d,j_d}) \right) \right) \\
\approx & (\boldsymbol{a}, \boldsymbol{a}\boldsymbol{R}_0, \boldsymbol{v}', \boldsymbol{v}'')
\end{aligned}$$

where $\boldsymbol{a}, \boldsymbol{a}' \xleftarrow{\$} R_q^k$, $\boldsymbol{R}_0 \xleftarrow{\$} [-\rho, \rho]_R^{k \times k}$, $\boldsymbol{v}', \boldsymbol{v}'' \xleftarrow{\$} R_q^k$. The second and the third distributions above are $\mathsf{negl}(n)$-close by Eq.(4.49). Note that we intentionally ignored all the $\boldsymbol{a}\boldsymbol{R}_{i,j}$ terms to keep the argument simple, since focusing on the $\boldsymbol{a}\boldsymbol{R}_0$ term is enough to prove randomness of $[\boldsymbol{v}' | \boldsymbol{v}' \boldsymbol{R}_{\mathsf{ID}^\star}]$. Therefore, we conclude that $|\Pr[X_8] - \Pr[X_9]| = \mathsf{negl}(n)$.

Analysis. From the above, we have

$$\left| \Pr[X_9] - \frac{1}{2} \right| = \left| \Pr[X_1] - \frac{1}{2} + \sum_{i=1}^{8} (\Pr[X_{i+1}] - \Pr[X_i]) \right|$$

$$\begin{aligned}
&\geq \left|\Pr[X_1] - \frac{1}{2}\right| - \sum_{i=1}^{8} |\Pr[X_{i+1}] - \Pr[X_i]| \\
&\geq \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(\frac{\epsilon}{2} - \frac{dQ}{n^c}\right) - \mathsf{negl}(n) \\
&= \frac{1}{\mathsf{poly}(n)} \left(\frac{\epsilon}{2} - \frac{dQ}{n^c}\right) - \mathsf{negl}(n) \qquad (4.50)
\end{aligned}$$

where the last equality follows from the facts that c and d are constants and $\kappa = \mathsf{poly}(n)$. Since the challenge ciphertext is independent from the value of coin in Game_9, we have $\Pr[X_9] = 1/2$ and thus $|\Pr[X_9] - 1/2| = 0$. Therefore, we have that $\epsilon/2 - dQ/n^c$ is negligible. However, by Eq.(4.38),

$$\frac{\epsilon}{2} - \frac{dQ}{n^c} \geq \frac{dQ+1}{n^c} - \frac{dQ}{n^c} = \frac{1}{n^c}$$

holds for infinitely many n, which is a contradiction. □

To complete the proof of Theorem 4.1, it remains to prove Lemma 4.9 and 4.10.

Lemma 4.9. *For any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr[X_1] - \frac{1}{2}\right| \geq \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(\frac{\epsilon}{2} - \frac{dQ}{n^c}\right).$$

Proof. For a sequence of identities $\mathbb{ID} = (\mathsf{ID}^\star, \mathsf{ID}_1, \ldots, \mathsf{ID}_Q) \in \mathcal{ID}^{Q+1}$, we define $\gamma(\mathbb{ID})$ as

$$\gamma(\mathbb{ID}) = \Pr_{\boldsymbol{y}}[\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star) = 0 \wedge \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_1) \neq 0 \wedge \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_2) \neq 0 \wedge \cdots \wedge \mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_Q) \neq 0]$$

where the probability is taken over $\boldsymbol{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[d,\ell]})$, which is chosen as specified in Game_1. Then, it suffices to show

$$\frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(1 - \frac{2dQ}{n^c}\right) \leq \gamma(\mathbb{ID}) \leq \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \qquad (4.51)$$

since by Lemma 4.4, this implies

$$\begin{aligned}
\left|\Pr[X_1] - \frac{1}{2}\right| &\geq \frac{\epsilon}{(\kappa c^d n^d)^{(c-1)d+1}} \left(1 - \frac{2dQ}{n^c}\right) - \frac{1}{2(\kappa c^d n^d)^{(c-1)d+1}} \left(1 - \left(1 - \frac{2dQ}{n^c}\right)\right) \\
&= \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(\epsilon\left(1 - \frac{2dQ}{n^c}\right) - \frac{dQ}{n^c}\right) \\
&\geq \frac{1}{(\kappa c^d n^d)^{(c-1)d+1}} \left(\frac{\epsilon}{2} - \frac{dQ}{n^c}\right)
\end{aligned}$$

where the last inequality follows from Eq.(4.38). In the following, we will prove Eq.(4.51) by applying Lemma 4.5. We set

$$\begin{array}{lll}
\nu = 2,\ \mu = d\ell & \Phi = R_q, & \\
\Omega_j = R_q/\langle t_j\rangle, & \pi_j : R_q \to R_q/\langle t_j\rangle, & \text{for } j \in [2], \\
S_0 = [-\kappa(cn)^d, -1]_{R,(c-1)d+1}, & S_1 = [1, n]_{R,c} &
\end{array}$$

where π_j is a natural homomorphism and t_1, t_2 are elements in R_q as defined in Lemma 2.14. Therefore, the map $\Pi : \Phi \ni y \mapsto (\pi_1(y), \pi_2(y)) \in \Omega_1 \times \Omega_2$ is an isomorphism. We define $f_i(\{Y_{j,j'}\}_{(j,j')\in[d]\times[\ell]})$ for $i \in [0, Q]$ as

$$f_i\left(\{Y_{j,j'}\}_{(j,j')\in[d]\times[\ell]}\right) = \sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}_i)} Y_{1,j'_1} Y_{2,j'_2} \cdots Y_{d,j'_d}$$

where we define $\mathsf{ID}_0 := \mathsf{ID}^\star$. Note that we have $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}_i) = y_0 + f_i(\{y_{i,j}\}_{(i,j)\in[d]\times[\ell]})$. We now check that the three conditions for Lemma 4.5 hold.

- We prove that π_j is injective on S_1 for $j \in \{1, 2\}$. Assume for contradiction that there are $a_1, a_2 \in S_1$ with $a_1 \neq a_2$ and $\pi_j(a_1) = \pi_j(a_2) \Leftrightarrow \pi_j(a_1 - a_2) = 0$. We then have $a_1 - a_2 \notin R_q^*$. On the other hand, we have $\|\phi(a_1 - a_2)\|_2 \leq \sqrt{c}n < \sqrt{q}$. However, this contradicts Lemma 2.14.

- For $i \in [1, Q]$, we have

$$f_0\left(\{Y_{j,j'}\}\right) - f_i\left(\{Y_{j,j'}\}\right) = \sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}^\star)} Y_{1,j'_1} Y_{2,j'_2} \cdots Y_{d,j'_d} - \sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}_i)} Y_{1,j'_1} Y_{2,j'_2} \cdots Y_{d,j'_d}.$$

  Since $\mathsf{ID}^\star \neq \mathsf{ID}_i$ and S is an injective map, we have $S(\mathsf{ID}^\star) \neq S(\mathsf{ID}_i)$. Therefore, there exists $(j_1^\star, \dots, j_d^\star) \in [\ell]^d$ such that $(j_1^\star, \dots, j_d^\star) \in S(\mathsf{ID}^\star) \triangle S(\mathsf{ID}_i)$, where $S(\mathsf{ID}^\star) \triangle S(\mathsf{ID}_i)$ denotes the symmetric difference of $S(\mathsf{ID}^\star)$ and $S(\mathsf{ID}_i)$. Thus, the above polynomial is a non-zero polynomial with degree d. Since the coefficients of $f_0 - f_i$ are all in $\{-1, 0, 1\}$ and $\pi_j(\pm 1) = \pm 1$, $\pi_j(f_0 - f_i)$ is a non-zero polynomial for $j \in \{1, 2\}$ as well.

- We prove $S_0 \supseteq \{-f_i(\{y_{j,j'}\}_{(j,j')\in[d]\times[\ell]}) | y_{1,1}, \dots, y_{d,\ell} \in S_1\}$ for all $i \in [0, Q]$. By our assumption $d(c-1) < n$ and by regarding elements $y_{j,j'}$ as polynomials in $\mathbb{Z}[X]/(X^n + 1)$ with degree $c - 1$, we have $f_i(\{y_{j,j'}\})$ are all in $[*, *]_{R, d(c-1)+1}$ where $*$ represents some integer. It then suffices to show $\|\phi(f_i(\{y_{j,j'}\}_{(j,j')\in[d]\times[\ell]}))\|_\infty \leq \kappa(cn)^d$. For any $\{y_{j,j'}\}_{(j,j')\in[d]\times[\ell]}$, we have

$$\begin{aligned}
\|\phi(f_i(\{y_{j,j'}\}_{(j,j')\in[d]\times[\ell]}))\|_\infty &= \left\|\phi\left(\sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}_i)} y_{1,j'_1} y_{2,j'_2} \cdots y_{d,j'_d}\right)\right\|_\infty && (4.52)\\
&= \left\|\sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}_i)} \phi(y_{1,j'_1} y_{2,j'_2} \cdots y_{d,j'_d})\right\|_\infty && (4.53)\\
&\leq \sum_{(j'_1,\dots,j'_d)\in S(\mathsf{ID}_i)} \left\|\phi(y_{1,j'_1} y_{2,j'_2} \cdots y_{d,j'_d})\right\|_\infty && (4.54)\\
&\leq \kappa(cn)^d && (4.55)
\end{aligned}$$

  where Eq.(4.52) follows from the definition, Eq.(4.53) holds because ϕ^{-1} is a homomorphism, Eq.(4.54) is from the triangle inequality, and Eq.(4.55) is from Lemma 4.3 and the fact that $\|y_{j,j'}\|_\infty \leq n$.

This completes the proof of Lemma 4.9. □

Lemma 4.10. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[X_5] - \Pr[X_6]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{RLWE}_{n,k+1,q,D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}}}.$$

In particular, under the $\mathsf{RLWE}_{n,k+1,q,D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}}$ assumption, we have $|\Pr[X_5] - \Pr[X_6]| = \mathsf{negl}(n)$.

Proof. Suppose an adversary $\mathcal{A}$ that has non-negligible advantage in distinguishing Game_5 and Game_6. We use $\mathcal{A}$ to construct an RLWE algorithm denoted $\mathcal{B}$, which proceeds as follows.

Instance. $\mathcal{B}$ is given the problem instance of RLWE $(\{a_i, v_i\}_{i=0}^{k}) \in (R_q \times R_q)^{k+1}$. We can assume without loss of generality that $v_i = v_i' + x_i$ for $x_i \xleftarrow{\$} D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}$. Then $\mathcal{B}$'s task is to distinguish whether $v_i' = a_i s$ for some $s \in R_q$ or $v_i' \xleftarrow{\$} R_q$. We note this subtle change from the standard RLWE problem is done only for convenience of the proof.

Setup. To construct master public key mpk, $\mathcal{B}$ first sets

$$u := a_0, \quad \boldsymbol{a} := (a_1, \ldots, a_k), \quad v_0 := v_0, \quad \boldsymbol{v} := (v_1, \ldots, v_k)$$

It also picks $\boldsymbol{y}$ as in Game_1, $\boldsymbol{R}_0, \boldsymbol{R}_{i,j}$ as in Game_2 and sets $\boldsymbol{b}_0$ and $\boldsymbol{b}_{i,j}$ as in Eq.(4.42). Finally, it returns $\mathsf{mpk} = (\boldsymbol{a}, \boldsymbol{b}_0, \{\boldsymbol{b}_{i,j}\}_{(i,j)\in[d,\ell]}, u)$ to $\mathcal{A}$. $\mathcal{B}$ also picks a random bit $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and keeps it secret.

Phase 1 and Phase 2. The key extraction queries made by $\mathcal{A}$ are answered as in Game_4. This is done by using $\boldsymbol{R}_0$ and $\boldsymbol{R}_{i,j}$.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge identity $\mathsf{ID}^\star$ and message m, $\mathcal{B}$ first computes $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star)$. Then, it aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ if $\mathsf{F}_{\boldsymbol{y}}(\mathsf{ID}^\star) \neq 0$. Otherwise, it proceeds as follows. If $\mathsf{coin} = 0$, it computes $\boldsymbol{R}_{\mathsf{ID}^\star}$ and $\mathbf{c} \in \mathbb{Z}_q^{2k}$ as in Eq.(4.47). It then sets the challenge ciphertext $C^\star$ as in Eq. (4.46). In the case of $\mathsf{coin} = 1$, $\mathcal{B}$ picks $c_0 \xleftarrow{\$} R_q$, $\boldsymbol{c}_1 \xleftarrow{\$} R_q^{2k}$ and sets $C^\star = (c_0, \mathbf{c}_1)$. In both cases, $\mathcal{B}$ returns $C^\star$ to $\mathcal{A}$.

Guess. At last, $\mathcal{A}$ outputs its guess $\widehat{\mathsf{coin}}$ (if the abort condition has not been satisfied). Then, $\mathcal{B}$ sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

Analysis. It can be seen that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in Game_5 if $\{a_i, v_i' + x_i\}_{i=0}^{k}$ are valid RLWE samples (i.e., $v_i' = a_i s$) and Game_6 otherwise (i.e., $v_i' \xleftarrow{\$} R_q$). We therefore conclude that $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{RLWE}_{n,k+1,q,D^{\mathsf{coeff}}_{\mathbb{Z}^n,\alpha q}}} = |\Pr[X_5] - \Pr[X_6]|$ as desired.

□

4.5 Construction from Bilinear Maps

4.5.1 Single-bit Variant

In the following, we present our IBE scheme from bilinear maps. Here, for simplicity, we first present the scheme with only single-bit message space. A variant of our scheme that can deal with longer message space will appear later. Let the identity space of the scheme be $\mathcal{ID} = \{0,1\}^\kappa$ for some $\kappa \in \mathbb{N}$. For our construction, we consider an efficiently computable injective map S that maps an identity $\mathsf{ID} \in \{0,1\}^\kappa$ to a subset $S(\mathsf{ID})$ of $[1,\ell] \times [1,\ell]$, where $\ell = \lceil\sqrt{\kappa}\rceil$. We would typically set $\kappa = O(\lambda)$, and thus $\ell = O(\sqrt{\lambda})$ in such a case. We also use $\mathsf{GL}(\mathsf{K}, \mathsf{rand})$ to denote the Goldreich-Levin hardcore bit [GL89] of K using randomness rand. Recall that $\mathsf{GL}(\mathsf{K}, \mathsf{rand})$ is the bitwise inner product between K and rand.

$\mathsf{Setup}(1^\lambda)$: On input 1^λ, it chooses an asymmetric bilinear group $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ with efficiently computable map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order $p = p(\lambda)$. Let g and h be generators of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. It then picks $w_0, w_{1,1}, \ldots, w_{1,\ell}, w_{2,1}, \ldots, w_{2,\ell}, \alpha, \beta \xleftarrow{\$} \mathbb{Z}_p$ and $\mathsf{rand} \xleftarrow{\$} \{0,1\}^{|\mathbb{G}_T|}$. It finally outputs

$$\begin{aligned} \mathsf{mpk} &= (g, W_0 = g^{w_0}, \{W_{1,i} = g^{w_{1,i}}\}_{i=1}^{\ell}, \{W_{2,i} = g^{w_{2,i}}\}_{i=1}^{\ell}, g^\alpha, h^\beta, \mathsf{rand}) \quad \text{and} \\ \mathsf{msk} &= (h, \alpha, \beta, w_0, w_{1,1}, \ldots, w_{1,\ell}, w_{2,1}, \ldots, w_{2,\ell}) \end{aligned}$$

In the following, we use a deterministic function $\mathsf{H} : \mathcal{ID} \to \mathbb{Z}_p$ that is defined as follows.

$$\mathsf{H}(\mathsf{ID}) = w_0 + \sum_{(i,j) \in S(\mathsf{ID})} w_{1,i} w_{2,j} \in \mathbb{Z}_p. \tag{4.56}$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: It first computes $\mathsf{H}(\mathsf{ID})$ using msk and picks $r \xleftarrow{\$} \mathbb{Z}_p$. It then returns

$$\mathsf{sk}_{\mathsf{ID}} = (\ A_1 = h^{\alpha\beta + r \cdot \mathsf{H}(\mathsf{ID})},\ A_2 = h^{-r},\ \{B_j = h^{r w_{2,j}}\}_{j=1}^{\ell}\ ). \tag{4.57}$$

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}$, it picks $s, t_1, \ldots, t_\ell \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$\begin{aligned} C_0 &= \mathsf{M} \oplus \mathsf{GL}\big(e(g^\alpha, h^\beta)^s, \mathsf{rand}\big), \quad C_1 = g^s, \quad C_2 = W_0^s \cdot \prod_{j \in [1,\ell]} W_{2,j}^{t_j}, \\ D_j &= g^{t_j} \cdot \left(\prod_{i \in \{i \in [1,\ell] | (i,j) \in S(\mathsf{ID})\}} W_{1,i} \right)^{-s} \qquad \text{for} \quad j \in [1,\ell] \end{aligned} \tag{4.58}$$

Finally, it returns the ciphertext $C = (C_0, C_1, C_2, \{D_j\}_{j=1}^{\ell})$.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C)$: To decrypt a ciphertext $C = (C_0, C_1, C_2, \{D_j\}_{j=1}^{\ell})$ using a private key $\mathsf{sk}_{\mathsf{ID}} = (A_1, A_2, \{B_j\}_{j=1}^{\ell})$, it first computes

$$e(C_1, A_1) \cdot e(C_2, A_2) \cdot \prod_{j \in [1,\ell]} e(D_j, B_j) = e(g,h)^{s\alpha\beta}. \tag{4.59}$$

Then it retrieves the message by $C_0 \oplus \mathsf{GL}(e(g,h)^{s\alpha\beta}, \mathsf{rand})$.

Correctness of the Single-bit Variant

To verify the correctness of the scheme, it suffices to show Eq.(4.59). Let $g_T := e(g,h)$. We have

$$\begin{aligned} &\log_{g_T} \left(e(C_1, A_1) \cdot e(C_2, A_2) \cdot \prod_{j \in [1,\ell]} e(D_j, B_j) \right) \\ = \ &\log_{g_T} e(C_1, A_1) - r \left(s w_0 + \sum_{j \in [1,\ell]} t_j w_{2,j} \right) + \sum_{j \in [1,\ell]} r w_{2,j} \left(t_j - s \sum_{i \in \{i \in [1,\ell] | (i,j) \in S(\mathsf{ID})\}} w_{1,i} \right) \\ = \ &\log_{g_T} e(C_1, A_1) - r s w_0 - r s \sum_{j \in [1,\ell]} \left(\sum_{i \in \{i \in [1,\ell] | (i,j) \in S(\mathsf{ID})\}} w_{1,i} w_{2,j} \right) \end{aligned}$$

$$\begin{aligned}
&= s\alpha\beta + rs\left(w_0 + \sum_{(i,j)\in S(\mathsf{ID})} w_{1,i}w_{2,j}\right) - rs\left(w_0 + \sum_{(i,j)\in S(\mathsf{ID})} w_{1,i}w_{2,j}\right)\\
&= s\alpha\beta.
\end{aligned}$$

Therefore, Eq.(4.59) follows.

Security Proof for the Single-bit Variant

The security of the scheme is proven under the 3-CBDHE assumption defined below.

Definition 4.1 (3-Computational Bilinear Diffie-Hellman Exponent (3-CBDHE) Assumption)**.** *We say that* 3*-CBDHE holds on* $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ *if*

$$\Pr[\mathcal{A}(g, g^s, g^a, g^{a^2}, h, h^a, h^{a^2}) \to e(g,h)^{sa^3}]$$

is negligible for any PPT adversary $\mathcal{A}$ *where* $g \xleftarrow{\$} \mathbb{G}_1$*,* $h \xleftarrow{\$} \mathbb{G}_2$*,* $s, a \xleftarrow{\$} \mathbb{Z}_p$*.*

We also introduce the following lemma concerning the Goldreich-Levin hardcore bit function which we use during our security proof.

Lemma 4.11 ([GL89])**.** *Let us assume that the 3-CBDHE assumption holds. Then, for any PPT adversary* $\mathcal{A}$*,*

$$\mathsf{Adv}^{\mathsf{3CBDHE}}_{\mathcal{A}} = \left|\Pr[\mathcal{A}\big(\Psi, \mathsf{rand}, \mathsf{GL}(e(g,h)^{sa^3}, \mathsf{rand})\big) \to 1] - \Pr[\mathcal{A}\big(\Psi, \mathsf{rand}, T\big) \to 1]\right|$$

is negligible where $\Psi = (g, g^s, g^a, g^{a^2}, h, h^a, h^{a^2})$*,* $a, s \xleftarrow{\$} \mathbb{Z}_p$*,* $T \xleftarrow{\$} \{0,1\}$ *and* $\mathsf{rand} \xleftarrow{\$} \{0,1\}^{|\mathbb{G}_T|}$*.*

The following theorem addresses the security of the scheme.

Theorem 4.2. *The above IBE scheme is adaptively secure assuming the* 3*-CBDHE assumption.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks adaptive security of the scheme. In addition, let $\epsilon = \epsilon(\lambda)$ and $Q = Q(\lambda)$ be its advantage and the upper bound on the number of key extraction queries, respectively. Since $\mathcal{A}$ is PPT, there exists a constant number $c_1 \in \mathbb{N}$ such that $4(Q+1) \le \lambda^{c_1}$ for all $\lambda \in \mathbb{N}$. Similarly, since $\mathcal{A}$ breaks the security of the scheme, there exists $c_2 \in \mathbb{N}$ such that $2\epsilon \ge \lambda^{-c_2}$ holds for infinitely many λ. By setting $c = c_1 + c_2$, we have that

$$4Q \le \lambda^c \quad \text{for all } \lambda \in \mathbb{N} \quad \text{and} \quad \frac{\epsilon}{2(Q+1)} \ge \frac{1}{\lambda^c} \quad \text{for infinitely many } \lambda \in \mathbb{N}. \tag{4.60}$$

In the following, we assume that $p > \lambda^c$. Since the size of p is exponential in λ, this holds for sufficiently large λ.

We show the security of the scheme via the following games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. While it is set $\mathsf{coin}' = \widehat{\mathsf{coin}}$ in the first game, these values might be different in the later games. In the following, we define X_i be the event that $\mathsf{coin}' = \mathsf{coin}$ in Game_i.

Game_0 : This is the real security game. Since the message space is $\{0,1\}$, without loss of generality, we assume that the adversary always chooses $\mathsf{M}_0 = 0$ and $\mathsf{M}_1 = 1$ as its target in the challenge phase. Then the challenger picks a random coin $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and returns an encryption of $\mathsf{M}_{\mathsf{coin}} = \mathsf{coin}$ as the challenge ciphertext. At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By the definition, we have

$$\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : In this game, we change Game_0 so that the challenger performs the following additional step at the end of the game. First, the challenger picks $\mathbf{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[2]\times[\ell]})$ as

$$y_0 \xleftarrow{\$} [-\kappa\lambda^{2c}, -1] \quad \text{and} \quad y_{i,j} \xleftarrow{\$} [1, \lambda^c] \quad \text{for} \quad (i,j) \in [2]\times[\ell]. \tag{4.61}$$

We define a function $\mathsf{F}_{\mathbf{y}} : \mathcal{ID} \to \mathbb{Z}_p$ as follows:

$$\mathsf{F}_{\mathbf{y}}(\mathsf{ID}) = y_0 + \sum_{(j_1,j_2)\in S(\mathsf{ID})} y_{1,j_1} y_{2,j_2}.$$

Then the challenger checks whether the following condition holds:

$$\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star) = 0 \ \wedge \ \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_1) \neq 0 \ \wedge \ \cdots \ \wedge \ \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_Q) \neq 0 \tag{4.62}$$

where $\mathsf{ID}^\star$ is the challenge identity, and $\mathsf{ID}_1, \ldots, \mathsf{ID}_Q$ are identities for which $\mathcal{A}$ has made key extraction queries. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$, and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$. Otherwise, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. In Lemma 4.12, we will show that

$$\left| \Pr[X_1] - \frac{1}{2} \right| \geq \frac{1}{\kappa\lambda^{2c}} \left(\frac{\epsilon}{2} - \frac{Q}{\lambda^c} \right).$$

Game_2 In this game, we change the way α, β, w_0, and $w_{i,j}$ are chosen. At the beginning of the game, the challenger picks $\mathbf{y}$ as in Game_1. It then picks $a, \tilde{w}_0, \tilde{w}_{1,1}, \ldots, \tilde{w}_{1,\ell}, \tilde{w}_{2,1}, \ldots, \tilde{w}_{2,\ell} \xleftarrow{\$} \mathbb{Z}_p$, $\tilde{\alpha}, \tilde{\beta} \xleftarrow{\$} \mathbb{Z}_p^*$ and sets

$$\alpha = a\tilde{\alpha}, \ \beta = a^2\tilde{\beta}, \ w_0 = a^2 y_0 + \tilde{w}_0, \ w_{i,j} = a y_{i,j} + \tilde{w}_{i,j} \text{ for } (i,j) \in [2]\times[\ell]. \tag{4.63}$$

This change does not alter the distribution of w_0, $w_{i,j}$, α, and β. Since this change is only conceptual, we have

$$\Pr[X_2] = \Pr[X_1].$$

Game_3 Recall that in the previous game, the challenger aborts at the end of the game, if the condition (4.62) is not satisfied. In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have

$$\Pr[X_3] = \Pr[X_2].$$

Before describing the next game, we observe that $\mathsf{H}(\mathsf{ID})$ can be written as an polynomial in a with degree 2 whose coefficients depend on ID and $\mathbf{y}$.

$$\begin{aligned} & \mathsf{H}(\mathsf{ID}) \\ = \ & w_0 + \sum_{(i,j)\in S(\mathsf{ID})} w_{1,i} w_{2,j} \\ = \ & y_0 a^2 + \tilde{w}_0 + \sum_{(i,j)\in S(\mathsf{ID})} (y_{1,i} a + \tilde{w}_{1,i})(y_{2,j} a + \tilde{w}_{2,j}) \end{aligned}$$

$$\begin{aligned}
&= \underbrace{\left(y_0 + \sum_{(i,j)\in S(\mathsf{ID})} y_{1,i}y_{2,j}\right)}_{=\mathsf{F_y}(\mathsf{ID})} a^2 + \underbrace{\left(\sum_{(i,j)\in S(\mathsf{ID})} \tilde{w}_{1,i}y_{2,j} + y_{1,i}\tilde{w}_{2,j}\right)}_{:=\mathsf{G_y}(\mathsf{ID})} a + \underbrace{\left(\tilde{w}_0 + \sum_{(i,j)\in S(\mathsf{ID})} \tilde{w}_{1,i}\tilde{w}_{2,j}\right)}_{:=\mathsf{I_y}(\mathsf{ID})} \\
&= \mathsf{F_y}(\mathsf{ID})a^2 + \mathsf{G_y}(\mathsf{ID})a + \mathsf{I_y}(\mathsf{ID}).
\end{aligned}$$

Game$_4$: In this game, we change the way the key extraction queries are answered. When $\mathcal{A}$ makes a key extraction query for an identity ID, the challenger aborts if $\mathsf{F_y}(\mathsf{ID}) = 0$ as the previous game. Otherwise, it first picks $\tilde{r} \overset{\$}{\leftarrow} \mathbb{Z}_p$ and sets r as

$$r = \tilde{r} - \frac{\tilde{\alpha}\tilde{\beta}}{\mathsf{F_y}(\mathsf{ID})} a. \tag{4.64}$$

Then the private key is generated as Eq.(4.57). Clearly, this is only a conceptual change and does not change the view of $\mathcal{A}$. Therefore, we have

$$\Pr[X_4] = \Pr[X_3].$$

Here, we observe that

$$\begin{aligned}
&\alpha\beta + r\mathsf{H}(\mathsf{ID}) \\
=\ & a^3\tilde{\alpha}\tilde{\beta} + \left(\mathsf{F_y}(\mathsf{ID})a^2 + \mathsf{G_y}(\mathsf{ID})a + \mathsf{I_y}(\mathsf{ID})\right)\left(\tilde{r} - \frac{\tilde{\alpha}\tilde{\beta}}{\mathsf{F_y}(\mathsf{ID})}a\right) \\
=\ & \left(\tilde{r}\mathsf{F_y}(\mathsf{ID}) - \frac{\tilde{\alpha}\tilde{\beta}\mathsf{G_y}(\mathsf{ID})}{\mathsf{F_y}(\mathsf{ID})}\right)a^2 + \left(\tilde{r}\mathsf{G_y}(\mathsf{ID}) - \frac{\tilde{\alpha}\tilde{\beta}\mathsf{I_y}(\mathsf{ID})}{\mathsf{F_y}(\mathsf{ID})}\right)a + \tilde{r}\cdot\mathsf{I_y}(\mathsf{ID}) \qquad (4.65)
\end{aligned}$$

and

$$\begin{aligned}
rw_{2,j} &= \left(\tilde{r} - \frac{\tilde{\alpha}\tilde{\beta}}{\mathsf{F_y}(\mathsf{ID})}a\right)(y_{2,j}a + \tilde{w}_{2,j}) \\
&= -\frac{\tilde{\alpha}\tilde{\beta}y_{2,j}}{\mathsf{F_y}(\mathsf{ID})}a^2 + \left(\tilde{r}y_{2,j} - \frac{\tilde{\alpha}\tilde{\beta}\tilde{w}_{2,j}}{\mathsf{F_y}(\mathsf{ID})}\right)a + r\tilde{w}_{2,j}. \qquad (4.66)
\end{aligned}$$

It can be seen that the term $a^3\tilde{\alpha}\tilde{\beta}$ cancels out in Eq.(4.65). Looking ahead, this is essential for the reduction from the 3-CBDHE assumption (Lemma 4.13) to be possible.

Game$_5$ In this game, we change the way the challenge ciphertext is created. When creating the challenge ciphertext, the challenger first picks $s', \tilde{t}_1, \dots, \tilde{t}_\ell \overset{\$}{\leftarrow} \mathbb{Z}_p$ and sets

$$s = \frac{s'}{\tilde{\alpha}\tilde{\beta}}, \quad t_j = \begin{cases} \tilde{t}_1 + s\left(-\dfrac{\mathsf{G_y}(\mathsf{ID}^\star)}{y_{2,1}} + \displaystyle\sum_{i\in\{i\in[1,\ell] \mid (i,1)\in S(\mathsf{ID}^\star)\}} w_{1,i}\right) & \text{for } j = 1 \\ \tilde{t}_j + s\left(\displaystyle\sum_{i\in\{i\in[1,\ell] \mid (i,j)\in S(\mathsf{ID}^\star)\}} w_{1,i}\right) & \text{for } j \in [2,\ell]. \end{cases} \tag{4.67}$$

Then, the challenge ciphertext is computed as Eq.(4.58). Note that since $1 \le y_{2,1} \le \lambda^c < p$ and thus $y \ne 0 \mod p$, the denominator in Eq.(4.67) is well-defined. Clearly, this is only a conceptual change and does not change the view of $\mathcal{A}$. Therefore, we have

$$\Pr[X_5] = \Pr[X_4].$$

Here, we observe that

$$C_0 = \mathsf{coin} \oplus \mathsf{GL}\big(e(g,h)^{s'a^3}, \mathsf{rand}\big),\ D_1 = g^{\tilde{t}_1}(g^{s'})^{-\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)/\tilde{\alpha}\tilde{\beta}y_{2,1}},\ D_j = g^{\tilde{t}_j} \ \text{ for } \ j \in [2,\ell] \tag{4.68}$$

and

$$\begin{aligned}
& \log_g C_2 \\
= \ & sw_0 + \sum_{j\in[1,\ell]} w_{2,j} t_j \\
= \ & sw_0 - w_{2,1}s\left(\frac{\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)}{y_{2,1}}\right) + \sum_{j\in[1,\ell]} w_{2,j}\left(\tilde{t}_j + s\left(\sum_{i\in\{i\in[1,\ell]|(i,j)\in S(\mathsf{ID}^\star)\}} w_{1,i}\right)\right) \\
= \ & -w_{2,1}s\left(\frac{\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)}{y_{2,1}}\right) + \left(\sum_{j\in[1,\ell]} w_{2,j}\tilde{t}_j\right) + s\underbrace{\left(w_0 + \sum_{j\in[1,\ell]}\sum_{i\in\{i\in[1,\ell]|(i,j)\in S(\mathsf{ID}^\star)\}} w_{1,i}w_{2,j}\right)}_{=\mathsf{H}(\mathsf{ID}^\star)} \\
= \ & -s(y_{2,1}a + \tilde{w}_{2,1})\left(\frac{\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)}{y_{2,1}}\right) + \left(\sum_{j\in[1,\ell]} w_{2,j}\tilde{t}_j\right) + s\left(\underbrace{\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star)}_{=0}a^2 + \mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)a + \mathsf{I}_{\mathbf{y}}(\mathsf{ID}^\star)\right) \\
= \ & -\cancel{\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)sa} - s\left(\frac{\tilde{w}_{2,1}\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)}{y_{2,1}}\right) + \left(\sum_{j\in[1,\ell]} w_{2,j}\tilde{t}_j\right) + \cancel{\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)sa} + s\cdot\mathsf{I}_{\mathbf{y}}(\mathsf{ID}^\star) \\
= \ & s'\left(\frac{y_{2,1}\cdot\mathsf{I}_{\mathbf{y}}(\mathsf{ID}^\star) - \tilde{w}_{2,1}\cdot\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)}{\tilde{\alpha}\tilde{\beta}y_{2,1}}\right) + a\left(\sum_{j\in[1,\ell]} y_{2,j}\tilde{t}_j\right) + \left(\sum_{j\in[1,\ell]} \tilde{w}_{2,j}\tilde{t}_j\right).
\end{aligned} \tag{4.69}$$

It can be seen that the term $-\mathsf{G}_{\mathbf{y}}(\mathsf{ID}^\star)sa$ cancels out in Eq.(4.69). Looking ahead, this is essential for the reduction from the 3-CBDHE assumption (Lemma 4.13) to be possible.

Game_6 In this game, the component C_0 in the challenge ciphertext is changed to be a random bit. As we will show in Lemma 4.13, assuming the 3-CBDHE assumption is hard, we have

$$|\Pr[X_6] - \Pr[X_5]| = \mathsf{negl}(n). \tag{4.70}$$

Analysis. From the above, we have

$$\begin{aligned}
\left|\Pr[X_6] - \frac{1}{2}\right| &= \left|\Pr[X_1] - \frac{1}{2} + \sum_{i=1}^{5}\Pr[X_{i+1}] - \Pr[X_i]\right| \\
&\geq \left|\Pr[X_1] - \frac{1}{2}\right| - \sum_{i=1}^{5}|\Pr[X_{i+1}] - \Pr[X_i]| \\
&\geq \frac{1}{\kappa\lambda^{2c}}\left(\frac{\epsilon}{2} - \frac{Q}{\lambda^c}\right) - \mathsf{negl}(\lambda) \\
&= \frac{1}{\mathsf{poly}(\lambda)}\left(\frac{\epsilon}{2} - \frac{Q}{\lambda^c}\right) - \mathsf{negl}(\lambda).
\end{aligned} \tag{4.71}$$

Since the challenge ciphertext is independent from the value of coin in Game_6, we have $\Pr[X_6] = 1/2$ and thus $|\Pr[X_6] - 1/2| = 0$. Therefore, we have that $\epsilon/2 - Q/\lambda^c$ is negligible. However, by Eq.(4.60),

$$\frac{\epsilon}{2} - \frac{Q}{\lambda^c} \geq \frac{Q+1}{\lambda^c} - \frac{Q}{\lambda^c} = \frac{1}{\lambda^c}$$

holds for infinitely many λ, which is a contradiction. $\square$

To complete the proof of Theorem 4.2, it remains to show Lemma 4.12 and Lemma 4.13.

Lemma 4.12. *For any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr[X_1] - \frac{1}{2}\right| \geq \frac{1}{\kappa\lambda^{2c}}\left(\frac{\epsilon}{2} - \frac{Q}{\lambda^c}\right).$$

Proof. For a sequence of identities $\mathbb{ID} = (\mathsf{ID}^\star, \mathsf{ID}_1, \dots, \mathsf{ID}_Q) \in \mathcal{ID}^{Q+1}$, we define $\gamma(\mathbb{ID})$ as

$$\gamma(\mathbb{ID}) = \Pr_{\mathbf{y}}[\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star) = 0 \wedge \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_1) \neq 0 \wedge \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_2) \neq 0 \wedge \cdots \wedge \mathsf{F}_{\mathbf{y}}(\mathsf{ID}_Q) \neq 0]$$

where the probability is taken over $\boldsymbol{y} = (y_0, \{y_{i,j}\}_{(i,j)\in[2,\ell]})$, which is chosen as specified in Game_1. It suffices to show

$$\frac{1}{\kappa\lambda^{2c}}\left(1 - \frac{2Q}{\lambda^c}\right) \leq \gamma(\mathbb{ID}) \leq \frac{1}{\kappa\lambda^{2c}} \tag{4.72}$$

since due to Lemma 4.4, this implies

$$\begin{aligned}\left|\Pr[X_1] - \frac{1}{2}\right| &\geq \frac{\epsilon}{\kappa\lambda^{2c}}\left(1 - \frac{2Q}{\lambda^c}\right) - \frac{1}{2\kappa\lambda^{2c}}\left(1 - \left(1 - \frac{2Q}{\lambda^c}\right)\right) \\ &\geq \frac{1}{\kappa\lambda^{2c}}\left(\frac{\epsilon}{2} - \frac{Q}{\lambda^c}\right)\end{aligned}$$

where we used Eq.(4.60) in the last inequality. In the following, we will prove Eq.(4.72) by applying Lemma 4.5. We would set

$$\begin{array}{lll} d = 2, & \nu = 1, & \Phi = \Omega_1 = \mathbb{Z}_p, \\ \Pi = \pi_1 = \mathrm{id}_{\mathbb{Z}_p}, & S_0 = [-\kappa\lambda^{2c}, -1], & S_1 = [1, \lambda^c] \end{array}$$

where $\mathrm{id}_{\mathbb{Z}_p}$ denotes the identity map on $\mathbb{Z}_p$. We set $\mu = 2\ell$ and define $f_i(\{Y_{j,j'}\}_{(j,j')\in[2]\times[\ell]})$ for $i \in [0, Q]$ as

$$f_i\left(\{Y_{j,j'}\}_{(j,j')\in[2]\times[\ell]}\right) = \sum_{(j'_1,j'_2)\in S(\mathsf{ID}_i)} Y_{1,j'_1} Y_{2,j'_2}$$

where we define $\mathsf{ID}_0 := \mathsf{ID}^\star$. Note that we have $\mathsf{F}_{\mathbf{y}}(\mathsf{ID}_i) = y_0 + f_i(\{y_{j,j'}\}_{(j,j')\in[2]\times[\ell]})$. We now check that the three conditions for Lemma 4.5 hold.

- π_1 is injective on S_1 because it is the identity map on $\mathbb{Z}_p$ and $\lambda^c < p$.

- For $i \in [1, Q]$, we have
$$f_0\left(\{Y_{j,j'}\}\right) - f_i\left(\{Y_{j,j'}\}\right) = \sum_{(j_1',j_2') \in S(\mathsf{ID}^\star)} Y_{1,j_1'} Y_{2,j_2'} - \sum_{(j_1',j_2') \in S(\mathsf{ID}_i)} Y_{1,j_1'} Y_{2,j_2'}.$$
Since $\mathsf{ID}^\star \neq \mathsf{ID}_i$ and S is an injective map, we have $S(\mathsf{ID}^\star) \neq S(\mathsf{ID}_i)$. Therefore, there exists $(j_1^\star, j_2^\star) \in [\ell] \times [\ell]$ such that $(j_1^\star, j_2^\star) \in S(\mathsf{ID}^\star) \triangle S(\mathsf{ID}_i)$, where $S(\mathsf{ID}^\star) \triangle S(\mathsf{ID}_i)$ denotes the symmetric difference of $S(\mathsf{ID}^\star)$ and $S(\mathsf{ID}_i)$. Thus, the above polynomial is a non-zero polynomial with degree 2.

- Since $S_1 = [1, \lambda^c]$, we have
$$1 \leq f_i(\{y_{j,j'}\}) = \sum_{(j_1',j_2') \in S(\mathsf{ID}_i)} y_{1,j_1'} y_{2,j_2'} \leq \sum_{(j_1',j_2') \in S(\mathsf{ID}_i)} \lambda^c \cdot \lambda^c \leq \kappa \lambda^{2c}$$
for $i \in [Q]$. Therefore, we have $S_0 \supseteq \{-f_i(\{y_{j,j'}\}_{(j,j') \in [2] \times [\ell]}) | y_{j,j'} \in S_1\}$ for all $i \in [0, Q]$.

This completes the proof of Lemma 4.12. $\square$

Lemma 4.13. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*
$$|\Pr[X_5] - \Pr[X_6]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{3CBDHE}}.$$
In particular, under the 3CBDHE *assumption, we have* $|\Pr[X_5] - \Pr[X_6]| = \mathsf{negl}(n)$.

Proof. Suppose an adversary $\mathcal{A}$ that has non-negligible advantage in distinguishing Game_5 and Game_6. We use $\mathcal{A}$ to construct an 3CBDHE algorithm denoted $\mathcal{B}$, which proceeds as follows.

Instance. $\mathcal{B}$ is given the problem instance of 3CBDHE $\left(g, g^{s'}, g^a, g^{a^2}, h, h^a, h^{a^2}, \mathsf{rand}, T\right)$. The task of $\mathcal{B}$ is to distinguish whether $T = \mathsf{GL}(e(g,h)^{s'a^3}, \mathsf{rand})$ or $T \xleftarrow{\$} \{0,1\}$.

Setup. To construct master public key mpk, $\mathcal{B}$ first picks $\mathbf{y}$ as in Game_2. It also picks $\tilde{w}_0, \tilde{w}_{i,j}, \tilde{\alpha}, \tilde{\beta}$ and implicitly sets $w_0, w_{i,j}, \alpha, \beta$ as in Game_3. Then, $\mathcal{B}$ computes mpk as follows:
$$\mathsf{mpk} = \left(g, \begin{array}{ll} g^\alpha = (g^a)^{\tilde{\alpha}}, & W_0 = (g^{a^2})^{y_0} \cdot g^{\tilde{w}_0}, \\ h^\beta = (h^{a^2})^{\tilde{\beta}}, & \{W_{i,j} = (g^a)^{y_{i,j}} \cdot g^{\tilde{w}_{i,j}}\}_{(i,j) \in [2,\ell]}, \end{array} \mathsf{rand} \right). \tag{4.73}$$
Note that these values can be computed without explicitly knowing a. Finally, it returns mpk to $\mathcal{A}$. $\mathcal{B}$ also picks a random bit $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and keeps it secret.

Phase 1 and Phase 2. When $\mathcal{A}$ makes a key extraction query for ID, $\mathcal{B}$ proceeds as follows. We assume $\mathsf{F}(\mathsf{ID}) \neq 0$ since otherwise $\mathcal{B}$ aborts. By the change introduced in Game_4, we have that each component of $\mathsf{sk}_{\mathsf{ID}}$ can be written as a linear combination of (h, h^a, h^{a^2}) with the coefficients being known to $\mathcal{B}$ (See Eq.(4.64), (4.65), and (4.66)). Therefore, $\mathcal{B}$ can compute the secret key without explicitly knowing the value of a.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge identity $\mathsf{ID}^\star$, $\mathcal{B}$ proceeds as follows. We assume $\mathsf{F}_{\mathbf{y}}(\mathsf{ID}^\star) \neq 0$ since otherwise $\mathcal{B}$ aborts. By the change introduced in Game_6, C_1, C_2, $\{D_j\}_{j=1}^{\ell}$ in the challenge ciphertext can be written as a linear combination of $(g^{s'}, g, g^a, g^{a^2})$ (See Eq.(4.68) and (4.69)). $\mathcal{B}$ can therefore compute these components. Finally, $\mathcal{B}$ sets $C_0 = T \oplus \mathsf{coin}$ and gives the challenge ciphertext $C^\star = (C_0, C_1, C_2, \{D_j = g^{t_j}\}_{j \in [1,\ell]})$ to $\mathcal{A}$.

Guess. At last, $\mathcal{A}$ outputs its guess $\widehat{\mathsf{coin}}$ (if the abort condition has not been satisfied). Then, $\mathcal{B}$ sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

Analysis. It can be seen that the view of $\mathcal{A}$ corresponds to that in Game_5 if $T = \mathsf{GL}(e(g,h)^{s'a^3}, \mathsf{rand})$ and Game_6 if $T \xleftarrow{\$} \{0,1\}$. Therefore, we have $|\Pr[X_5] - \Pr[X_6]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{3CBDHE}}$. $\square$

4.5.2 Multi-bit Variant

Let us try to extend our single-bit scheme in the previous section to be a multi-bit scheme with message space $\{0,1\}^{\ell_M}$ for some $\ell_M \in \mathbb{N}$. The most obvious way to achieve this is to just run the encryption algorithm ℓ_M times. However, this naive method will make the ciphertext ℓ_M times longer. Another way would be to prepare ℓ_M copies of g^α and h^β and put them into the master public key. However, this approach will result in a scheme with master public key containing extra $O(\ell_M)$ group elements. In this section, we show that it is possible to obtain a multi-bit scheme with the same ciphertext-size as the single-bit scheme, by adding only $O(\sqrt{\ell_M})$ group elements to the master public key. This can be accomplished by incorporating our single bit scheme in Section 4.5 with the technique from [HJKS10, YKHK10].

For simplicity, we assume that $\ell_M = (\ell')^2$ for some $\ell' \in \mathbb{N}$ in the following.

$\mathsf{Setup}(1^\lambda)$: On input 1^λ, it chooses an asymmetric bilinear group $\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T$ with efficiently computable map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$ of prime order $p = p(\lambda)$. Let g and h be generator of $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. It then picks $w_0, w_{1,1}, \ldots, w_{1,\ell}, w_{2,1}, \ldots, w_{2,\ell}, \alpha_1, \ldots \alpha_{\ell'}, \beta_1, \ldots, \beta_{\ell'} \xleftarrow{\$} \mathbb{Z}_p$ and $\mathsf{rand} \xleftarrow{\$} \{0,1\}^{|\mathbb{G}_T|}$. It finally outputs

$$\begin{aligned} \mathsf{mpk} &= (g, W_0 = g^{w_0}, \{W_{i,j} = g^{w_{i,j}}\}_{(i,j)\in[2]\times[\ell]}, \{g^{\alpha_i}\}_{i=1}^{\ell'}, \{g^{\beta_i}\}_{i=1}^{\ell'}, \mathsf{rand}) \quad \text{and} \\ \mathsf{msk} &= (h, \{\alpha_i\}_{i\in[\ell']}, \{\beta_i\}_{i\in[\ell']}, w_0, w_{1,1}, \ldots, w_{1,\ell}, w_{2,1}, \ldots, w_{2,\ell}) \end{aligned}$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{ID})$: It first computes $\mathsf{H}(\mathsf{ID})$ (defined as Eq.(4.56)) using msk and picks $r^{(i,j)} \xleftarrow{\$} \mathbb{Z}_p$ for $(i,j) \in [\ell'] \times [\ell']$. It then computes

$$\mathsf{sk}_{\mathsf{ID}}^{(i,j)} = \left(A_1^{(i,j)} = h^{\alpha_i\beta_j + r^{(i,j)}\cdot \mathsf{H}(\mathsf{ID})},\ A_2^{(i,j)} = h^{-r^{(i,j)}},\ \{B_k^{(i,j)} = h^{r^{(i,j)}w_{2,k}}\}_{k=1}^{\ell} \right)$$

for $(i,j) \in [\ell'] \times [\ell']$. It then outputs $\mathsf{sk}_{\mathsf{ID}} = \{\mathsf{sk}_{\mathsf{ID}}^{(i,j)}\}_{(i,j)\in[\ell']\times[\ell']}$.

$\mathsf{Encrypt}(\mathsf{mpk}, \mathsf{ID}, \mathsf{M})$: To encrypt a message $\mathsf{M} = \{0,1\}^{\ell_M}$, it picks $s, t_1, \ldots, t_\ell \xleftarrow{\$} \mathbb{Z}_p$ and computes

$$C_1 = g^s,\ C_2 = W_0^s \cdot \prod_{j\in[1,\ell]} W_{2,j}^{t_j},\ D_j = g^{t_j} \cdot \left(\prod_{i\in\{i\in[1,\ell] \mid (i,j)\in S(\mathsf{ID})\}} W_{1,i} \right)^{-s} \quad \text{for } j \in [1,\ell]$$

It also computes $e((g^{\alpha_i})^s, h^{\beta_j}) = e(g,h)^{s\alpha_i\beta_j}$ and sets

$$\mathsf{K}^{(i,j)} = \mathsf{GL}(e(g,h)^{s\alpha_i\beta_j}, \mathsf{rand})$$

for all $(i,j) \in [\ell', \ell']$. It then sets $\mathsf{K} = \mathsf{K}^{(1,1)} \| \mathsf{K}^{(1,2)} \| \cdots \| \mathsf{K}^{(\ell',\ell')}$ and $C_0 = \mathsf{K} \oplus \mathsf{M}$. Finally, it returns the ciphertext $C = (C_0, C_1, C_2, \{D_j\}_{j=1}^{\ell})$.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{ID}}, C)$: To decrypt a ciphertext $C = (C_0, C_1, C_2, \{D_j\}_{j=1}^{\ell})$ using a private key $\mathsf{sk}_{\mathsf{ID}} = (\{A_1^{(i,j)}, A_2^{(i,j)}, \{B_k^{(i,j)}\}_{k=1}^{\ell}\}_{(i,j)\in[\ell']\times[\ell']})$, it first computes

$$e(C_1, A_1^{(i,j)}) \cdot e(C_2, A_2^{(i,j)}) \cdot \prod_{k\in[1,\ell]} e(D_j, B_k^{(i,j)}) = e(g,h)^{s\alpha_i\beta_j}.$$

for $(i,j) \in [\ell'] \times [\ell']$. Then it sets $\mathsf{K}^{(i,j)} = \mathsf{GL}(e(g,h)^{s\alpha_i\beta_j}, \mathsf{rand})$ and $\mathsf{K} = \mathsf{K}^{(1,1)} \| \mathsf{K}^{(1,2)} \| \cdots \| \mathsf{K}^{(\ell',\ell')}$. Finally, it retrieves the message by $C_0 \oplus \mathsf{K} = \mathsf{M}$.

Correctness of the scheme can be checked similarly to the single-bit version in Section 4.5.

4.5.3 Security of the Multi-bit Variant

Security of the multi-bit scheme is reduced to the security of a certain variant of the single-bit scheme. Concretely, we consider a variant of our single-bit scheme with the master public key being changed to

$$\mathsf{mpk} = (g, W_0 = g^{w_0}, \boxed{h^{w_0}}, \{W_{i,j} = g^{w_{i,j}}\}_{(i,j)\in[2]\times[\ell]}, \boxed{\{h^{w_{2,i}}\}_{i=1}^{\ell}}, g^{\alpha}, h^{\beta}, \boxed{\{h^{w_{1,i}w_{2,j}}\}_{(i,j)\in[\ell]\times[\ell]}}, \mathsf{rand})$$

Namely, we add h^{w_0}, $\{h^{w_{2,i}}\}_{i\in[\ell]}$, and $\{h^{w_{1,i}w_{2,j}}\}_{(i,j)\in[\ell]\times[\ell]}$ to mpk. The rest of the scheme is unchanged. We call the scheme "single bit scheme with redundant key". We claim that the security of this scheme can also be proven under the 3-CBDHE assumption with almost an identical proof to that of Theorem 4.2. The only place where we need to change is Lemma 4.13. Here, we have to simulate the above additional terms. In fact, this can easily be done using the problem instance of the 3-CBDHE assumption, since we have

$$h^{w_0} = (h^{a^2})^{y_0} h^{\tilde{w}_0}, \quad h^{w_{2,i}} = (h^{a})^{y_{2,i}} h^{\tilde{w}_{2,i}}, \quad h^{w_{1,i}w_{2,j}} = (h^{a^2})^{y_{1,i}y_{2,j}} \cdot (h^{a})^{y_{1,i}\tilde{w}_{2,j} + y_{2,j}\tilde{w}_{1,i}} \cdot h^{\tilde{w}_{1,i}\tilde{w}_{2,j}}.$$

Summing up the above discussion, we have the following theorem.

Theorem 4.3. *The single-bit scheme with redundant key is adaptively secure under the 3-CBDHE assumption.*

Therefore, to prove the security of our multi-bit variant, it suffices to show the following.

Theorem 4.4. *Assuming the single-bit scheme with redundant key is adaptively secure, so is the multi-bit scheme.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the adaptive security of the scheme. To prove the theorem, we consider the following hybrid games for $(i,j) \in \{(1,0)\} \cup ([\ell'] \times [\ell'])$. For convenience, we will denote $(i, \ell'+1) := (i+1, 1)$ and $(i, 0) := (i-1, \ell')$.

$\mathsf{Game}^{(i,j)}$: This is the real game except that the challenger encrypts a message

$$\mathsf{M}_1^{(1,1)} \| \mathsf{M}_1^{(1,2)} \| \cdots \| \mathsf{M}_1^{(i,j)} \| \mathsf{M}_0^{(i,j+1)} \| \cdots \| \mathsf{M}_0^{(\ell',\ell')}$$

where $\mathsf{M}_b^{(i,j)}$ denotes the $(i-1)\ell' + j$th bit of M_b for $b \in \{0,1\}$.

It can be seen that $\mathsf{Game}^{(1,0)}$ corresponds to the case of $\mathsf{coin} = 0$ (M_0 is always encrypted) and $\mathsf{Game}^{(\ell',\ell')}$ corresponds to the case of $\mathsf{coin} = 1$ (M_1 is always encrypted). We denote the event that $\mathcal{A}$ outputs 1 in $\mathsf{Game}^{(i,j)}$ be $X^{(i,j)}$. We have

$$\begin{aligned}
\left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| &= \left|\frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 1 | \mathsf{coin} = 1] + \frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 0 | \mathsf{coin} = 0] - \frac{1}{2}\right| \\
&= \left|\frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 1 | \mathsf{coin} = 1] - \frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 1 | \mathsf{coin} = 0]\right| \\
&= \frac{1}{2}\left|\Pr[X^{(1,0)}] - \Pr[X^{(\ell',\ell')}]\right| \\
&= \frac{1}{2}\left|\sum_{(i,j)\in[\ell']\times[\ell']} \Pr[X^{(i,j-1)}] - \Pr[X^{(i,j)}]\right|
\end{aligned}$$

$$\leq \frac{1}{2} \sum_{(i,j)\in[\ell']\times[\ell']} \left| \Pr[X^{(i,j-1)}] - \Pr[X^{(i,j)}] \right| .$$

where the third equality follows from the definition of $X^{(i,j)}$ and the fourth equation follows from our definition $\mathsf{Game}^{(i,0)} = \mathsf{Game}^{(i-1,\ell')}$. Therefore, to prove the theorem, it suffices to show that $|\Pr[X^{(i,j-1)}] - \Pr[X^{(i,j)}]|$ is negligible for all $(i,j) \in [\ell'] \times [\ell']$.

Lemma 4.14. *For any $i^\star, j^\star \in [\ell']$, there exists PPT adversary $\mathcal{B}$ whose advantage against the adaptive security of the single-bit scheme with redundant key is at least $|\Pr[X^{(i^\star,j^\star-1)}] - \Pr[X^{(i^\star,j^\star)}]|/2$.*

Proof. Suppose an adversary $\mathcal{A}$ that has non-negligible advantage in distinguishing $\mathsf{Game}^{(i^\star,j^\star-1)}$ and $\mathsf{Game}^{(i^\star,j^\star)}$. We use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ against the variant of the single-bit scheme, which proceeds as follows.

Setup. At the beginning of the game, $\mathcal{B}$ is given the master public key $\mathsf{mpk}' = (g, W_0, \{W_{i,j}\}_{[i,j]\in[2]\times[\ell]}, g^\alpha, h^\beta, \{h^{w_{2,i}}\}_{i\in[\ell]}, \{h^{w_{1,i}w_{2,j}}\}_{(i,j)\in[\ell]\times[\ell]}, \mathsf{rand})$ for the single bit scheme. Then, $\mathcal{B}$ picks $\tilde{\alpha}_i \xleftarrow{\$} \mathbb{Z}_p$ for $i \in [\ell']\backslash\{i^\star\}$ and $\tilde{\beta}_j \xleftarrow{\$} \mathbb{Z}_p$ for $j \in [\ell']\backslash\{j^\star\}$ and sets

$$g^{\alpha_i} = \begin{cases} g^{\tilde{\alpha}_i} & \text{for } i \in [\ell']\backslash\{i^\star\} \\ g^\alpha & \text{for } i = i^\star \end{cases}, \qquad h^{\beta_j} = \begin{cases} h^{\tilde{\beta}_j} & \text{for } j \in [\ell']\backslash\{i^\star\} \\ h^\beta & \text{for } j = j^\star \end{cases}.$$

Note that $\mathcal{B}$ implicitly sets $\alpha_{i^\star} = \alpha$ and $\beta_{j^\star} = \beta$ here. Finally, it gives the master public key of the multi-bit scheme $\mathsf{mpk} = (g, W_0, \{W_{1,i}\}_{i=1}^{\ell}, \{W_{2,i}\}_{i=1}^{\ell}, \{g^{\alpha_i}\}_{i=1}^{\ell'}, \{h^{\beta_i}\}_{i=1}^{\ell'}, \mathsf{rand})$ to $\mathcal{A}$. $\mathcal{B}$ does not give h^{w_0}, $\{h^{w_{2,i}}\}_{i\in[\ell]}$, and $\{h^{w_{1,i}w_{2,j}}\}_{(i,j)\in[\ell]\times[\ell]}$ to $\mathcal{A}$ and keeps them secret.

Phase 1 and Phase 2. When $\mathcal{A}$ makes a key extraction query for ID, $\mathcal{B}$ proceeds as follows. We first observe that $\mathcal{B}$ can compute $h^{\alpha_i\beta_j}$ for all $(i,j) \in ([\ell'] \times [\ell'])\backslash\{(i^\star, j^\star)\}$ as follows:

$$h^{\alpha_i\beta_j} = \begin{cases} h^{\tilde{\alpha}_i\tilde{\beta}_j} & \text{for } i \neq i^\star,\, j \neq j^\star \\ (h^\alpha)^{\tilde{\beta}_j} & \text{for } i = i^\star,\, j \neq j^\star \\ (h^\beta)^{\tilde{\alpha}_i} & \text{for } i \neq i^\star,\, j = j^\star \end{cases}. \tag{4.74}$$

For $(i,j) \in ([\ell']\times[\ell'])\backslash\{(i^\star,j^\star)\}$, $\mathcal{B}$ picks $r^{(i,j)} \xleftarrow{\$} \mathbb{Z}_p$ and computes $\mathsf{sk}^{(i,j)} = (A_1^{(i,j)}, A_2^{(i,j)}, \{B_k^{(i,j)}\}_{k=1}^{\ell})$ as

$$A_1^{(i,j)} = h^{\alpha_i\beta_j} \cdot \left(h^{w_0} \prod_{(i',j')\in S(\mathsf{ID})} h^{w_{1,i'}w_{2,j'}} \right)^{r^{(i,j)}}, \; A_2^{(i,j)} = h^{-r^{(i,j)}}, \; \left\{ B_k^{(i,j)} = (h^{w_{2,k}})^{r^{(i,j)}} \right\}_{k=1}^{\ell}.$$

These can be computed using h^{w_0}, $h^{w_{2,i'}}$, and $h^{w_{1,i'}w_{2,j'}}$. To generate other parts of the private key (i.e., $\mathsf{sk}_{\mathsf{ID}}^{(i^\star,j^\star)}$), $\mathcal{B}$ resort to its challenger. Namely, $\mathcal{B}$ makes key extraction query for ID and obtains $\mathsf{sk}'_{\mathsf{ID}} = (A_1 = h^{\alpha\beta + r\cdot\mathsf{H}(\mathsf{ID})} = h^{\alpha_{i^\star}\beta_{j^\star} + r\cdot\mathsf{H}(\mathsf{ID})}, A_2 = h^{-r}, \{B_k = h^{rw_{2,k}}\}_{k=1}^{\ell})$. Then, it sets

$$\mathsf{sk}_{\mathsf{ID}}^{(i^\star,j^\star)} = \left(A_1^{(i^\star,j^\star)} = A_1, \quad A_2^{(i^\star,j^\star)} = A_2, \quad \{B_k^{(i^\star,j^\star)} = B_k\}_{k=1}^{\ell} \right).$$

Finally, it returns the secret key $\mathsf{sk}_{\mathsf{ID}} = \{\mathsf{sk}_{\mathsf{ID}}^{(i,j)}\}_{(i,j)\in[\ell']\times[\ell']}$.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge identity $\mathsf{ID}^\star$ and messages $\mathsf{M}_0, \mathsf{M}_1 \in \{0,1\}^{\ell_M}$, $\mathcal{B}$ proceeds as follows. It makes a challenge query for its challenger for the identity $\mathsf{ID}^\star$ and messages $(\mathsf{M}_0^{(i^\star,j^\star)}, \mathsf{M}_1^{(i^\star,j^\star)})$, where $\mathsf{M}_b^{(i^\star,j^\star)}$ is the $(i^\star-1)\ell' + j^\star$th bit of M_b. Then, the challenge ciphertext

$$\Big(C_0' = \mathsf{M}_{\mathsf{coin}}^{(i^\star,j^\star)} \oplus \mathsf{GL}\big(e(g,h)^{s\alpha\beta}, \mathsf{rand}\big), \quad C_1' = g^s, \quad C_2', \quad \{D_j'\}_{j=1}^{\ell} \Big)$$

is given to $\mathcal{B}$. $\mathcal{B}$ then computes $\mathsf{K}^{(i,j)} = \mathsf{GL}\big(e(C_1, h^{\alpha_i\beta_j}), \mathsf{rand}\big) = \mathsf{GL}\big(e(g,h)^{s\alpha_i\beta_j}, \mathsf{rand}\big)$ for $(i,j) \in ([\ell']\times[\ell'])\backslash\{(i^\star,j^\star)\}$. This is possible because $h^{\alpha_i\beta_j}$ for $(i,j) \neq (i^\star,j^\star)$ can be efficiently computable as we observed in Eq.(4.74). Finally, $\mathcal{B}$ sets $C_0 \in \{0,1\}^{\ell_M}$ as follows. In the following, $C_0^{(i,j)}$ denotes $(i-1)\ell' + j$th bit of C_0.

$$C_0^{(i,j)} = \begin{cases} \mathsf{K}^{(i,j)} \oplus \mathsf{M}_1^{(i,j)} & \text{for } (i < i^\star) \vee (i = i^\star \wedge j < j^\star) \\ C_0' & \text{for } i = i^\star,\ j = j^\star \\ \mathsf{K}^{(i,j)} \oplus \mathsf{M}_0^{(i,j)} & \text{for } (i > i^\star) \vee (i = i^\star \wedge j > j^\star) \end{cases}.$$

Finally, $\mathcal{B}$ returns the challenge ciphertext $(C_0, C_1, C_2, \{D_j\}_{j=1}^{\ell})$ to $\mathcal{B}$.

Guess. At last, $\mathcal{A}$ outputs $\widehat{\mathsf{coin}}$. Then, $\mathcal{B}$ outputs $\mathsf{coin}' = \widehat{\mathsf{coin}}$.

Analysis. It can be seen that the view of $\mathcal{A}$ corresponds to that in $\mathsf{Game}^{(i^\star,j^\star-1)}$ if $\mathsf{coin} = 0$ and $\mathsf{Game}^{(i^\star,j^\star)}$ if $\mathsf{coin} = 1$. Therefore, $\mathcal{B}$'s advantage is

$$\begin{aligned} \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| &= \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| \\ &= \left|\frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 1|\mathsf{coin} = 1] + \frac{1}{2}\Pr[\widehat{\mathsf{coin}} = 0|\mathsf{coin} = 0] - \frac{1}{2}\right| \\ &= \frac{1}{2}\left|\Pr[\widehat{\mathsf{coin}} = 1|\mathsf{coin} = 1] - \Pr[\widehat{\mathsf{coin}} = 1|\mathsf{coin} = 0]\right| \\ &= \frac{1}{2}\left|\Pr[X^{(i^\star,j^\star-1)}] - \Pr[X^{(i^\star,j^\star)}]\right| \end{aligned}$$

as desired. This completes the proof of Lemma 4.14. □

This completes the proof of Theorem 4.4. □

4.6 Comparisons and Discussions

In this section, we compare our IBE schemes obtained in Section 4.4 and 4.5 with previous schemes. Throughout this section, $|\mathsf{mpk}|$, $|C|$, and $|\mathsf{sk}_{\mathsf{ID}}|$ denote the sizes of the master public keys, ciphertexts, and private keys, respectively. We denote by κ the length of the identity, which corresponds to the output length of the collision resistant hash if we choose to hash the bit string representing an identity.

Ideal Lattice Based IBE. In Section 4.4. we proposed a new ideal lattice based IBE scheme. By changing the base b of the $\boldsymbol{g}_b$-trapdoor, we obtain two types of instantiation offering tradeoffs. Namely, by setting $b = 2$ we obtain the Type 1 IBE scheme and by setting $b = n^{\frac{1}{4}}$ we obtain the Type 2 IBE scheme presented in Section 4.4.1. The Type 2 IBE allows for a more compact size parameters compared to the Type 1 IBE, whereas the Type 1 IBE allows for a more efficient sampling procedure due to the smaller Gaussian width. Note that the technique of changing the base b is applicable for other existing IBE schemes as well, offering a similar tradeoff presented above. Both of our schemes achieve the best efficiency among existing adaptively secure IBE schemes assuming the fixed polynomial approximation of the RLWE problem. This is illustrated in Table 4.1. We point out that the largest improvement from the Yamada's IBE is that we greatly weakened the underlying hardness assumption while improving the overall efficiency of the scheme.

Figure 4.1: Comparison of Lattice-Base IBEs in the Standard Model.

Schemes	$\lvert \mathsf{mpk} \rvert$	$\lvert C \rvert$, $\lvert \mathsf{sk}_{\mathsf{ID}} \rvert$	$1/\alpha$ for LWE Assumption	Anonymous?
[CHKP10]	$O(n\kappa \log^2 n)$	$O(n\kappa \log^2 n)$	Fixed $\mathsf{poly}(n)$	Yes
[ABB10]+[Boy10]*	$O(n\kappa \log^2 n)$	$O(n \log^2 n)$	Fixed $\mathsf{poly}(n)$	Yes
[Yam16]: Scheme 1	$O(n\kappa^{\frac{1}{d}} \log^4 n)$	$O(n \log^4 n)$	$n^{\omega(1)}$	Yes
[Yam16]: Scheme 2	$O(n\kappa^{\frac{1}{d}} \log^4 n)$	$O(n \log^4 n)$	All $\mathsf{poly}(n)$	No
Ours: Section 4.4. Type 1.	$O(n\kappa^{\frac{1}{d}} \log^2 n)$	$O(n \log^2 n)$	Fixed $\mathsf{poly}(n)$	Yes
Ours: Section 4.4. Type 2.	$O(n\kappa^{\frac{1}{d}} \log n)$	$O(n \log n)$	Fixed $\mathsf{poly}(n)$	Yes

All parameters presented in the table are obtained by instantiating the schemes in the ring setting. $d \in \mathbb{N}$ is a flexible constant, which can be set to be any value. "$1/\alpha$" for LWE assumption refers to the underlying LWE assumption used in the security reduction. "Fixed $\mathsf{poly}(n)$" means that the corresponding scheme is proven secure under the LWE assumption with $1/\alpha$ being some fixed polynomial (e.g., n^3). "All $\mathsf{poly}(n)$" mean that we have to assume the LWE assumption for all polynomial.

* In the security proof for the adaptively secure variant of IBE in [ABB10], we have a restriction that $q > Q$. Namely, only bounded form of the security is proven. This restriction is removed in the refined analysis due to Boyen [Boy10].

Bilinear Map Based IBE. Here, we compare our scheme in Section 4.5 with other adaptively secure IBE schemes based on the hardness of computational/search problems on bilinear maps in the standard model. To base the security of IBE schemes on such problems, we have to mask the message using the Goldreich-Levin hardcore bit [GL89]. To the best of our knowledge, there are only two IBE schemes that we can apply this modification: Waters IBE [Wat05] and Naccache IBE [Nac07]. As shown in Table 4.2, our scheme achieves asymptotically shorter master public key size than these schemes. We note that to compare the efficiency, we count the number of group elements. However our method comes at the cost of increasing the ciphertext and private key size and we further have to rely on a stronger assumption than theirs.

Figure 4.2: Comparison of IBE from Bilinear Maps in the Standard Model.

Schemes	$\lvert \mathsf{mpk} \rvert$	$\lvert C \rvert$, $\lvert \mathsf{sk}_{\mathsf{ID}} \rvert$	Assumption
[Wat05] + Hardcore bit [GL89]	$O(\kappa)$	2	CBDH
[Nac07] + Hardcore bit [GL89]	$O(\kappa/\log(\lambda)) = O(\kappa/\log(\kappa))$	2	CBDH
Ours: Section 4.5	$O(\sqrt{\kappa})$	$O(\sqrt{\kappa})$	3-CBDHE

Chapter 5

Encoding Predicates by Arithmetic Circuits and Their Applications

5.1 Introduction

A *predicate* is a function $P : \mathcal{X} \to \{0, 1\}$ that partitions an input domain $\mathcal{X}$ into two distinct sets according to some relation. Due to its natural compatibility with cryptographic primitives, predicates have been used in many scenarios to control the disclosure of secrets. This may either come up explicitly during construction (e.g., attribute-based encryptions [SW05, GPSW06], predicate encryptions [BW07, SBC+07, KSW08]) or implicitly during security proofs (e.g., in the form of programmable hashes [HK08, ZCZ16], admissible hashes [BB04a, CHKP10]). However, how to express predicates as (arithmetic) circuits is usually not given much attention in these works. Since the way we embed predicates into a cryptographic primitive has a direct effect on the concrete efficiency of the schemes, it is important to know how efficiently we can embed predicates. In this chapter, we propose an efficient encoding for a specific class of predicates and focus on two primitives that benefit from this: verifiable random functions (VRFs) and predicate encryptions (PE) schemes.

Verifiable Random Functions. VRFs introduced by Micali, Rabin and Vadhan [MRV99] are a special form of pseudorandom functions (PRFs), which additionally enables a secret key holder to create a non-interactive and publicly verifiable proof that validates the output value. An attractive property for the VRF to have is the notion of *all the desired properties* coined by [HJ16], which captures the following features: an exponential-sized input space, adaptive pseudorandomness, and security under a non-interactive complexity assumption.

There currently exist two approaches for constructing VRFs with all the desired properties. The first approach is to use a specific number theory setting (mainly bilinear groups) to handcraft VRFs [HW10, BMR10, ACF14, Jag15, HJ16, Yam17], and the second approach is to use a more generic approach and build VRFs from general cryptographic primitives [GHKW17, Bit17, BGJS17]. While the second approach provides us with better insight on VRFs and allows us to base security on hardness assumptions other than bilinear map based ones, the major drawback is the need for large verification key / proof sizes or the need for strong hardness assumptions such as the subexponential Learning with Errors (LWE) assumption to instantiate the underlying primitives. Concretely, all generic approaches require general non-interactive witness indistin-

[0]The contents of this chapter is based on the work presented at Asiacrypt 2017 under the title "On the Untapped Potential of Encoding Predicates by Arithmetic Circuits and Their Applications" [Kat17].

guishable proofs (NIWIs) and constrained PRFs for admissible hash friendly functions, which we currently do not know how to simultaneously construct compactly and base security under a weak hardness assumption.

The first approach is more successful overall in light of compactness and the required hardness assumptions, however, they come with their own shortcomings. Notably, [Yam17] presents three constructions where only $\omega(\log \lambda)$ group elements[1] are required for either the verification key or the proof. In particular, in one of their schemes, only sub-linear group elements are required for both verification key and proof. However, all three schemes require an L-DDH[2] assumption where $L = \tilde{\Omega}(\lambda)$. In contrast, [Jag15] presents a scheme secure under a much weaker L-DDH assumption where $L = O(\log \lambda)$ and [HJ16] under the DLIN assumption. However, these approaches require a linear number of group elements in the verification key and proof in the security parameter. Therefore, we currently do not know how to construct VRFs that are both compact and secure under a weak hardness assumption.

Predicate Encryption. A predicate encryption (PE) scheme [BW07, SBC$^+$07, KSW08] is a paradigm for public-key encryption that supports searching on encrypted data. In predicate encryption, ciphertexts are associated with some attribute X, secret keys are associated with some predicate P, and the decryption is successful if and only if $P(X) = 1$. The major difficulty of constructing predicate encryption schemes stems from the security requirement that enforces the privacy of the attribute X and the plaintext even amidst multiple secret key queries.

Some of the motivating applications for predicate encryption schemes that are often stated in the literatures are: inspection of recorded log files for network intrusions, credit card fraud investigation and conditional disclosure of patient records. Notably, all the above applications only require checking whether a subset or range conjunction predicate is satisfied. (For a more thorough discussion, see [BW07, SBC$^+$07, KSW08].) Therefore, in some sense many of the applications that motivates for predicate encryption schemes can be implemented by predicate encryption schemes for the class of predicates that are expressive enough to support subset or range conjunctions.

On the surface, the present situation on lattice-based predicate encryption schemes seem bright. We have concrete constructions based on LWE for the class of predicates that supports equality [ABB10, CHKP10], inner-products [AFV11], multi-dimensional equality ($\mathsf{MultD\text{-}Eq}$)[3] [GMW15], and all circuits [GVW15a, GKW17, WZ17][4]. Therefore, in theory, we can realize all the above applications in a secure manner, since subset or range conjunctions can be efficiently encoded by any predicate as expressive as the $\mathsf{MultD\text{-}Eq}$ predicate, i.e., the works of [GMW15, GVW15a, GKW17, WZ17] are all sufficient for the above applications. However, all of these schemes may be too inefficient to use in real-life applications. Namely, the scheme of [GMW15] highly resembles the bilinear map based construction of [SBC$^+$07] and inherits the same problem; it takes $\Omega(2^D)$ decryption time where D roughly corresponds to the number of set elements specifying the subset predicate or the number of conjunctions used in the range conjunction predicate. Further, the schemes of [GVW15a, GKW17, WZ17] are powerful and elegant, albeit they all require subexponential LWE assumptions. Therefore, aiming at predicate encryption schemes with the above applications in mind, we currently do not have satisfactorily efficient

[1]Here, $\omega(f(\lambda))$ denotes any function that grows asymptotically faster than $f(\lambda)$, e.g., $\log^2 \lambda = \omega(\log \lambda)$

[2] The L-DDH problem is where we are given $(h, g, g^{\alpha}, \cdots, g^{\alpha^L}, \Psi)$ and have to decided whether $\Psi = e(g,h)^{1/\alpha}$ or a uniform random element.

[3] The precise definition and discussions of this predicate are given in Section 5.4.2. For the time being, it is enough to view it as a subset predicate.

[4] [GKW17, WZ17] give a generic conversion from ABEs to PEs that uses an obfuscation for a specific program proven secure under the subexponential LWE assumption. Therefore, we have provably secure lattice-based PEs for all circuits using the lattice-based ABE of [GVW13, BGG$^+$14a].

lattice-based schemes. In particular, we do not know how to construct efficient lattice-based PE schemes for the class of $\mathsf{MultD\text{-}Eq}$ predicates. This is in sharp contrast with the bilinear map setting where we know how to obtain efficient schemes for the above applications [BW07].

5.1.1 Our Contributions

In this chapter, we provide two results: a compact VRF under a weak assumption and an efficient lattice-based PE scheme for the class of $\mathsf{MultD\text{-}Eq}$ predicates. For the time being, it suffices to think of the $\mathsf{MultD\text{-}Eq}$ predicate as simply a predicate that supports the subset predicate. Here, although the two results may seem independent, they are in fact related by a common theme that they both implicitly or explicitly embed the subset predicates in their constructions.

Our idea is simple. We first detach predicates from cryptographic constructions, and view predicates simply as a function. Then, we introduce the notion of *predicate encoding schemes*[5], where we encode predicates as simple (arithmetic) circuits that have different properties fit for the underlying cryptographic applications. For example, we might not care that a predicate P outputs 0 or 1. We may only care that P behaves differently on satisfied/non-satisfied inputs, e.g., P outputs a value in S_0 when it is satisfied and S_1 otherwise, where S_0, S_1 are disjoint sets. In particular, we provide two predicate encoding schemes $\mathsf{PES}_{\mathsf{FP}}$ and $\mathsf{PES}_{\mathsf{Lin}}$ with different properties encoding the $\mathsf{MultD\text{-}Eq}$ predicates. Then, based on these encoded $\mathsf{MultD\text{-}Eq}$ predicates, we construct our VRFs, and PE schemes for the class of $\mathsf{MultD\text{-}Eq}$ predicates. The following is a summary of our two results.

VRF. We propose two VRFs with all the desired properties. A detailed comparison is provided in Table 5.1. Note that we intentionally excluded the recent VRF constructions of [Bit17, BGJS17, GHKW17] from the table, since their schemes cannot be instantiated efficiently due to the lack of efficient (general) NIWIs and constrained PRFs.

Our constructions are inspired by the bilinear map based VRFs of [Yam17], where they noticed that an admissible hash function [BB04b, CHKP10] can be represented much more compactly by using a subset predicate[6]. We improve their works by further noticing that subset predicates, when viewed as simply a function, can be encoded in various ways into a circuit. In particular, we propose a more efficient circuit encoding ($\mathsf{PES}_{\mathsf{FP}}$) of the subset predicates that is compatible with the underlying algebraic structure of the VRF. We note that at the technical level the constructions are quite different; [Yam17] uses the inversion-based techniques [DY05, BMR10] whereas we do not. Here, simply using $\mathsf{PES}_{\mathsf{FP}}$ already provides us with an improvement over previous schemes, however, by exploiting a special linear structure in $\mathsf{PES}_{\mathsf{FP}}$, we can further improve the efficiency using an idea native to our scheme. Namely, we can skip some of the verification steps required to check the validity of the proof, hence, lowering the number of group elements in the verification key. Our schemes can be viewed as combining the best of [Jag15] and [Yam17]. In the following, to compare the efficiency, we count the number of group elements of the verification key and proof.

[5] We note that the term "predicate encoding" has already been used in a completely different context by [Wee14]. See the section of related work for the differences.

[6] In particular, our idea is inspired by the VRFs based on the admissible hash function of [Yam17], Section 6. However, the construction is more similar to the VRF based on the variant of Water's hash in their Appendix C.

Figure 5.1: Comparison of VRFs with all the desired properties.

Schemes	$\lvert\mathsf{vk}\rvert$ (# of $\mathbb{G}$)	$\lvert\mathsf{sk}\rvert$ (# of $\mathbb{Z}_p$)	$\lvert\pi\rvert$ (# of $\mathbb{G}$)	Assumption	Reduction Cost
[BMR10]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$-DDH	$O(\epsilon/\lambda)$
[HW10]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\lambda Q/\epsilon)$-DDHE	$O(\epsilon^2/\lambda Q)$
[ACF14]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$
[Jag15]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	$O(\log(Q/\epsilon))$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$
[HJ16]	$O(\lambda)$	$O(\lambda)$	$O(\lambda)$	DLIN	$O(\epsilon^{\nu+1}/\lambda Q^\nu)^*$
[Yam17]: Section 7.1.	$\omega(\lambda\log\lambda)$	$\omega(\log\lambda)$	$\omega(\log\lambda)$	$\omega(\lambda\log\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$
[Yam17]: Section 7.3.	$\omega(\log\lambda)$	$\omega(\log\lambda)$	$\omega(\sqrt{\lambda}\log\lambda)$	$\omega(\lambda\log\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$
[Yam17]: App. C.	$\omega(\log\lambda)$	$\omega(\log\lambda)$	$\mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$-DDH	$O(\epsilon^2/\lambda^2 Q)$
Ours: Section 5.5.2.	$\omega(\log^2\lambda)$	$\omega(\log^2\lambda)$	$\omega(\lambda\log^2\lambda)$	$\omega(\log^2\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$
Ours: Section 5.5.4.	$\omega(\sqrt{\lambda}\log\lambda)$	$\omega(\log^2\lambda)$	$\omega(\log\lambda)$	$\omega(\log^2\lambda)$-DDH	$O(\epsilon^{\nu+1}/Q^\nu)^*$

To measure the verification key size $\lvert\mathsf{vk}\rvert$ and proof size $\lvert\pi\rvert$ (resp. secret key size $\lvert\mathsf{sk}\rvert$), we count the number of group elements in $\mathbb{G}$ (resp. $\mathbb{Z}_p$). Q, ϵ denotes the number of adversarial queries and advantage, respectively. We measure all the reduction cost using the techniques of [BR09]. $\omega(f(\lambda))$ means that it can be taken as any function that grows asymptotically faster than $f(\lambda)$; for simplicity one can instead interpret the above $\omega(f(\lambda))$ terms as $O(\log\lambda \cdot f(\lambda))$. $\mathsf{poly}(\lambda)$ represents a fixed polynomial that does not depend on Q, ϵ.

* ν is a constant satisfying $c = 1 - 2^{-1/\nu}$, where c is the relative distance of the underlying error correcting code $C : \{0,1\}^n \to \{0,1\}^\ell$. We can make ν arbitrary close to 1 by choosing $c < 1/2$ appropriately and setting ℓ large enough. (For further detail, see [Gol08], Appendix E.1)

- In our first scheme, the verification key size is $\omega(\log^2\lambda)$, the proof size is $\omega(\lambda\log^2\lambda)$, and the scheme is proven secure under the L-DDH assumption with $L = \omega(\log^2\lambda)$. This is the first scheme that simultaneously achieves a small verification key size and security under an L-DDH assumption where L is poly-logarithm in the security parameter.

- Our second scheme is a modification of our first VRF with some additional ideas; the verification key size is $\omega(\sqrt{\lambda}\log\lambda)$, the proof size is $\omega(\log\lambda)$, and the scheme is proven secure under the L-DDH assumption with $L = \omega(\log^2\lambda)$. This achieves the smallest verification key and proof size among all the previous schemes while also reducing the underlying L of the L-DDH assumption significantly to poly-logarithm.

PE Schemes for the MultD-Eq Predicates. Based on the predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}}$ for the MultD-Eq predicates, we propose a lattice-based PE scheme for the MultD-Eq predicates. Due to the symmetry of the MultD-Eq predicates, we obtain key-policy and ciphertext-policy predicate encryption schemes for the class of predicates that can be expressed as MultD-Eq, such as subset and range conjunction. A detailed comparison is provided in Table 5.2. Note that we exclude the generic constructions of [GVW15a, GKW17, WZ17] to keep the presentation simple. Although the generic constructions are very powerful and elegant, they all require subexponential LWE even if we restrict the underlying ABE to simple circuit classes and determining the concrete efficiency is not obvious, e.g., in [GKW17] the secret key size depends on the underlying FHE scheme.

Figure 5.2: Comparison of lattice PEs for MultD-Eq predicates (over $\mathbb{Z}_p^{D\times\ell}$).

Schemes	$\lvert\mathsf{mpk}\rvert$ (# of $\mathbb{Z}_q^{n\times m}$)	$\lvert\mathsf{sk}\rvert$ (# of $\mathbb{Z}^{2m}$)	$\lvert\mathsf{ct}\rvert$ (# of $\mathbb{Z}_q^{m}$)	LWE param $1/\alpha$	Dec. Time (# of IP)
[GMW15]	$O(D\ell)$	$O(D\ell)$	$O(D\ell)$	$\tilde{O}(\sqrt{D}\cdot n^{1.5})^\dagger$	$O(\ell^D)$
Ours: Section 5.6.2	$O(D\ell p)$	1	$O(D\ell p)$	$\tilde{O}(\max\{\frac{n^2}{\sqrt{D\ell p}}, \sqrt{D\ell p}\cdot n\})$	1

To compare (space) efficiency, we measure the master public key size $\lvert\mathsf{mpk}\rvert$, secret key size $\lvert\mathsf{sk}\rvert$ and ciphertext size $\lvert\mathsf{ct}\rvert$ by the required number of elements in $\mathbb{Z}_q^{n\times m}, \mathbb{Z}^{2m}, \mathbb{Z}_q^m$, respectively. We measure the decryption time as the number of inner products computed between vectors in $\mathbb{Z}_q^{2m}$.

† To be fair, we provided a more rigorous analysis for their parameter selections (as we did with our scheme).

Our scheme achieves the best efficiency in terms of decryption time and the required modulus size q; recall [GMW15] needs to perform $\Omega(2^D)$ number of inner product operations (between secret key vectors and ciphertext vectors) to decrypt a ciphertext, and [GVW15a, GKW17, WZ17] require subexponential LWE for security. Furthermore, compared with [GMW15], the number of secret keys (i.e., vectors in $\mathbb{Z}^{2m}$) we require are only one, whereas they require at least $O(D)$. Our construction follows very naturally from the predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}}$ for the MultD-Eq predicates, and builds upon the proof techniques of [AFV11, BGG$^+$14a].

Other Applications. We also show how to make the identity-based encryption (IBE) scheme of [Yam17] more efficient by using our predicate encoding scheme for the MultD-Eq predicate. In particular, we are able to lower the approximation factor of the LWE problem from $\tilde{O}(n^{11})$ to $\tilde{O}(n^{5.5})$ (with some additional analysis). Furthermore, we are able to significantly reduce the parallel complexity of the matrix multiplications required during encryption and key generation. Notably, our construction does not rely on the sequential matrix multiplication technique of [GV15] as the IBE scheme of [Yam17]. Finally, we note that the size of the public matrices and ciphertexts are unchanged.

5.1.2 Related Works

The idea of encoding predicates to another form has already been implicitly or explicitly used in other works. The notion of randomized encoding [IK00, AIK04] (not specific to predicates) aims to trade the computation of a "complex" function $f(x)$ for the computation of a "simpler" randomized function $\hat{f}(x;r)$ whose output distribution on an input x encodes the value for $f(x)$. The notion of predicate encoding [Wee14, CGW15] (and also the related notion of pair encoding [Att14, Att16]) has already been used previously, in a completely different context, as a generic framework that abstracts the concept of dual system encryption techniques for bilinear maps, and not as a tool for lowering the circuit complexity of predicates.

5.2 Technical Overview

We now give a brief overview of our technical approaches. A formal treatment is given in the main body. We break our overview in two pieces. First, we give intuition for our notion of predicate encoding schemes PES and illustrate the significance of the MultD-Eq predicates. Then, we overview how the different types of PES schemes for the MultD-Eq predicates can be used to construct VRFs, and PE schemes for the MultD-Eq predicates.

Different Ways of Encoding Predicates. Predicates are often times implicit in cryptographic constructions and in some cases there lies an untapped potential. To highlight this, we recall the observation of [Yam17]. An admissible hash function is one of the central tools used to prove adaptive security (e.g., digital signatures, identity-based encryptions, verifiable random functions). At a high level, during the security proof, it allows the simulator to secretly partition the input space into two disjoint sets, so there is a noticeable probability that the input values submitted by the adversary as challenge queries fall inside the intended sets. Traditionally, the partition made by the admissible hash function is viewed as a bit-fixing predicate; a bit-fixing predicate is specified by a string $K \in \{0, 1, \perp\}^\ell$ where the number of non-$\perp$ symbols are $O(\log \lambda)$, and the input space $\{0, 1\}^\ell$ is partitioned by the rule whether the string $x \in \{0, 1\}^\ell$ matches the string K on all non-$\perp$ symbols.

[Yam17] observed that a bit-fixing predicate can be encoded as a subset predicate; an observation not made since the classical works of [BB04b, CHKP10]. In particular, Yamada observed that K has many meaningless $\perp$ symbols and only has $O(\log \lambda)$ meaningful non-$\perp$ symbols. Under this observation, he managed to encode K into a very small set T_K (e.g., $|\mathsf{T}_K| = O(\log^2 \ell)$) where each element indicates the position of the non-$\perp$ symbols. Now, the partition of the input space is done by checking whether the input includes the set T_K or not. Since admissible hash functions are implicitly embedded in the public parameters, this idea allowed them to significantly reduce the number of public parameters for identity-based encryption (IBE) schemes and the size of the verification key (or the proof size) for VRFs.

We take this observation one step further. A predicate defines a function, but often a function may be represented as a polynomial[7] in various ways depending on what kind of properties we require. This is easiest to explain through an example. Let us continue with the above example of the subset predicate used in [Yam17]: $P_\mathsf{T} : 2^{[2n]} \to \{0, 1\}$, where $P_\mathsf{T}(\mathsf{S}) = 1$ iff $\mathsf{T} \subseteq \mathsf{S}$. Here, assume $|\mathsf{T}| = m$ and all the inputs to P_T have cardinality n. One of the most natural ways to represent the subset predicate as a polynomial is by its boolean circuit representation:

$$\prod_{i=1}^{m} \underbrace{\left(1 - \prod_{j=1}^{n} \left(1 - \underbrace{\prod_{k=1}^{\zeta} \left(1 - (t_{i,k} - s_{j,k})^2\right)}_{\text{is } t_i = s_j?}\right)\right)}_{\text{is } t_i \in \mathsf{S}?} = \begin{cases} 1 & \text{if} \quad \mathsf{T} \subseteq \mathsf{S} \\ 0 & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S} \end{cases}, \tag{5.1}$$

where $\zeta = \lfloor \log 2n \rfloor + 1$, $\mathsf{T} = \{t_i\}_{i \in [m]}, \mathsf{S} = \{s_j\}_{j \in [n]} \subseteq [2n]$ and $t_{i,k}, s_{j,k}$ are the k-th bit of the binary representation of t_i, s_j. Here Eq. (5.1) is the polynomial representation of the boolean logic $\bigwedge_{i \in [m]} \bigvee_{j \in [n]} \bigwedge_{k \in [\zeta]} (t_{i,k} = s_{j,k})$. This is essentially what was used for the lattice-based IBE construction of [Yam17] with very short public parameters. Observe that this polynomial has degree $2mn\zeta$, which is $O(\lambda \log^3 \lambda)$ if we are considering the subset predicate specifying the admissible hash function, where we have $m = O(\log^2 \lambda), n = O(\lambda)$ and $\zeta = O(\log \lambda)$. However, in general, using a high degree polynomial may be undesirable for many reasons, even if it is only of degree linear in the security parameter. For the case of the IBE scheme of [Yam17], due to the highly multiplicative structure, the encryption and key generation algorithms require to rely on a linear number of heavy sequentialized matrix multiplication technique of [GV15]. Therefore, it is a natural question to ask whether we can embed a predicate into a polynomial with lower degree, and in some cases into a linear polynomial.

[7] It might be more precise to state that a predicate is represented by a circuit, however, in this section we adopt the view of polynomials to better convey the intuition.

Indeed, we show that it is possible for the above predicate. Namely, we can do much better by noticing the extra structure of subset predicates; we know there exists at most one $j \in [n]$ that satisfies $t_i = s_j$. Therefore, we can equivalently express Eq. (5.1) as the following polynomial:

$$\prod_{i=1}^{m} \sum_{j=1}^{n} \prod_{k=1}^{\zeta} \left(1 - (t_{i,k} - s_{j,k})^2\right) = \begin{cases} 1 & \text{if} \quad \mathsf{T} \subseteq \mathsf{S} \\ 0 & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S} \end{cases} . \tag{5.2}$$

This polynomial is now down to degree $2m\zeta$. When this subset predicate specifies the admissible hash function, Eq. (5.2) significantly lowers the degree down to $O(\log^3 \lambda)$. Furthermore, if we do not require the output to be exactly 0 or 1, and only care that the predicate behaves differently on satisfied/non-satisfied inputs, we can further lower the degree down to 2ζ. In particular, consider the following polynomial:

$$m - \sum_{i=1}^{m} \sum_{j=1}^{n} \prod_{k=1}^{\zeta} \left(1 - (t_{i,k} - s_{j,k})^2\right) = \begin{cases} 0 & \text{if} \quad \mathsf{T} \subseteq \mathsf{S} \\ \neq 0 & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S} \end{cases} , \tag{5.3}$$

which follows from the observation that $|\mathsf{T}| = m$. Since, the output of the polynomial is different for the case $\mathsf{T} \subseteq \mathsf{S}$ and $\mathsf{T} \not\subseteq \mathsf{S}$, Eq. (5.3) indeed properly encodes the information of the subset predicate. Using this polynomial instead of Eq. (5.1) already allows us to significantly optimize the concrete parameters of the lattice-based IBE of [Yam17]. In fact, by encoding the inputs T, S in a different way and with some additional ideas, we can encode the subset predicate into a *linear* polynomial.

To summarize, depending on what we require for the encoding of a predicate (e.g., preserve the functionality, linearize the encoding) one has the freedom of choosing how to express a particular predicate. We formalize this idea of a "right encoding" by introducing the notion of *predicate encoding schemes*. In the above we used the subset predicate as an motivating example, however, in our work we focus on a wider class of predicates called the *multi-dimensional equality* $\mathsf{MultD\text{-}Eq}$ predicates, and propose two encoding schemes $\mathsf{PES}_{\mathsf{FP}}$ and $\mathsf{PES}_{\mathsf{Lin}}$ with different applications in mind.

Finally, we state two justifications for why we pursue the construction of predicate encoding schemes for the class of $\mathsf{MultD\text{-}Eq}$ predicates. First, the $\mathsf{MultD\text{-}Eq}$ predicates are expressive enough to encode many useful predicates that come up in cryptography (e.g., bit-fixing, subset conjunction, range conjunction predicates), that being for constructions of cryptographic primitives or for embedding secret information during in the security proof. Second, in spite of its expressiveness, the $\mathsf{MultD\text{-}Eq}$ predicates have a simple structure that we can exploit and offers us plenty of freedom on the types of predicate encoding schemes we can achieve. The definition and a more detailed discussion on the expressiveness of $\mathsf{MultD\text{-}Eq}$ are provided in Section 5.4.2, 5.4.3.

Constructing VRFs. Similarly to many of the prior works [BMR10, ACF14, Jag15, Yam17] on VRFs with all the desired properties, we use admissible hash functions and base security on the L-DDH assumption, which states that given $(h, g, g^{\alpha}, \cdots, g^{\alpha^L}, \Psi)$ it is hard to distinguish whether $\Psi = e(g, h)^{1/\alpha}$ or a random element. Here, we briefly review the core idea used during the security proof of [Yam17] for the pseudorandomness property of the VRF. We note that many of the arguments made below are informal for the sake of intuition. Their observation was that the admissible hash function embedded during simulation can be stated in the following way using a subset predicate:

$$\mathsf{F}_{\mathsf{T}}(X) = \begin{cases} 0 & \text{if } \mathsf{T} \subseteq \mathsf{S}(X) \\ 1 & \text{if } \mathsf{T} \not\subseteq \mathsf{S}(X) \end{cases} \qquad \text{where} \quad \mathsf{S}(X) = \{2i - C(X)_i \mid i \in [n]\}.$$

Here, $C(\cdot)$ is a public hash function that maps an input X (of the VRF) to a bit string $\{0,1\}^n$, and $\mathsf{T} \subseteq [2n]$ is a set defined as $\mathsf{T} = \{2i - K_i \mid i \in [n], K_i \neq \perp\}$ where K is the secret string in $\{0,1,\perp\}^n$ that specifies the partition made by the admissible hash. Since, the number of non-$\perp$ symbols in K are $O(\log^2 \lambda)$, the above function can be represented by a set T with cardinality $O(\log^2 \lambda)$. During security proof, by the property and definition of F_{T}, we have

$$\left(\mathsf{T} \not\subseteq \mathsf{S}(X^{(1)})\right) \wedge \cdots \wedge \left(\mathsf{T} \not\subseteq \mathsf{S}(X^{(Q)})\right) \wedge \left(\mathsf{T} \subseteq \mathsf{S}(X^{*})\right),$$

with non-negligible probability, where X^* is the challenge input and $X^{(1)}, \cdots, X^{(Q)}$ are the inputs for which the adversary has made evaluation queries. The construction of [Yam17] is based on previous inversion-based VRFs [DY05, BMR10]. Here, we ignore the problem of how to add verifiability to the scheme and overview on how they prove pseudorandomness of the VRF evaluation. Informally, during simulation, the simulator uses the following polynomial to encode the admissible hash function:

$$Q(\alpha) \Big/ \left(\prod_{i=1}^{m}\prod_{j=1}^{n}(\alpha + t_i - s_j)\right) = \begin{cases} \frac{\text{const}}{\alpha} + \mathsf{poly}(\alpha) & \text{if} \quad \mathsf{T} \subseteq \mathsf{S}(X) \\ \mathsf{poly}(\alpha) & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S}(X) \end{cases}, \tag{5.4}$$

where $Q(\alpha)$ is some fixed polynomial with degree roughly $4n$ independent of the input X. Here, recall $\alpha \in \mathbb{Z}_p$ is that of the L-DDH problem, and notice that in Eq. (5.4) the polynomial will have α in the denominator if and only if $\mathsf{T} \subseteq \mathsf{S}(X)$. Although this may not seem quite like it, this polynomial is indeed an encoding of the subset predicate[8] since it acts differently depending on $\mathsf{T} \subseteq \mathsf{S}(X)$ and $\mathsf{T} \not\subseteq \mathsf{S}(X)$. Finally, we note that the output Y of the VRF is obtained by simply putting the above polynomial in the exponent of $e(g,h)$.

Now, if the simulator is given enough $(g^{\alpha^i})_{i \in [L]}$ as the L-DDH challenge, it can create a valid evaluation Y for inputs X such that $\mathsf{T} \not\subseteq \mathsf{S}(X)$, since it can compute terms of the form $e(g^{\mathsf{poly}(\alpha)}, h) = e(g,h)^{\mathsf{poly}(\alpha)}$. Furthermore, for the challenge query X^* it will use Ψ; if $\Psi = e(g,h)^{1/\alpha}$ it can correctly simulate for the case $\mathsf{T} \subseteq \mathsf{S}(X^*)$, otherwise the evaluation Y^* of the VRF is independent of X^*. Therefore, under the hardness of the L-DDH assumption, the output is proven pseudorandom. Observe that for the simulator to compute $e(g,h)^{\mathsf{poly}(\alpha)}$ from Eq. (5.4), it needs to have $(g^{\alpha^i})_{i \in [L]}$ where $L = O(n)$. Then, since $n = O(\lambda)$, we need to base this on an L-DDH assumption where $L = O(\lambda)$.[9] To reflect the above polynomial, the verification keys are set as $(h, \hat{g}, (W_i = \hat{g}^{w_i}))$ in the actual construction. During simulation the parameters are (roughly) set as $\hat{g} = g^{Q(\alpha)}$, $\hat{g}^{w_i} = \hat{g}^{\alpha + t_i}$.

The above construction is rather naive in that it checks whether $\mathsf{T} \subseteq \mathsf{S}(X)$ in a brute-force manner (as also noted in [Yam17]). Our idea is to instead use the polynomial from Eq. (5.2) to represent the admissible hash function. In other words, we embed the following polynomial during simulation:

$$\frac{1}{\alpha} \cdot \prod_{i=1}^{m}\sum_{j=1}^{n}\prod_{k=1}^{\zeta}\left(1 - (\alpha + t_{i,k} - s_{j,k})^2\right) = \begin{cases} \frac{1}{\alpha} + \mathsf{poly}(\alpha) & \text{if} \quad \mathsf{T} \subseteq \mathsf{S}(X) \\ \mathsf{poly}(\alpha) & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S}(X) \end{cases}. \tag{5.5}$$

We note that in our actual construction, we use an optimized version of Eq. (5.2) called $\mathsf{PES}_{\mathsf{FP}}$. Similarly to above, we put the above polynomial in the exponent of $e(g,h)$ for the VRF evaluation.

[8] To be strict, this does not exactly fit the definition of predicate encoding we define in Section 5.4. However, we can do so by appropriately arguing the size of α or by viewing α as an indeterminate.

[9] In the actual construction we require $L = \omega(\lambda \log \lambda)$, since we need to simulate a higher degree polynomial in the exponent.

The difference is that the degree of the polynomial in Eq. (5.5) is significantly lowered down to merely $2m\zeta$, which is $O(\log^3 \lambda)$. Therefore, when the simulator needs to compute $e(g,h)^{\mathsf{poly}(\alpha)}$ during simulation, we only require $(g^{\alpha^i})_{i\in[L]}$ for $L = O(\log^3 \lambda)$. Hence, we significantly reduced the required L of the L-DDH assumption to poly-logarithm. Note that we need to validate the output in a different way now, since the terms α, t_i, s_j that appear in the left-hand polynomial are not in the denominator as in Eq. (5.4). Now, to generate the proof, we take the so called "step ladder approach" [Lys02, ACF09, HW10], where we publish values of the form $(g^{\theta_{i'}})_{i'\in[m]}, (g^{\theta_{i,j,k'}})_{(i,j,k')\in[m]\times[n]\times[\zeta]}$ defined as follows:

$$\theta_{i'} = \prod_{i=1}^{i'} \sum_{j=1}^{n} \prod_{k=1}^{\zeta} \left(1 - (w_{i,k} - s_{j,k})^2\right), \quad \theta_{i,j,k'} = \prod_{k=1}^{k'} \left(1 - (w_{i,k} - s_{j,k})^2\right),$$

where we (roughly) set $g^{w_{i,k}}$ as $g^{\alpha + t_{i,k}}$ during simulation. Although this scheme achieves a very short verification key, it comes at the cost of a rather long proof size of $O(mn\zeta) = O(\lambda \log^3 \lambda)$.

Finally, we describe how to make the proof much shorter, while still maintaining a sub-linear verification key size. As a first step, we can use the simple trick used in [Yam17] to make the proof much shorter. Namely, we add helper components to the verification key so that anyone can compute $(\theta_{i,j,k'})$ publicly. However, as in [Yam17], this leads to a long verification key with size $\tilde{\Omega}(\lambda)$. Interestingly, for our construction, we can do much better and shorten the verification key by a quadratic factor by in a sense *skipping* some ladders. The main observation is the additive structure in $(\theta_{i'})_{i'}$. In particular, if each $\theta_{i'}$ were simply a large product $\prod_{i,j,k}\left(1 - (w_{i,k} - s_{j,k})^2\right)$, we would have to prepare all the necessary helper components in the verification key that would allow to compute $g^{\theta_{i,j,\zeta}}$. This is because in the step ladder approach, after computing $g^{\theta_{i,j,\zeta}}$, we have to reuse this as an input to the bilinear map to validate the next term in the ladder. However, in our case, we only need the ability to publicly compute $e(g,g)^{\theta_{i,j,\zeta}}$. Here, we crucially rely on the additive structure in $\theta_{i'}$ that allows us to compute $e(g,g)^{\sum_{j\in[n]}\theta_{i,j,\zeta}}$ by ourselves; thus the notion of skipping some ladders. Note that we are not able to publicly compute $e(g,g)^{\prod_{j\in[n]}\theta_{i,j,\zeta}}$. Finally, we continue with the step ladder approach for the outer $\prod_{i=1}^{i'}$ products. Therefore, since we only need the ability to generate $e(g,g)^{\theta_{i,j,\zeta}}$ rather than $g^{\theta_{i,j,\zeta}}$, we can reduce quadratically the number of helper components we have to publish in the verification key.

Constructing PE for the $\mathsf{MultD\text{-}Eq}$ Predicates. Our proposed predicate encryption scheme for the $\mathsf{MultD\text{-}Eq}$ predicates follows the general framework of [AFV11, BGG+14a], which allows us to compute an inner product of a private attribute vector X associated to a ciphertext and a (public) predicate vector Y associated to a secret key. To accommodate this framework, we use our proposed *linear* predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}}$ for the $\mathsf{MultD\text{-}Eq}$ predicates. In the overview, we continue with our examples with the subset predicate for simplicity. The core idea is the same as for the $\mathsf{MultD\text{-}Eq}$ predicates. Essentially, $\mathsf{PES}_{\mathsf{Lin}}$ will allow us to further modify Eq. (5.3), to the following linear polynomial:

$$\sum_{i=1}^{L} a_i \mathsf{X}_i = \begin{cases} 0 & \text{if} \quad \mathsf{T} \subseteq \mathsf{S} \\ \neq 0 & \text{if} \quad \mathsf{T} \not\subseteq \mathsf{S} \end{cases}, \tag{5.6}$$

where $(\mathsf{X}_i)_{i\in[L]}, (a_i)_{i\in[L]} \in \mathbb{Z}_q^L$ are encodings of the attribute set T and the predicate set S, respectively.

Following the general framework, the secret key for a user with predicate set S is a short vector $\mathbf{e}$ such that $[\mathbf{A}|\mathbf{B}_{\mathsf{S}}]\mathbf{e} = \mathbf{u}$ for a random public vector $\mathbf{u}$, where $\mathbf{B}_{\mathsf{S}}$ is defined as in Eq. (5.7) below.

Furthermore, we privately embed an attribute set T into the ciphertext as

$$[\mathbf{c}_1^\top \mid \cdots \mid \mathbf{c}_L^\top] = \mathbf{s}^\top[\mathbf{B}_1 + \mathsf{X}_1\mathbf{G} \mid \cdots \mid \mathbf{B}_L + \mathsf{X}_L\mathbf{G}] + [\mathbf{z}_1^\top \mid \cdots \mid \mathbf{z}_L^\top].$$

Using the gadget matrix $\mathbf{G}$ of [MP12], a user corresponding to the predicate set S can transform the ciphertext without knowledge of T as follows:

$$\sum_{i=1}^{L} \mathbf{c}_i^\top \mathbf{G}^{-1}(a_i\mathbf{G}) = \mathbf{s}^\top \Big(\underbrace{\sum_{i=1}^{L} \mathbf{B}_i \mathbf{G}^{-1}(a_i\mathbf{G})}_{= \mathbf{B}_{\mathsf{S}}} + \sum_{i=1}^{L} a_i \mathsf{X}_i \cdot \mathbf{G} \Big) + \underbrace{\sum_{i=1}^{L} \mathbf{z}_i^\top \mathbf{G}^{-1}(a_i\mathbf{G})}_{= \mathbf{z} \text{ (noise term)}}. \tag{5.7}$$

Observe the matrix $\mathbf{B}_{\mathsf{S}}$ is defined independently of X (i.e., the attribute set S). By Eq. (5.6) and the correctness of the predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}}$, we have $\sum_{i\in[L]} a_i\mathsf{X}_i = 0$ when the subset predicate is satisfied, as required for decryption. To prove security, we set the matrices $\{\mathbf{B}_i\}_{i\in[L]}$ as $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - \mathsf{X}_i^* \cdot \mathbf{G}$, where $\mathbf{A}$ is from the problem instance of LWE, $\mathbf{R}_i$ is a random matrix with small coefficients and $(\mathsf{X}_i^*)_{i\in[L]}$ is the encoding of the challenge attribute set T^*. During simulation we have

$$\mathbf{B}_{\mathsf{S}} = \mathbf{A}\mathbf{R}_{\mathsf{S}} - \sum_{i=1}^{L} a_i \mathsf{X}^* \mathbf{G}, \quad \text{where} \quad \mathbf{R}_{\mathsf{S}} = \sum_{i=1}^{L} \mathbf{R}_i \mathbf{G}^{-1}(a_i\mathbf{G}).$$

for any set S. Here, we have $\sum_{i\in[L]} a_i\mathsf{X}^* \neq 0$ iff $\mathsf{T}^* \not\subseteq \mathsf{S}$. Therefore, for the key extraction queries for S such that $\mathsf{T}^* \not\subseteq \mathsf{S}$, we can use $\mathbf{R}_{\mathsf{S}}$ as the $\mathbf{G}$-trapdoor [MP12] for the matrix $[\mathbf{A}|\mathbf{B}_{\mathsf{S}}]$ to simulate the secret keys. We are able to generate the challenge ciphertext for the subset T^* by computing

$$\underbrace{(\mathbf{s}^\top\mathbf{A} + \mathbf{z}'^\top)}_{\text{LWE Problem}}[\mathbf{I}|\mathbf{R}_1|\cdots|\mathbf{R}_L] = \mathbf{s}^\top[\mathbf{A}|\mathbf{B}_1 + \mathsf{X}_1^*\mathbf{G}|\cdots|\mathbf{B}_L + \mathsf{X}_L^*\mathbf{G}] + \underbrace{\mathbf{z}'^\top[\mathbf{I}|\mathbf{R}_1|\cdots|\mathbf{R}_L]}_{\text{simulation noise term}}$$

A subtle point here is that the simulation noise term is not distributed correctly as in Eq. (5.7). However, this can be resolved by the noise rerandomization technique of [KY16].[10]

Finally, we propose a technique to finer analyze the growth of the noise term $\mathbf{z} = \sum_{i\in[L]} \mathbf{z}_i^\top \mathbf{G}^{-1}(a_i\mathbf{G})$ and the $\mathbf{G}$-trapdoor $\mathbf{R}_{\mathsf{S}} = \sum_{i\in[L]} \mathbf{R}_i\mathbf{G}^{-1}(a_i\mathbf{G})$ used during simulation. This allows us to choose narrower Gaussian parameters and let us base security on a weaker LWE assumption. The main observation is that $\mathbf{G}^{-1}(a_i\mathbf{G}) \in \{0,1\}^{nk\times nk}$ is a block-diagonal matrix with n square matrices with size k along its diagonals where $n = O(\lambda)$ and $k = O(\log\lambda)$. Exploiting this additional block-diagonal structure, we are able to finer control the growth of $\|\mathbf{v}\|_2$ and $s_1(\mathbf{R}_{\mathsf{S}})$ (i.e., the largest singular value of $\mathbf{R}_{\mathsf{S}}$).

5.3 Preparation

5.3.1 Verifiable Random Functions

We define a verifiable random function $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify})$ as a tuple of three probabilistic polynomial time algorithms [MRV99].

[10] Alternatively, we could have used the techniques of [AFV11, BGG$^+$14a] and altered the real scheme by multiplying the error vectors of the ciphertexts by random matrices with small coefficients.

$\mathsf{Gen}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$: The key generation algorithm takes as input the security parameter 1^λ and outputs a verification key vk and a secret key sk.

$\mathsf{Eval}(\mathsf{sk}, X) \to (Y, \pi)$: The evaluation algorithm takes as input the secret key sk and an input $X \in \{0,1\}^n$, and outputs a value $Y \in \mathcal{Y}$ and a proof π, where $\mathcal{Y}$ is some finite set.

$\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi)) \to 0/1$: The verification algorithm takes as input the verification key vk, $X \in \{0,1\}^n$, $Y \in \mathcal{Y}$ and a proof π, and outputs a bit.

Definition 5.1. *We say a tuple of polynomial time algorithms* $\mathsf{VRF} = (\mathsf{Gen}, \mathsf{Eval}, \mathsf{Verify})$ *is a verifiable random function if all of the following requirements hold:*

Correctness. *For all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ *and all* $X \in \{0,1\}^n$*, if* $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$ *then* $\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi))$*.*

Uniqueness. *For an arbitrary string* $\mathsf{vk} \in \{0,1\}^*$ *(not necessarily generated by* Gen*) and all* $X \in \{0,1\}^n$*, there exists at most a single* $Y \in \mathcal{Y}$ *for which there exists an accepting proof* π*.*

Pseudorandomness. *This security notion is defined by the following game between a challenger and an adversary* $\mathcal{A}$*.*

Setup. *The challenger runs* $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Gen}(1^\lambda)$ *and gives* vk *to* $\mathcal{A}$*.*

Phase 1. $\mathcal{A}$ *adaptively submits an evaluation query* $X \in \{0,1\}^n$ *to the challenger, and the challenger returns* $(Y, \pi) \leftarrow \mathsf{Eval}(\mathsf{sk}, X)$*.*

Challenge Query. *At any point,* $\mathcal{A}$ *may submit a challenge input* $X^* \in \{0,1\}^n$*. Here, we require that* $\mathcal{A}$ *has not submitted* X^* *as an evaluation query in Phase 1. The challenger picks a random coin* $\mathsf{coin} \leftarrow \{0,1\}$*. Then it runs* $(Y_0^*, \pi_0^*) \leftarrow \mathsf{Eval}(\mathsf{sk}, X^*)$ *and picks* $Y_1^* \leftarrow \mathcal{Y}$*. Finally it returns* Y_{coin}^* *to* $\mathcal{A}$*.*

Phase 2. $\mathcal{A}$ *may continue on submitting evaluation queries as in Phase 1 with the added restriction that* $X \neq X^*$*.*

Guess. *Finally,* $\mathcal{A}$ *outputs a guess* $\widehat{\mathsf{coin}}$ *for* coin*.*

The advantage of $\mathcal{A}$ *is defined as* $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}|$*. We say that the* VRF *satisfies (adaptive) pseudorandomness if the advantage of any probabilistic polynomial time algorithm* $\mathcal{A}$ *is negligible.*

5.3.2 Predicate Encryptions

We present the definition of predicate encryption [BW07, KSW08, AFV11]. A predicate encryption PE scheme with attribute space $\mathcal{X}$ and predicate space $\mathcal{P}$ consists of four probabilistic polynomial time algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$.

$\mathsf{Setup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input a security parameter 1^λ and outputs a master public key mpk and a master secret key msk.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, P) \to \mathsf{sk}_P$: The key generation algorithm takes as input the master public key mpk, the master secret key msk, and a predicate $P \in \mathcal{P}$. It outputs a secret key sk_P. We assume the description of P is implicitly included in sk_P.

$\mathsf{Encrypt}(\mathsf{mpk}, X, \mathsf{M}) \to \mathsf{ct}$: The encryption algorithm takes as input a master public key mpk, an attribute vector $X \in \mathcal{X}$ and a message M. It outputs a ciphertext ct.

$\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_P, \mathsf{ct}) \to \mathsf{M}$ or $\perp$: The decryption algorithm takes as input the master public key mpk, a secret key sk_P, and a ciphertext ct. It outputs the message M or $\perp$, which means that the ciphertext is not in a valid form.

Definition 5.2. *We say a tuple of algorithms* $\mathsf{PE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Encrypt}, \mathsf{Decrypt})$ *is a predicate encryption scheme if all of the following requirements hold:*

Correctness. *For all* $\lambda \in \mathbb{N}$*, all* $X \in \mathcal{X}, P \in \mathcal{P}$ *such that* $P(X) = 1$[11] *and all* M *in the specified message space,* $\Pr[\mathsf{Decrypt}(\mathsf{mpk}, \mathsf{sk}_P, \mathsf{Encrypt}(\mathsf{mpk}, X, \mathsf{M})) = \mathsf{M}] = 1 - \mathsf{negl}(\lambda)$ *holds, where the probability is taken over the randomness used in all of the algorithms.*

Security. *This security notion is defined by the following game between a challenger and an adversary* $\mathcal{A}$*.*

> **Setup.** *At the outset of the game,* $\mathcal{A}$ *submits to the challenger an attribute* $X^* \in \mathcal{X}$ *on which it wishes to be challenged. Then, the challenger runs* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda)$ *and gives the public parameter* mpk *to* $\mathcal{A}$*.*
>
> **Phase 1.** $\mathcal{A}$ *adaptively submits key extraction queries. If* $\mathcal{A}$ *submits a predicate* $P \in \mathcal{P}$ *to the challenger, the challenger returns* $\mathsf{sk}_P \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, P)$*. Here, we require the predicates* P *to satisfy* $P(X^*) = 0$ *(that is,* sk_P *does not decrypt the challenge ciphertext).*
>
> **Challenge Phase.** *At any point,* $\mathcal{A}$ *outputs a message* M^**. The challenger picks a random coin* $\mathsf{coin} \leftarrow \{0,1\}$ *and a random ciphertext* ct_1^* *from the ciphertext space. If* $\mathsf{coin} = 0$*, it runs* $\mathsf{ct}_0^* \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, X^*, \mathsf{M}^*)$ *and gives the challenge ciphertext* ct_0^* *to* $\mathcal{A}$*. If* $\mathsf{coin} = 1$*, it gives* ct_1^* *to* $\mathcal{A}$*.*
>
> **Phase 2.** $\mathcal{A}$ *may continue to make key extraction queries as in Phase 1.*
>
> **Guess.** *Finally,* $\mathcal{A}$ *outputs a guess* $\widehat{\mathsf{coin}}$ *for* coin*.*

The advantage of $\mathcal{A}$ *is defined as* $|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}|$*. We say that the* PE *scheme is* selectively secure *and* weakly attribute hiding*, if the advantage of any PPT* $\mathcal{A}$ *is negligible.*

5.3.3 Background on Bilinear Maps.

Certified Group Generators. We define certified bilinear group generators as introduced in [HJ16]. We require that there is an efficient bilinear group generator algorithm GrpGen that on input 1^λ outputs a description of bilinear groups $\mathbb{G}$, $\mathbb{G}_T$ with prime order p and a map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We also require that GrpGen is certified. Namely, there is an efficient algorithm GrpVfy that on input a (possibly incorrectly generated) description of the bilinear groups and outputs whether the description is valid or not. Furthermore, we require that each group element has unique encoding, which can be efficiently recognized.

Definition 5.3. *A bilinear group generator is a PPT algorithm* GrpGen *that takes as input a security parameter* 1^λ *and outputs* $\Pi = (p, \mathbb{G}, \mathbb{G}_T, \circ, \circ_T, e, \phi(1))$ *such that the following requirements are satisfied.*

1. p *is prime and* $\log(p) = \Omega(\lambda)$*.*

[11] We follow the convention that $P(X) = 1$ signifies the ability to decrypt. This is opposite to the convention used in the recent lattice-based schemes, and is done purely for convenience of our presentation.

2. *$\mathbb{G}$ and $\mathbb{G}_T$ are subsets of $\{0,1\}^*$, defined by the algorithmic descriptions of maps $\phi : \mathbb{Z}_p \to \mathbb{G}$ and $\phi_T : \mathbb{Z}_p \to \mathbb{G}_T$.*

3. *$\circ$ and $\circ_T$ are algorithmic descriptions of efficiently computable (in the security parameter) maps $\circ : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ and $\circ_T : \mathbb{G}_T \times \mathbb{G}_T \to \mathbb{G}_T$, such that*
   - *$(\mathbb{G}, \circ)$ and $(\mathbb{G}_T, \circ_T)$ form algebraic groups*
   - *ϕ is a group isomorphism from $(\mathbb{Z}_p, +)$ to $(\mathbb{G}, \circ)$*
   - *ϕ_T is a group isomorphism from $(\mathbb{Z}_p, +)$ to $(\mathbb{G}_T, \circ_T)$*

4. *e is an algorithmic description of an efficiently computable (in the security parameter) bilinear map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$. We require that e is non-degenerate, that is,*
$$x \neq 0 \quad \Rightarrow \quad e(\phi(x), \phi(x)) \neq \phi_T(0).$$

Definition 5.4. *We say that a group generator* GrpGen *is certified, if there exists a deterministic polynomial time algorithm* GrpVfy *with the following properties.*

1. ***Parameter validation.*** *Given a string Π (which is not necessarily generated by* GrpGen*), algorithm* GrpVfy(Π) *outputs 1 if and only if Π has the form*
$$\Pi = (p, \mathbb{G}, \mathbb{G}_T, \circ, \circ_T, e, \phi(1))$$
and all requirements from Definition 5.3 are satisfied.

2. ***Recognition and unique representation of elements of $\mathbb{G}$.*** *Each element in $\mathbb{G}$ has a unique representation that is efficiently recognizable. Namely, on input two strings Π and s,* GrpVfy(Π, s) *outputs 1 if and only if* GrpVfy$(\Pi) = 1$ *and it holds that $s = \phi(x)$ for some $x \in \mathbb{Z}_p$. Here $\phi : \mathbb{Z}_p \to \mathbb{G}$ denotes the fixed group isomorphism contained in Π to specify the representation of elements of $\mathbb{G}$ (see Definition 5.3).*

Hardness Assumption.

Definition 5.5 (L-Diffie-Hellman Assumption)**.** *For a PPT algorithm $\mathcal{A}$, an advantage for the decisional L-Diffie-Hellman problem L-DDH of $\mathcal{A}$ with respect to* GrpGen *is defined as follows:*

$$\mathsf{Adv}_{\mathcal{A}}^{L\text{-}DDH} = |\Pr[\mathcal{A}(\Pi, g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^L}, \Psi_0) \to 1] - \Pr[\mathcal{A}(\Pi, g, h, g^{\alpha}, g^{\alpha^2}, \cdots, g^{\alpha^L}, \Psi_1) \to 1]|,$$

where $\Pi \leftarrow \mathsf{GrpGen}(1^\lambda), \alpha \leftarrow \mathbb{Z}_p^*, g, h \leftarrow \mathbb{G}, \Psi_0 = e(g,h)^{1/\alpha}$ and $\Psi_1 \leftarrow \mathbb{G}_T$. We say that L-DDH assumption holds if $\mathsf{Adv}_{\mathcal{A}}^{L\text{-}DDH}$ is negligible for all PPT $\mathcal{A}$.

5.3.4 Other Facts.

The following lemma is taken from [KY16], and is implicit in [BR09, Jag15, Yam16].

Lemma 5.1 ([KY16], Lemma 8)**.** *Let us consider a VRF and an adversary $\mathcal{A}$ that breaks pseudorandomness with advantage ϵ. Let the input space be $\mathcal{X}$ and consider a map γ that maps a sequence of inputs to a value in $[0,1]$. We consider the following experiment. We first execute the security game for $\mathcal{A}$. Let X^* be the challenge input and $X_1, \cdots, X_Q$ be the inputs for which evaluation queries were made. We denote $\mathbb{X} = (X^*, X_1, \cdots, X_Q)$. At the end of the game, we set*

$\mathsf{coin}' \in \{0,1\}$ *as* $\mathsf{coin}' = \widehat{\mathsf{coin}}$ *with probability* $\gamma(\mathbb{X})$ *and* $\mathsf{coin}' \leftarrow \{0,1\}$ *with probability* $1-\gamma(\mathbb{X})$. *Then, the following holds.*

$$\left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| \geq \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$

where $\gamma_{\min}$ *(resp.* $\gamma_{\max}$*) is the maximum (resp. minimum) of* $\gamma(\mathbb{X})$ *taken over all possible* $\mathbb{X}$.

As noted in [Yam17], the lemma was originally proven for IBE schemes in [KY16], however, the exact same proof works for VRFs.

5.4 Encoding Predicates with Arithmetic Circuits

Here, we formalize the intuition outlined in the introduction on how to encode predicates as circuits. In doing so, we first define *predicates* and *arithmetic circuits*. Notably, to capture the algebraic properties of circuits, we adapt the view of treating circuits as polynomials and vice versa. (For further details, see [SY10].)

Predicates. A *predicate* is simply a function $P : \mathcal{X} \to \{0,1\}$ over some domain $\mathcal{X}$ with image $\{0,1\}$. In particular, predicate P divides the input space $\mathcal{X}$ into two disjoint sets according to some specified relation. Often times, it will be more meaningful to consider a set of predicates $\mathcal{P} = \{P | P : \mathcal{X} \to \{0,1\}\}$.

Arithmetic Circuits. An *arithmetic circuit* C over a ring $\mathcal{R}$ and a set of variables $X = \{x_1, \cdots, x_n\}$ is a directed acyclic graph as follows, where the vertices of C are called gates. Every gate in C of in-degree 0 (*input gate*) is labelled by either a variable from X or a ring element in $\mathcal{R}$. Every other gate in C is labeled by either $+$ (*addition gate*) of $\times$ (*product gate*) and has in-degree $\geq$2. The unique gate of out-degree 0 is called an *output gate.*[12] The *depth* of C is the length of the longest directed path reaching to the output gate. For two gates u and v in C, if (u,v) is an edge in C, then u is called a child of v, and v is called a parent of u. For a gate v in C, define C_v to be the sub-circuit of C rooted at v.

An arithmetic circuit computes a polynomial in a natural way. For a gate v in C, define $p_v \in \mathcal{R}[X]$ to be the polynomial computed by C_v as follows: If v is an input gate labelled by $\alpha \in \mathcal{R} \cup X$, then $p_v = \alpha$. If v is an addition gate with $v_1, v_2, \cdots, v_k$ as children, then $p_v = \sum_{i \in [k]} p_{v_k}$. If v is a product gate with $v_1, v_2, \cdots, v_k$ as children, then $p_v = \prod_{i \in [k]} p_{v_k}$. For a polynomial $p \in \mathcal{R}[X]$, and a gate v in C, we say that v computes p if $p = C_v$. In particular, we say p is a *polynomial representation* of C when the output gate of C computes p. We define the *degree* of C to be the degree of the maximal-degree monomial of the polynomial representation of C.

Finally, it is clear that given some representation of a polynomial, we can uniquely reconstruct the original arithmetic circuit by iteratively converting each monomials into gates beginning from the most inner monomials and moving outward. Note that since one function may be expressed as a polynomial in number of ways, the reconstructed arithmetic circuit may be different even if it has the same functionality, e.g., although $(x_1 + x_2)^2$ and $x_1^2 + 2x_1x_2 + x_2^2$ have the same functionality, $(x_1 + x_2)^2$ will be of depth 2 consisting of 1 addition gate and 1 product gate, but $x_1^2 + 2x_1x_2 + x_2^2$ will be of depth 2 consisting of 1 addition gate and 3 product gates. In the following work, we will use the terms circuits and polynomials interchangeably.

[12] Here, we only consider arithmetic circuits with a single output.

5.4.1 Predicate Encoding Scheme

We formalize our main tool: predicate encoding scheme.

Definition 5.6 (**Predicate Encoding Scheme**). *Let $\mathcal{P} = \{\mathcal{P}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of set of efficiently computable predicates where $\mathcal{P}_\lambda$ is a set of predicates of the form $P : \mathcal{X}_\lambda \to \{0,1\}$ for some input space $\mathcal{X}_\lambda$, and let $\mathcal{R} = \{\mathcal{R}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of rings. We define a* predicate encoding scheme over a family of rings $\mathcal{R}$ for a family of set of predicates $\mathcal{P}$, *as a tuple of* deterministic *polynomial time algorithms* $\mathsf{PES} = (\mathsf{EncInpt}, \mathsf{EncPred})$ *such that*

- $\mathsf{EncInpt}(1^\lambda, \boldsymbol{x}) \to \hat{\boldsymbol{x}}$: *The input encoding algorithm takes as inputs the security parameter 1^λ and input $\boldsymbol{x} \in \mathcal{X}_\lambda$, and outputs an encoding $\hat{\boldsymbol{x}} \in \{0_{\mathcal{R}_\lambda}, 1_{\mathcal{R}_\lambda}\}^t \subseteq \mathcal{R}_\lambda^t$, where $t = t(\lambda)$ is an integer valued polynomial and $0_{\mathcal{R}_\lambda}, 1_{\mathcal{R}_\lambda}$ denote the zero and identity element of the ring $\mathcal{R}_\lambda$, respectively.*

- $\mathsf{EncPred}(1^\lambda, P) \to \hat{C}$: *The predicate encoding algorithm takes as inputs the security parameter 1^λ and a predicate $P \in \mathcal{P}_\lambda$, and outputs a polynomial representation of an arithmetic circuit $\hat{C} : \mathcal{R}_\lambda^t \to \mathcal{R}_\lambda$. We denote $\hat{\mathcal{C}}_\lambda$ as the set of arithmetic circuits $\{\hat{C} \mid \hat{C} \leftarrow$* $\mathsf{EncPred}(1^\lambda, P), \forall P \in \mathcal{P}_\lambda\}$.

Correctness. *We require a predicate encoding scheme over a family of rings $\mathcal{R}$ for a family of set of predicates $\mathcal{P}$ to satisfy the following: for all $\lambda \in \mathbb{N}$ and all $b \in \{0,1\}$, there exist disjoint subsets $S_{\lambda,0}, S_{\lambda,1} \subset \mathcal{R}_\lambda$ (i.e., $S_{\lambda,0} \cap S_{\lambda,1} = \phi$), such that for all predicates $P \in \mathcal{P}_\lambda$, all inputs $\boldsymbol{x} \in \mathcal{X}_\lambda$ if $P(\boldsymbol{x}) = b$ then $\hat{C}(\hat{\boldsymbol{x}}) \in S_{\lambda,b}$, where $\hat{\boldsymbol{x}} \leftarrow \mathsf{EncInpt}(1^\lambda, \boldsymbol{x}), \hat{C} \leftarrow \mathsf{EncPred}(1^\lambda, P)$.*

Degree. *We say that a predicate encoding scheme* PES *is of degree $d = d(\lambda)$ if the maximal degree of the circuits in $\hat{\mathcal{C}}_\lambda$ (in their polynomial representation) is d. In case $d = 1$, we say* PES *is* linear.

In the following, we will be more loose in our use of notations. For simplicity, we omit the subscripts expressing the domain or the security parameter such as $0_{\mathcal{R}}, S_{\lambda,b}, \mathcal{R}_\ell$ when it is clear from context. We also omit the expression family and simply state that it is a predicate encoding scheme over a ring $\mathcal{R}$ for a set of predicates P. Finally, in the following we assume that the algorithms $\mathsf{EncInpt}(1^\lambda, \cdot), \mathsf{EncPred}(1^\lambda, \cdot)$ will implicitly take the security parameter 1^λ as input and omit it stated otherwise.

The following is an illustrative example showing that the equality predicate $\mathsf{Eq}_{\boldsymbol{y}} : \{0,1\}^\ell \to \{0,1\}$ where $\mathsf{Eq}_{\boldsymbol{y}}(\boldsymbol{x}) = 1$ iff $\boldsymbol{y} = \boldsymbol{x}$ can be encoded in a variety of ways into an arithmetic circuit with different properties.

Example. (Encoding Equality Predicates) Let $\mathcal{P}$ be a set of predicates $\{\mathsf{Eq}_{\boldsymbol{y}} \mid \boldsymbol{y} \in \{0,1\}^\ell\}$ where $\mathsf{Eq}_{\boldsymbol{y}}$ is defined as above, and let the input domain be $\mathcal{X} = \{0,1\}^\ell$ where we denote $\mathcal{X} \ni \boldsymbol{x} = (x_1, \cdots, x_\ell)$. We first consider a predicate encoding scheme PES_1 over the finite field $\mathbb{Z}_2$. Namely, for all $\mathsf{Eq}_{\boldsymbol{y}} \in \mathcal{P}$, let $\mathsf{EncPred}(1^\lambda, \mathsf{Eq}_{\boldsymbol{y}})$ output $\hat{C}_{\boldsymbol{y}} : \mathbb{Z}_2^\ell \to \mathbb{Z}_2$ such that $\hat{C}_{\boldsymbol{y}}(\hat{\boldsymbol{x}}) = \Pi_{i\in[\ell]}\big(1 - \hat{x}_i - \hat{y}_i\big)$ where $\hat{\boldsymbol{x}} = \boldsymbol{x} \in \{0,1\}^\ell$ (resp. $\hat{\boldsymbol{y}} = \boldsymbol{y}$) is the output of $\mathsf{EncInpt}(\boldsymbol{x})$ (resp. $\mathsf{EncInpt}(\boldsymbol{y})$). Recalling $-1 = 1$ over $\mathbb{Z}_2$, it can be checked that we have correctness with $S_0 = \{0\}, S_1 = \{1\}$, and the degree of PES_1 is $d = \ell$. Next, we consider a predicate encoding scheme PES_2 over the ring $\mathbb{Z}_q$ with a much lower degree where $d = 1$. In particular, for all $\mathsf{Eq}_{\boldsymbol{y}} \in \mathcal{P}$ and any integer $q > \ell$, let $\mathsf{EncPred}(\mathsf{Eq}_{\boldsymbol{y}})$ output $\hat{C}_{\boldsymbol{y}} : \mathbb{Z}_q^\ell \to \mathbb{Z}_q$ such that $\hat{C}_{\boldsymbol{y}}(\hat{\boldsymbol{x}}) = \ell - \sum_{i\in[\ell]}\big((1 - \hat{y}_i) + (-1 + 2\hat{y}_i) \cdot \hat{x}_i\big)$ where $\hat{\boldsymbol{x}}, \hat{\boldsymbol{y}} \in \{0,1\}^\ell$ are encoded in the same way as above. Now, observing

$$(1 - \hat{y}_i) + (-1 + 2\hat{y}_i) \cdot \hat{x}_i = \hat{x}_i\hat{y}_i + (1 - \hat{x}_i)(1 - \hat{y}_i), \tag{5.8}$$

it can be checked that $\hat{C}_{\boldsymbol{y}} = 0$ if and only if $\hat{x}_i = \hat{y}_i = 0$ or $\hat{x}_i = \hat{y}_i = 1$, i.e., $\mathsf{Eq}_{\boldsymbol{y}}(\boldsymbol{x}) = 1$. Therefore, we have correctness with $S_0 = \{1, \cdots, \ell\}$, $S_1 = \{0\}$. Furthermore, since $d = 1$, PES_2 is linear. In the following, we continue on to use the left hand form of Eq. (5.8) to express equality.

Remark 5.1 (An alternative notion for the input encoding algorithm)**.** *We remark that an alternative more liberal way of defining the* $\mathsf{EncInpt}$ *algorithm is to allow it to encode the input* $\boldsymbol{x}$ *as any element in* $\mathcal{R}$*, rather than only in* $\{0_{\mathcal{R}}, 1_{\mathcal{R}}\}$*. Then, for example, we may create a trivial* PES *scheme with degree* $d = 1$ *for the equality predicate* $\mathsf{Eq}_{\boldsymbol{y}}$ *by encoding elements* $\boldsymbol{x} \in \{0,1\}^\ell$ *as an element in* $\mathcal{R} = \mathbb{Z}_{2^\ell}$*, and encoding* $\mathsf{Eq}_{\boldsymbol{y}}$ *as an arithmetic circuit* $\hat{C}_{\boldsymbol{y}}(\hat{\boldsymbol{x}}) = \hat{\boldsymbol{x}} - \hat{\boldsymbol{y}}$ *over the ring* $\mathcal{R}$*. However, in this work, we limit ourselves to the stricter notation of simply encoding inputs as 0, 1 elements, since it will provide a more useful and natural compatibility with the algebraic structure of the underlying cryptographic schemes we consider.*

5.4.2 Encoding Multi-Dimensional Equality Predicates

Here, we propose two predicate encoding schemes for the *multi-dimensional equality predicate*[13] ($\mathsf{MultD\text{-}Eq}$) whose constructions are motivated by different applications. As we show later, the multi-dimensional equality predicate is expressive enough to encode many useful predicates that come up in cryptography (e.g., bit-fixing, subset conjunction, range conjunction predicates), that being for constructions of cryptographic primitives or for embedding secret information during in the security proof.

We first define the domains on which the multi-dimensional equality predicates $\mathsf{MultD\text{-}Eq}$ are defined over, and then formally define what they are.

Definition 5.7 (Compatible Domains for $\mathsf{MultD\text{-}Eq}$)**.** *Let* p, D, ℓ *be positive integers. We call a pair of domains* $(\mathcal{X}, \mathcal{Y}) \subseteq \mathbb{Z}_p^{D\times\ell} \times \mathbb{Z}_p^{D\times\ell}$ *to be compatible with the multi-dimensional equality predicates if it satisfies the following:*

For all $\mathsf{X} \in \mathcal{X}, \mathsf{Y} \in \mathcal{Y}$ *and for all* $i \in [D]$*, there exists at most one* $j \in [\ell]$ *such that* $\mathsf{X}_{i,j} = \mathsf{Y}_{i,j}$*, where* $\mathsf{X}_{i,j}$ *and* $\mathsf{Y}_{i,j}$ *denotes the* (i,j)*-th element of* X *and* Y *respectively.*

Definition 5.8 ($\mathsf{MultD\text{-}Eq}$ Predicates)**.** *Let* p, D, ℓ *be positive integers and let* $(\mathcal{X}, \mathcal{Y}) \subseteq \mathbb{Z}_p^{D\times\ell} \times \mathbb{Z}_p^{D\times\ell}$ *be any compatible domains for* $\mathsf{MultD\text{-}Eq}$*. Then, for all* $\mathsf{Y} \in \mathcal{Y}$*, the* multi-dimensional equality predicate $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} : \mathcal{X} \to \{0,1\}$ *is defined as follows:*

$$\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}) = \begin{cases} 1 & \textit{if} \quad \forall i \in [D],\ \exists \textit{unique } j \in [\ell] \textit{ such that } \ \mathsf{X}_{i,j} = \mathsf{Y}_{i,j} \\ 0 & \textit{otherwise} \end{cases},$$

where $\mathsf{X}_{i,j}$ *and* $\mathsf{Y}_{i,j}$ *denotes the* (i,j)*-th element of* X *and* Y *respectively.*

Note that $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X})$ is satisfied only if for each $i \in [D]$, there exists exactly one $j \in [\ell]$ such that $\mathsf{X}_{i,j} = \mathsf{Y}_{i,j}$. Furthermore, since we restrict (X, Y) to be over the compatible domains $(\mathcal{X}, \mathcal{Y})$ for $\mathsf{MultD\text{-}Eq}$, for all $i \in [D]$ we will never have $\mathsf{X}_{i,j} = \mathsf{Y}_{i,j}$ and $\mathsf{X}_{i,j'} = \mathsf{Y}_{i,j'}$ for distinct $j, j' \in [\ell]$. This restriction may appear contrived and inflexible at first, however, this proves to be very useful for constructing predicate encoding schemes with nice qualities, and in fact does not seem to lose much generality in light of expressiveness of the predicate. In particular, by

[13] This predicate is presented in the works of [GMW15] as the $\mathsf{AND\text{-}OR\text{-}EQ}$ predicate satisfying the so called "at most one" promise. We state the conceptual differences between their formalization and ours: they view predicates as functions on both variables X and Y, whereas we view only X as a variable and treat Y as a constant. (Compare [GMW15] Section 3.1 and our Definition 5.8).

appropriately instantiating the compatible domains, we can embed many useful predicates into the $\mathsf{MultD\text{-}Eq}$ predicate. Further discussions are given in the next section.

We now present two types of predicate encoding schemes for the $\mathsf{MultD\text{-}Eq}$ predicate.

Functionality Preserving Encoding Scheme $\mathsf{PES}_{\mathsf{FP}}$. Our first predicate encoding scheme preserves the functionality of the multi-dimensional equality predicate and can be viewed as an efficient polynomial representation of the circuit computing $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}$.

Lemma 5.2. *Let $q = q(\lambda), p = p(\lambda), D = D(\lambda), \ell = \ell(\lambda)$ be positive integers and let $(\mathcal{X}, \mathcal{Y}) \subseteq \mathbb{Z}_p^{D\times\ell} \times \mathbb{Z}_p^{D\times\ell}$ be any compatible domains for the $\mathsf{MultD\text{-}Eq}$ predicate. Further, let $\mathcal{P} = \{\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} : \mathcal{X} \to \{0,1\} \mid \mathsf{Y} \in \mathcal{Y}\}$ be a set of $\mathsf{MultD\text{-}Eq}$ predicates. Then the following algorithms $\mathsf{PES}_{\mathsf{FP}} = (\mathsf{EncInpt}_{\mathsf{FP}}, \mathsf{EncPred}_{\mathsf{FP}})$ is a predicate encoding scheme over the ring $\mathbb{Z}_q$ with degree $d = D\zeta$ where $\zeta = \lfloor \log p \rfloor + 1$:*

- $\mathsf{EncInpt}_{\mathsf{FP}}(\mathsf{X}) \to \hat{\mathsf{X}}$: *It takes as input $\mathsf{X} \in \mathcal{X}$, and outputs an encoding $\hat{\mathsf{X}} \in \{0,1\}^{D\ell\zeta}$ as follows:*

$$\hat{\mathsf{X}} = (\mathsf{X}_{i,j,k})_{(i,j,k)\in[D]\times[\ell]\times[\zeta]},$$

  where $\mathsf{X}_{i,j,k}$ is the k-th bit of the binary representation of the (i,j)-th element of X. Here, the output tuple $(\mathsf{X}_{i,j,k})$ is sorted in the lexicographical order. (See Section 5.3.)

- $\mathsf{EncPred}_{\mathsf{FP}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}) \to \hat{C}_{\mathsf{Y}}$: *It takes as input a predicate $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}$, and outputs the following polynomial representation of an arithmetic circuit $\hat{C}_{\mathsf{Y}} : \mathbb{Z}_q^{D\ell\zeta} \to \mathbb{Z}_q$:*

$$\hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}}) = \prod_{i=1}^{D}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1 - \hat{\mathsf{Y}}_{i,j,k}) + (-1 + 2\hat{\mathsf{Y}}_{i,j,k}) \cdot \hat{\mathsf{X}}_{i,j,k}\Big),$$

  where $\hat{\mathsf{X}}, \hat{\mathsf{Y}} \in \{0,1\}^{D\ell\zeta}$ are encodings of X, Y respectively.

The correctness of $\mathsf{PES}_{\mathsf{FP}}$ holds for the two disjoint subsets $S_0 = \{0\}$, $S_1 = \{1\} \subset \mathbb{Z}_q$.

Proof of Correctness. First, observe that the most inner product equals 1 if $\mathsf{X}_{i,j} = \mathsf{Y}_{i,j}$ and 0 otherwise, due to Eq. (5.8). Here, recall that $\hat{\mathsf{X}}$ is encoded as $(\mathsf{X}_{i,j,k})$ and $\mathsf{X}_{i,j}$ denotes the (i,j)-th element of X. In the following, denote $\mathsf{X}_i, \mathsf{Y}_i$ as the i-th row of X, Y, respectively. Now, since X and Y come from compatible domains of the $\mathsf{MultD\text{-}Eq}$ predicate, for each $i \in [D]$, we have $\mathsf{X}_{i,j} \neq \mathsf{Y}_{i,j}$ for all $j \in [\ell]$ when $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}_i}(\mathsf{X}_i) = 0$. Therefore, we have the following for all $i \in [D]$:

$$\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1 - \hat{\mathsf{Y}}_{i,j,k}) + (-1 + 2\hat{\mathsf{Y}}_{i,j,k}) \cdot \hat{\mathsf{X}}_{i,j,k}\Big) = \begin{cases} 1 & \text{if } \mathsf{MultD\text{-}Eq}_{\mathsf{Y}_i}(\mathsf{X}_i) = 1 \\ 0 & \text{otherwise} \end{cases}. \tag{5.9}$$

Finally, since $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}) = 1$ if and only if $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}_i}(\mathsf{X}_i) = 1$ for all $i \in [D]$, we have $\hat{C}_{\mathsf{Y}}(\mathsf{X}) = b$ when $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}) = b$ for $b \in \{0,1\}$. Thus, we have $S_0 = \{0\}, S_1 = \{1\}$. □

Linear Encoding Scheme $\mathsf{PES}_{\mathsf{Lin}}$. Our second construction is a linear predicate encoding scheme. It achieves linearity by increasing the length of the encoded input $\hat{\mathsf{X}}$ and takes advantage of the fact that we can change the functionality of the encoded arithmetic circuit $\hat{C}$; the output of $\hat{C}$ can be values other than 0 or 1, whereas outputs of predicates are defined to be in $\{0,1\}$.

Lemma 5.3. *Let $q = q(\lambda), p = p(\lambda), D = D(\lambda), \ell = \ell(\lambda)$ be positive integers such that $q > D$ and let $(\mathcal{X}, \mathcal{Y}) \subseteq \mathbb{Z}_p^{D\times\ell} \times \mathbb{Z}_p^{D\times\ell}$ be any compatible domains for the* $\mathsf{MultD\text{-}Eq}$ *predicate. Further, let $\mathcal{P} = \{\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} : \mathcal{X} \to \{0,1\} \mid \mathsf{Y} \in \mathcal{Y}\}$ be a set of* $\mathsf{MultD\text{-}Eq}$ *predicates. Then the following algorithms $\mathsf{PES}_{\mathsf{Lin}} = (\mathsf{EncInpt}_{\mathsf{Lin}}, \mathsf{EncPred}_{\mathsf{Lin}})$ is a predicate encoding scheme over the ring $\mathbb{Z}_q$ with degree $d = 1$, i.e., a linear scheme, where we set $L = 2^\zeta$ and $\zeta = \lfloor \log p \rfloor + 1$ below.*

- $\mathsf{EncInpt}_{\mathsf{Lin}}(\mathsf{X}) \to \hat{\mathsf{X}}$: *It takes as input $\mathsf{X} \in \mathcal{X}$, and outputs an encoding $\hat{\mathsf{X}} \in \{0,1\}^{D\ell L}$ defined as follows:*

$$\hat{\mathsf{X}} = \Big(\prod_{k=1}^{\zeta} (\mathsf{X}_{i,j,k})^{w_k} \Big)_{(i,j,w)\in[D]\times[\ell]\times[L]},$$

  where w_k and $\mathsf{X}_{i,j,k}$ is the k-th bit of the binary representation of $w - 1$[14] and the (i,j)-th element of X, respectively. In case $\mathsf{X}_{i,j,k} = w_k = 0$, we define $(\mathsf{X}_{i,j,k})^{w_k}$ to be 1.

- $\mathsf{EncPred}_{\mathsf{Lin}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}) \to \hat{C}_{\mathsf{Y}}$: *It takes as input a predicate $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}$, and outputs the following polynomial representation of an arithmetic circuit $\hat{C}_{\mathsf{Y}} : \mathbb{Z}_q^{D\ell L} \to \mathbb{Z}_q$:*

$$\hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}}) = D - \sum_{i=1}^{D} \sum_{j=1}^{\ell} \sum_{w=1}^{L} a_{i,j,w} \cdot \hat{\mathsf{X}}_{i,j,w},$$

  where $a_{i,j,w} \in \{-1,0,1\} \subset \mathbb{Z}_q$ is the coefficient for the term $\hat{\mathsf{X}}_{i,j,w} = \prod_{k=1}^{\zeta} (\mathsf{X}_{i,j,k})^{w_k}$ of the polynomial

$$\prod_{k=1}^{\zeta} \Big((1 - \mathsf{Y}_{i,j,k}) + (-1 + 2\mathsf{Y}_{i,j,k}) \cdot \mathsf{X}_{i,j,k} \Big).$$

  Here we treat Y as a constant.

The correctness of $\mathsf{PES}_{\mathsf{Lin}}$ holds for the two disjoint subsets $S_0 = \{1, \cdots, D\}, S_1 = \{0\} \subset \mathbb{Z}_q$.

Proof of Correctness. First, it is easy to check that $a_{i,j,w} \in \{-1,0,1\}$ for all $(i,j,w) \in [D] \times [\ell] \times [L]$, since we have $(1 - \mathsf{Y}_{i,j,k}) \in \{0,1\}$ and $(-1 + 2\mathsf{Y}_{i,j,k}) \in \{-1,1\}$ for any $\mathsf{Y} \in \mathcal{Y}$. The rest of the proof is similar to the previous proof for $\mathsf{PES}_{\mathsf{FP}}$. First, notice that the term $\sum_{j=1}^{\ell} \sum_{w=1}^{L} a_{i,j,w} \cdot \hat{\mathsf{X}}_{i,j,w}$ is the same as the left hand side of Eq. (5.9). Therefore, we have the following for all $i \in [D]$:

$$\sum_{j=1}^{\ell} \sum_{w=1}^{L} a_{i,j,w} \cdot \hat{\mathsf{X}}_{i,j,w} = \begin{cases} 1 & \text{if } \mathsf{MultD\text{-}Eq}_{\mathsf{Y}_i}(\mathsf{X}_i) = 1 \\ 0 & \text{otherwise} \end{cases},$$

where $\mathsf{X}_i, \mathsf{Y}_i$ are the i-th row of X, Y, respectively. Now, since $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}) = 1$ if and only if $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}_i}(\mathsf{X}_i) = 1$ for all $i \in [D]$, we have the following:

$$\sum_{i=1}^{D} \sum_{j=1}^{\ell} \sum_{w=1}^{L} a_{i,j,w} \cdot \hat{\mathsf{X}}_{i,j,w} = \begin{cases} D & \text{if } \mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}) = 1 \\ \in [0, D-1] & \text{otherwise} \end{cases}.$$

Finally, subtracting the above by D and from the fact that $q > D$, we obtain correctness. □

[14] This inconvenient notion is due to the fact that the bit length of p and L may differ by one in case $p = 2^n - 1$ for $n \in \mathbb{N}$.

Remark 5.2. *In some applications, the compatible domains $(\mathcal{X}, \mathcal{Y})$ for* $\mathsf{MultD\text{-}Eq}$ *will have some additional structures that we can exploit to obtain more efficient encoding schemes. For an example, in some case for all $\mathsf{X} \in \mathcal{X}$, all of the rows of X will be equal, i.e., $\mathsf{X}_i = \mathsf{X}_{i'}$ for all $i, i' \in [D]$ where X_i denotes the i-th row of X. In this case, we can reduce the output length of* $\mathsf{EncInpt}$ *by a factor of D by discarding the redundant terms. We will see some concrete examples for our construction of VRF schemes.*

5.4.3 Expressiveness of Multi-Dimensional Equality Predicates

In this section, we will look at the expressiveness of multi-dimensional equality predicates $\mathsf{MultD\text{-}Eq}$. In particular, the following are some predicates that can be expressed as the multi-dimensional equality predicate instantiated with appropriate compatible domains $(\mathcal{X}, \mathcal{Y})$. Combining this with the result of the previous section, we obtain a functionality preserving ($\mathsf{PES_{FP}}$) or a linear ($\mathsf{PES_{Lin}}$) encoding scheme for all the following predicates. Note that the choice of the compatible domains are not unique, and different applications would motivate for different constructions. (For further details, see also [BW07, SBC$^+$07, GMW15].) For completeness, we provide discussions on how to obtain the following predicates from the $\mathsf{MultD\text{-}Eq}$ predicate. Note that the following encoding is only one example; there are possibly many more "efficient" ways of encoding the predicates as $\mathsf{MultD\text{-}Eq}$ predicates, where the meaning of efficient may depend on the application in one's mind.

Bit-fixing predicates. For a vector $\mathbf{v} \in \{0,1,?\}^\ell$ the bit-fixing predicate $P^{\mathsf{BF}}_{\mathbf{v}} : \{0,1\}^\ell \to \{0,1\}$ is defined as

$$P^{\mathsf{BF}}_{\mathbf{v}}(\mathbf{x}) = 1 \iff \bigwedge_{i=1}^{\ell} \Big((\mathbf{v}_i = \mathbf{x}_i) \bigvee (\mathbf{v}_i = ?) \Big).$$

For example this can be built from $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $\mathcal{X}_{\mathsf{BF}}, \mathcal{Y}_{\mathsf{BF}} \subseteq \mathbb{Z}_3^{\ell \times 2}$. This predicate is also known as the *hidden-vector* predicate [BW07].

<u>Construction.</u> Without loss of generality, we set "?" to be 2. Then, map $\mathbf{v} \in \{0,1,2\}^\ell$ and $\mathbf{x} \in \{0,1\}^\ell$ to the following domains $\mathcal{X}_{\mathsf{BF}}, \mathcal{Y}_{\mathsf{BF}} \subseteq \mathbb{Z}_3^{\ell \times 2}$:

$$\mathbf{x} \to \bar{\mathbf{x}} = \begin{bmatrix} \mathbf{x}_1 & 2 \\ \mathbf{x}_2 & 2 \\ \vdots & \vdots \\ \mathbf{x}_\ell & 2 \end{bmatrix} \in \mathcal{X}_{\mathsf{BF}}, \qquad \mathbf{v} \to \bar{\mathbf{v}} = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_1 \\ \mathbf{v}_2 & \mathbf{v}_2 \\ \vdots & \vdots \\ \mathbf{v}_\ell & \mathbf{v}_\ell \end{bmatrix} \in \mathcal{Y}_{\mathsf{BF}}.$$

It is easy to check that $\mathcal{X}_{\mathsf{BF}}, \mathcal{Y}_{\mathsf{BF}} \subseteq \mathbb{Z}_3^{\ell \times 2}$ are indeed compatible with the $\mathsf{MultD\text{-}Eq}$ predicates, i.e., for all $i \in [\ell]$ there exists at most one $j \in [2]$ such that $\bar{\mathbf{x}}_{i,j} = \bar{\mathbf{v}}_{i,j}$. Furthermore, we have $P^{\mathsf{BF}}_{\mathbf{v}}(\mathbf{x}) = 1$ if and only if $\mathsf{MultD\text{-}Eq}_{\bar{\mathbf{v}}}(\bar{\mathbf{x}}) = 1$.

Equality conjunction predicates. For some finite alphabet Σ and a vector $\mathbf{v} \in \Sigma^\ell$ the equality conjunction predicate $P^{\mathsf{EC}}_{\mathbf{v}} : \Sigma^\ell \to \{0,1\}$ is defined as

$$P^{\mathsf{EC}}_{\mathbf{v}}(\mathbf{x}) = 1 \iff \bigwedge_{i=1}^{\ell} (\mathbf{v}_i = \mathbf{x}_i).$$

For example this can be built from $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $\mathcal{X}_{\mathsf{EC}}, \mathcal{Y}_{\mathsf{EC}} \subseteq \mathbb{Z}_{w_1}^{\ell w_2 \times 1}$, where $w_1^{w_2} = |\Sigma|$.

Construction. The most natural way is to simply map Σ^ℓ into $\mathbb{Z}_{|\Sigma|}^\ell$, and define $\mathcal{X}_{\mathsf{EC}} = \mathcal{Y}_{\mathsf{EC}} = \mathbb{Z}_{|\Sigma|}^{\ell\times 1}$. This domain is trivially compatible with the $\mathsf{MultD\text{-}Eq}$ predicate, and we have $P_{\mathbf{v}}^{\mathsf{EC}}(\mathbf{x}) = 1$ if and only if $\mathsf{MultD\text{-}Eq}_{\bar{\mathbf{v}}}(\bar{\mathbf{x}}) = 1$. Similarly, we can also map Σ into $\mathbb{Z}_{w_1}^{w_2}$ where $w_1^{w_2} = |\Sigma|$, and define $\mathcal{X}_{\mathsf{EC}} = \mathcal{Y}_{\mathsf{EC}} = \mathbb{Z}_{w_1}^{\ell w_2 \times 1}$.

Subset conjunction predicates. For some finite alphabet Σ, let $\mathsf{T}_i \in 2^\Sigma$ for $i \in [\ell]$ and set $\vec{\mathsf{T}} = (\mathsf{T}_1, \cdots, \mathsf{T}_\ell)$. Then the subset conjunction predicate $P_{\vec{\mathsf{T}}}^{\mathsf{SC}} : \prod_{i=1}^{\ell} 2^\Sigma \to \{0,1\}$ is defined as

$$P_{\vec{\mathsf{T}}}^{\mathsf{SC}}(\vec{\mathsf{S}}) = 1 \iff \bigwedge_{i=1}^{\ell} (\mathsf{T}_i \subseteq \mathsf{S}_i),$$

where $\vec{\mathsf{S}} = (\mathsf{S}_1, \cdots, \mathsf{S}_\ell)$. For example this can be built from $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $\mathcal{X}_{\mathsf{SC}}, \mathcal{Y}_{\mathsf{SC}} \subseteq \mathbb{Z}_3^{m\times|\Sigma|}$, where $m = \sum_{i=1}^{\ell} |\mathsf{T}_i|$. In particular, when $\ell = 1$, we simply call this predicate as the *subset predicate* $P_{\mathsf{T}}^{\mathsf{SS}} : 2^\Sigma \to \{0,1\}$.

Construction. For simplicity of presentation, we first consider an encoding for the subset predicate $P_{\mathsf{T}}^{\mathsf{Sub}} : 2^\Sigma \to \{0,1\}$ for $\mathsf{T} \in 2^\Sigma$. Further, we assume that all the inputs to $P_{\mathsf{T}}^{\mathsf{Sub}}(\cdot)$ have fixed cardinality of $n \le |\Sigma|$ (as in the case for the subset predicate embedded in the modified admissible hash function. See. Section 5.5.1). Below, let $\mathsf{T} = \{t_1, \cdots, t_m\}$, $\mathsf{S} = \{s_1, \cdots, s_n\}$ where $m = |\mathsf{T}|$, and view elements in Σ as elements in $\mathbb{Z}_{|\Sigma|}$. Then, we can map the sets $\mathsf{T}, \mathsf{S} \in 2^\Sigma$ to matrices in the following domains $\mathcal{X}_{\mathsf{Sub}}, \mathcal{Y}_{\mathsf{Sub}} \subseteq \mathbb{Z}_{|\Sigma|}^{m\times n}$:

$$\mathsf{T} \to \bar{\mathsf{T}} = \begin{bmatrix} t_1 & t_1 & \cdots & t_1 \\ t_2 & t_2 & \cdots & t_2 \\ \vdots & \vdots & \cdots & \vdots \\ t_m & t_m & \cdots & t_m \end{bmatrix} \in \mathcal{X}_{\mathsf{Sub}}, \qquad \mathsf{S} \to \bar{\mathsf{S}} = \begin{bmatrix} s_1 & s_2 & \cdots & s_n \\ s_1 & s_2 & \cdots & s_n \\ \vdots & \vdots & \cdots & \vdots \\ s_1 & s_2 & \cdots & s_n \end{bmatrix} \in \mathcal{Y}_{\mathsf{Sub}}. \tag{5.10}$$

It can be checked that $\mathcal{X}_{\mathsf{Sub}}, \mathcal{Y}_{\mathsf{Sub}} \in \mathbb{Z}_{|\Sigma|}^{m\times n}$ are compatible with the $\mathsf{MultD\text{-}Eq}$ predicates. Furthermore, we have $P_{\mathsf{T}}^{\mathsf{Sub}}(\mathsf{S}) = 1$ if and only if $\mathsf{MultD\text{-}Eq}_{\bar{\mathsf{T}}}(\bar{\mathsf{S}}) = 1$.

To get rid of the restriction that every input to $P_{\mathsf{T}}^{\mathsf{Sub}}(\cdot)$ needs to have cardinality n, we can use the embedding given in [GMW15], Section 3.2, where they map T, S to domains in $\mathbb{Z}_3^{m\times|\Sigma|}$. Finally, to obtain an encoding for the subset conjunction predicate we can simply concatenate the encodings of the subset predicates for each T_i.

Range conjunction predicates. For $T \in \mathbb{N}$ and $\mathbf{a} = (\mathbf{a}_1, \cdots, \mathbf{a}_\ell), \mathbf{b} = (\mathbf{b}_1, \cdots, \mathbf{b}_\ell) \in [T]^\ell$, the comparison conjunction predicate $P_{[\mathbf{a}:\mathbf{b}]}^{\mathsf{RC}} : [T]^\ell \to \{0,1\}$ is defined as

$$P_{[\mathbf{a}:\mathbf{b}]}^{\mathsf{RC}}(\mathbf{x}) = 1 \iff \bigwedge_{i=1}^{\ell} (\mathbf{a}_i \le \mathbf{x}_i \le \mathbf{b}_i).$$

For example this can be built from $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $\mathcal{X}_{\mathsf{RC}}, \mathcal{Y}_{\mathsf{RC}} \subseteq \mathbb{Z}_{T+1}^{\ell\times 2\lceil\log T\rceil}$.

Construction. We can use the tree data structure for storing intervals known as *segment trees* to encode range conjunction predicates as $\mathsf{MultD\text{-}Eq}$ predicates. Since the encoding is classical and rather contrived, we only present the results here, and refer the readers to [BKOS00], Chap.10 and [GMW15], Section 3.3 for further details. In particular, we can encode range conjunction predicates $P_{[\mathbf{a}:\mathbf{b}]}^{\mathsf{RC}} : [T]^\ell \to \{0,1\}$ as $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $\mathcal{X}_{\mathsf{RC}}, \mathcal{Y}_{\mathsf{RC}}$ in $\mathbb{Z}_{T+1}^{\ell\times\lceil\log T\rceil}$.

5.4.4 Exploitable Structures for More Efficient PES Schemes

Here we comment on Remark 5.2. In some cases, the compatible domains $\mathcal{X}, \mathcal{Y}$ for the multidimensional predicates $\mathsf{MultD\text{-}Eq}$ may have additional structures that we can exploit to obtain a more efficient predicate encoding PES scheme. We illustrate this in the following using as example the subset predicate that will be implicit in our VRF construction from Section 5.5.2.

For our VRF construction, we use a special type of subset predicate $P^{\mathsf{Sub}}_{\mathsf{T}} : 2^{\Sigma} \to \{0,1\}$ where the inputs have fixed cardinality of n, as discussed above. We showed in Eq. (5.10) that this particular subset predicate can be encoded as a $\mathsf{MultD\text{-}Eq}$ predicate with compatible domains $(\mathcal{X}_{\mathsf{Sub}}, \mathcal{Y}_{\mathsf{Sub}}) \in \mathbb{Z}^{m\times n}_{|\Sigma|} \times \mathbb{Z}^{m\times n}_{|\Sigma|}$. Therefore, for example, by Lemma 5.2 we can construct a functionality preserving predicate encoding scheme $\mathsf{PES}_{\mathsf{FP}}$ for the the subset predicate with $p = |\Sigma|, D = m, \ell = n$. Namely, for any $\bar{\mathsf{S}} \in \mathcal{X}_{\mathsf{Sub}}$ and $\bar{\mathsf{T}} \in \mathcal{Y}_{\mathsf{Sub}}$ we have

$$\begin{aligned}
\mathsf{EncInpt}_{\mathsf{FP}}(\bar{\mathsf{S}}) &\to \hat{\bar{\mathsf{S}}} = \left(\bar{\mathsf{S}}_{i,j,k}\right)_{(i,j,k)\in[m]\times[n]\times[\zeta]} \\
\mathsf{EncPred}_{\mathsf{FP}}(\mathsf{MultD\text{-}Eq}_{\bar{\mathsf{T}}}) &\to \hat{C}^{\mathsf{Sub}}_{\bar{\mathsf{T}}}, \\
&\quad \text{where} \quad \hat{C}^{\mathsf{Sub}}_{\bar{\mathsf{T}}}(\hat{\bar{\mathsf{S}}}) = \prod_{i=1}^{m}\sum_{j=1}^{n}\prod_{k=1}^{\zeta}\Big((1-\hat{\bar{\mathsf{T}}}_{i,j,k}) + (-1+2\hat{\bar{\mathsf{T}}}_{i,j,k})\cdot\hat{\bar{\mathsf{S}}}_{i,j,k}\Big),
\end{aligned}$$

where $\zeta = \lfloor \log(|\Sigma|)\rfloor + 1$ and $\bar{\mathsf{T}}_{i,j,k}, \bar{\mathsf{S}}_{i,j,k}$ are the k-th bit of the binary representation of $\bar{\mathsf{T}}_{i,j}, \bar{\mathsf{S}}_{i,j}$, respectively. However, as it can be observed from Eq. (5.10), for all $k \in [\zeta]$, we have $\bar{\mathsf{T}}_{i,1,k} = \bar{\mathsf{T}}_{i,j,k}$ for all $j \in [m]$, and $\bar{\mathsf{S}}_{1,j,k} = \bar{\mathsf{S}}_{i,j,k}$ for all $i \in [n]$. Therefore, we can in fact consider a more efficient predicate encoding scheme $\mathsf{PES}'_{\mathsf{FP}}$ that takes advantage of this redundancy:

$$\begin{aligned}
\mathsf{EncInpt}'_{\mathsf{FP}}(\bar{\mathsf{S}}) &\to \hat{\bar{\mathsf{S}}} = \left(\bar{\mathsf{S}}_{1,j,k}\right)_{(j,k)\in[n]\times[\zeta]} \\
\mathsf{EncPred}'_{\mathsf{FP}}(\mathsf{MultD\text{-}Eq}_{\bar{\mathsf{T}}}) &\to \hat{C}^{\mathsf{Sub}}_{\bar{\mathsf{T}}}, \\
&\quad \text{where} \quad \hat{C}^{\mathsf{Sub}}_{\bar{\mathsf{T}}}(\hat{\bar{\mathsf{S}}}) = \prod_{i=1}^{m}\sum_{j=1}^{n}\prod_{k=1}^{\zeta}\Big((1-\hat{\bar{\mathsf{T}}}_{1,j,k}) + (-1+2\hat{\bar{\mathsf{T}}}_{1,j,k})\cdot\hat{\bar{\mathsf{S}}}_{i,1,k}\Big).
\end{aligned}$$

Looking ahead, this encoding scheme (written slightly differently using the symmetry of $\hat{\bar{\mathsf{T}}}_{1,j,k}$ and $\hat{\bar{\mathsf{S}}}_{i,1,k}$) is what we will present in Section 5.5.2, Eq. (5.14). This extra optimization allows us to decrease a factor of m in the output size of $\mathsf{EncInpt}_{\mathsf{FP}}$. Since this idea translates to $\mathsf{PES}_{\mathsf{Lin}}$ as well, in applications such as the predicate encoding scheme we provide in Section 5.6.2, this will directly yield a PE scheme with shorter ciphertexts by a factor of D.

5.5 Verifiable Random Functions

5.5.1 Modified Admissible Hash Functions

To construct our VRF, we use the notion of *partitioning function* as introduced in [Yam17], which is a generalization of the standard admissible hash function [BB04b, CHKP10, FHPS13, Jag15]. At a high level, partitioning functions are similar to programmable hash functions, however, unlike programable hash functions that are defined on specific algebraic structures such as bilinear groups [HK08] and lattices [ZCZ16], partitioning functions are purely informational theoretic primitives.

Definition 5.9 (Partitioning Function). *Let* $\mathsf{F} = \{\mathsf{F}_\lambda : \mathcal{K}_\lambda \times \mathcal{X}_\lambda \to \{0,1\}\}_{\lambda\in\mathbb{N}}$ *be a family of functions. We say that* F *is a* partitioning function*, if there exists a PPT algorithm*

$\mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$, *which takes as input a polynomially bounded function* $Q = Q(\lambda)$ *where* $Q : \mathbb{N} \to \mathbb{N}$ *and a noticeable function* $\epsilon = \epsilon(\lambda)$ *where* $\epsilon : \mathbb{N} \to (0, 1/2]$, *and outputs a* partitioning key K *such that*

1. *There exists* $\lambda_0 \in \mathbb{N}$ *such that*
$$\Pr\left[K \in \mathcal{K}_\lambda : K \leftarrow \mathsf{PrtSmp}\Big(1^\lambda, Q(\lambda), \epsilon(\lambda)\Big)\right] = 1$$
for all $\lambda > \lambda_0$. *Here* λ_0 *may depend on the functions* Q *and* ϵ.

2. *For* $\lambda > \lambda_0$, *there exists functions* $\gamma_{\max}(\lambda)$ *and* $\gamma_{\min}(\lambda)$ *that depend on functions* Q *and* ϵ *such that for* $X^{(1)}, \cdots, X^{(Q(\lambda))}, X^* \in \mathcal{X}_\lambda$ *with* $X^* \notin \{X^{(1)}, \cdots, X^{(Q(\lambda))}\}$,
$$\gamma_{\min}(\lambda) \leq \Pr\left[\mathsf{F}(K, X^{(1)}) = \cdots = \mathsf{F}(K, X^{(Q(\lambda))}) = 1 \wedge \mathsf{F}(K, X^*) = 0\right] \leq \gamma_{\max}(\lambda) \qquad (5.11)$$
holds and the function $\tau(\lambda)$ *defined as*
$$\tau(\lambda) := \gamma_{\min}(\lambda) \cdot \epsilon(\lambda) - \frac{\gamma_{\max}(\lambda) - \gamma_{\min}(\lambda)}{2} \qquad (5.12)$$
is noticeable. The probability is taken over the choice of $K \leftarrow \mathsf{PrtSmp}(1^\lambda, Q(\lambda), \epsilon(\lambda))$.

In this work, we consider the particular partitioning function called the *modified admissible hash function*. This allows us to use the same techniques employed by admissible hash functions, while providing for a more compact representation. The following is obtained by the results of [Jag15] and [Yam17].

Definition 5.10. *(Modified Admissible Hash Function) Let* $n = n(\lambda), \ell = \ell(\lambda)$ *and* $\eta = \eta(\lambda)$ *be an integer-valued function of* λ *such that* $n, \ell = \Theta(\lambda)$ *and* $\eta = \omega(\log \lambda)$, *and* $\{C_n : \{0,1\}^n \to \{0,1\}^\ell\}_{n \in \mathbb{N}}$ *be a family of error correcting codes with minimal distance* $c \cdot \ell$ *for a constant* $c \in (0, 1/2)$. *Let*
$$\mathcal{K}_{\mathsf{MAH}} = \{\mathsf{T} \subseteq [2\ell] \mid |\mathsf{T}| < \eta\} \quad \textit{and} \quad \mathcal{X}_{\mathsf{MAH}} = \{0,1\}^n.$$
Then, we define the modified admissible hash function $\mathsf{F}_{\mathsf{MAH}} : \mathcal{K}_{\mathsf{MAH}} \times \mathcal{X}_{\mathsf{MAH}} \to \{0,1\}$ *as*
$$\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = \begin{cases} 0, & \textit{if } \; \mathsf{T} \subseteq \mathsf{S}(X) \\ 1, & \textit{otherwise} \end{cases} \quad \textit{where} \quad \mathsf{S}(X) = \{2i - C(X)_i \mid i \in [\ell]\}. \qquad (5.13)$$
In the above, $C(X)_i$ *is the i-th bit of* $C(X) \in \{0,1\}^\ell$.

Theorem 5.1. *There exists an efficient algorithm* $\mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon(\lambda))$ *which takes as input a polynomially bounded function* $Q = Q(\lambda)$ *where* $Q : \mathbb{N} \to \mathbb{N}$ *and a noticeable function* $\epsilon = \epsilon(\lambda)$ *where* $\epsilon : \mathbb{N} \to (0, 1/2]$, *and outputs* T *with cardinality exactly* $\eta' = \eta'(\lambda)$, *where*
$$\eta' := \left\lfloor \frac{\log(2Q + Q/\epsilon)}{-\log(1-c)} \right\rfloor,$$
such that Eq. (5.11) *and* (5.12) *hold with respect to* $\mathsf{F} := \mathsf{F}_{\mathsf{MAH}}, \mathsf{PrtSmp} := \mathsf{PrtSmp}_{\mathsf{MAH}}$ *and* $\tau(\lambda) = 2^{-\eta'-1} \cdot \epsilon$. *In particular,* $\mathsf{F}_{\mathsf{MAH}}$ *is a partitioning function.*

5.5.2 Construction

Intuition. In our VRF construction, we implicitly embed the partitioning function $\mathsf{F_{MAH}}$ in the output Y during simulation. In particular, as we mentioned in the technical overview, the strategy is to embed $\mathsf{F_{MAH}}$ in the exponent of g using an L-DDH assumption with the smallest possible L.

Since $\mathsf{F_{MAH}}$ checks whether the subset predicate $P^{\mathsf{Sub}}_{\mathsf{T}} : 2^{[2\ell]} \to \{0,1\}$ is satisfied or not, which is a special case of the subset conjunction predicate presented in Section 5.4.3, it can be encoded as the multi-dimensional equality predicate $\mathsf{MultD\text{-}Eq}$ with (exploitable) compatible domains $\mathcal{X}_{\mathsf{Sub}}, \mathcal{Y}_{\mathsf{Sub}} \subseteq \mathbb{Z}_{2\ell}^{|\mathsf{T}|\times\ell}$ as we showed in Section 5.4.4. For our VRF construction, we consider the functionality preserving encoding scheme $\mathsf{PES_{FP}}$ for this $\mathsf{MultD\text{-}Eq}$ predicate with compatible domains $(\mathcal{X}_{\mathsf{Sub}}, \mathcal{Y}_{\mathsf{Sub}})$, and set up the verification keys so that the following polynomial is implicitly embedded during the security reduction:

$$\hat{C}^{\mathsf{Sub}}_{\mathsf{T}}(\hat{\mathsf{S}}) = \prod_{i=1}^{\eta}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-\hat{\mathsf{S}}_{j,k}) + (-1+2\hat{\mathsf{S}}_{j,k})\cdot\hat{\mathsf{T}}_{i,k}\Big), \tag{5.14}$$

where (informaly) $\hat{\mathsf{T}}_{i,k}, \hat{\mathsf{S}}_{j,k}$ corresponds to the k-th bit of the binary representation of the i-th and j-th element in the set $\mathsf{T},\mathsf{S} \subseteq [2\ell]$, respectively. From the correctness of $\mathsf{PES_{FP}}$, we have $\hat{C}^{\mathsf{Sub}}_{\mathsf{T}}(\hat{\mathsf{S}}) = (\mathsf{T} \subseteq \mathsf{S})$ as desired. Here, recall that the set $\mathsf{S} = \mathsf{S}(X)$ is uniquely constructed for each input $X \in \{0,1\}^n$. Finally, since $\eta = \omega(\log\lambda)$, $\ell = \Theta(\lambda)$, $\zeta = \Theta(\log\lambda)$, the above polynomial will be of degree $\omega(\log^2\lambda)$. Thus, this allows us to simulate the proof π and evaluation Y (and hence prove security of our VRF) under a L-DDH assumption where $L = \omega(\log^2\lambda)$.

Construction. For simplicity of presentation, we deviate slightly from the notations used above. Below, $n, \ell, \eta, \mathsf{S}(\cdot)$ are the parameters and function specified by the modified admissible hash function (Definition 5.10) and ζ is set as $\lfloor \log p \rfloor + 1$.

$\mathsf{Gen}(1^\lambda)$: On input 1^λ, it runs $\Pi \leftarrow \mathsf{GrpGen}(1^\lambda)$ to obtain a group description. It then chooses random generators $g, h \leftarrow \mathbb{G}^*$ and $w_0, w_{i,k} \leftarrow \mathbb{Z}_p$ for $(i,k) \in [\eta]\times[\zeta]$. Finally, it outputs

$$\mathsf{vk} = \left(\Pi, g, h, g_0 := g^{w_0}, \left(g_{i,k} := g^{w_{i,k}}\right)_{(i,k)\in[\eta]\times[\zeta]}\right), \quad \text{and} \quad \mathsf{sk} = \left(w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}\right).$$

$\mathsf{Eval}(\mathsf{sk}, X)$: On input $X \in \{0,1\}^n$, it first computes $\mathsf{S}(X) = \{s_1, \cdots, s_\ell\} \in [2\ell]$. In the following, let $s_{j,k}$ be the k-th bit of the binary representation of s_j, where $k \in [\zeta]$. It then computes

$$\theta_{i'} = \prod_{i=1}^{i'}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k}) + (-1+2s_{j,k})\cdot w_{i,k}\Big), \quad \text{and} \quad \theta_{i,j,k'} = \prod_{k=1}^{k'}\Big((1-s_{j,k}) + (-1+2s_{j,k})\cdot w_{i,k}\Big)$$

for $i' \in [\eta]$ and $(i,j,k') \in [\eta]\times[\ell]\times[\zeta]$, and defines $\theta := \theta_\eta$. Finally, it outputs

$$Y = e(g,h)^{\theta/w_0}, \quad \text{and} \quad \pi = \left(\pi_0 := g^{\theta/w_0}, \left(\pi_{i'} := g^{\theta_{i'}}\right)_{i'\in[\eta]}, \left(\pi_{i,j,k'} := g^{\theta_{i,j,k'}}\right)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\right).$$

$\mathsf{Verify}(\mathsf{vk}, X, (Y,\pi))$: First, it checks the validity of vk. It outputs 0 if any of the following properties are not satisfied.

1. vk is of the form $\left(\Pi, g, h, g_0, (g_{i,k})_{(i,k)\in[\eta]\times[\zeta]}\right)$.

2. $\mathsf{GrpVfy}(\Pi) = 1$ and $\mathsf{GrpVfy}(\Pi, s) = 1$ for all $s \in (g, h, g_0) \cup (g_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$.

Then, it checks the validity of X, Y and π. In doing so, it first prepares the terms $\Phi_{i'}, \bar{g}_{i,j,k'}$ for all $i' \in [\eta], (i, j, k') \in [\eta] \times [\ell] \times [\zeta]$ defined as

$$\Phi_{i'} := \prod_{j=1}^{\ell} \pi_{i',j,\zeta}, \quad \text{and} \quad \bar{g}_{i,j,k'} := g^{1-s_{j,k'}} \cdot (g_{i,k'})^{-1+2s_{j,k'}}.$$

It outputs 0 if any of the following properties are not satisfied.

3. $X \in \{0,1\}^n, Y \in \mathbb{G}_T$, π is of the form $\Big(\pi_0, \big(\pi_{i'}\big)_{i'\in[\eta]}, \big(\pi_{i,j,k'}\big)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\Big)$.
4. It holds that for all $i' \in [\eta - 1]$ and $(i, j, k') \in [\eta] \times [\ell] \times [\zeta - 1]$,

$$\begin{aligned} e(\pi_1, g) &= e(\Phi_1, g), & e(\pi_{i,j,1}, g) &= e(\bar{g}_{i,j,1}, g), \\ e(\pi_{i'+1}, g) &= e(\Phi_{i'+1}, \pi_{i'}), & e(\pi_{i,j,k'+1}, g) &= e(\bar{g}_{i,j,k'+1}, \pi_{i,j,k'}). \end{aligned}$$

5. It holds that $e(\pi_\eta, g) = e(\pi_0, g_0)$ and $e(\pi_0, h) = Y$.

If all the above checks are passed, it outputs 1.

5.5.3 Correctness, Unique Provability, and Pseudorandomness

Theorem 5.2 (Correctness and Unique Provability). *Our scheme forms a correct verifiable random function and satisfies the unique provability requirement.*

Proof. We first prove the correctness of the scheme. It is easily seen that when Gen and Eval are properly run, then it passes Step 1, 2, 3 of the verification algorithm. Next, observe that

$$\begin{aligned} \Phi_{i'} &= \prod_{j=1}^{\ell} \pi_{i',j,\zeta} = g^{\sum_{j=1}^{\ell} \theta_{i',j,\zeta}} = g^{\sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \left((1-s_{j,k}) + (-1+2s_{j,k})\cdot w_{i',k}\right)} \\ \bar{g}_{i,j,k'} &= g^{1-s_{j,k'}} \cdot (g_{i,k'})^{-1+2s_{j,k'}} = g^{\left((1-s_{j,k'}) + (-1+2s_{j,k'})\cdot w_{i,k'}\right)}, \end{aligned}$$

for all $i' \in [\eta]$ and $(i, j, k') \in [\eta] \times [\ell] \times [\zeta]$. Since $\Phi_1 = \pi_1$ and $\bar{g}_{i,j,1} = \pi_{i,j,1}$, the first two equation in Step 4 holds. The equality of the rest of the equations in Step 4 follow using the additional observation that

$$\begin{cases} \theta_{i'+1} = \theta_{i'} \cdot \Big(\displaystyle\sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot w_{i'+1,k}\big) \Big) \\ \theta_{i,j,k'+1} = \theta_{i,j,k'} \cdot \big((1 - s_{j,k'+1}) + (-1 + 2s_{j,k'+1}) \cdot w_{i,k'+1}\big) \end{cases},$$

for all $i' \in [\eta - 1]$ and $(i, j, k') \in [\eta] \times [\ell] \times [\zeta - 1]$. Finally, since by definition $\pi_\eta = g^{\theta_\eta} = g^{\theta}$, Step 5 holds. This completes the proof of the correctness of the scheme.

Next, we turn to prove the unique provability of the scheme. We have to show that for any $\mathsf{vk} \in \{0,1\}^*$ and $X \in \{0,1\}^n$, there does not exist any (Y_0, π_0, Y_1, π_1) such that $Y_0 \neq Y_1$ and $\mathsf{Verify}(\mathsf{vk}, X, (Y_0, \pi_0)) = \mathsf{Verify}(\mathsf{vk}, X, (Y_1, \pi_1)) = 1$.

- First of all, in Step 1 and Step 2, the verification algorithm checks whether Π contains valid certified bilinear group parameters and checks whether all group elements $g, h, g_0, (g_{i,j})_{(i,j)\in[\eta]\times[\zeta]}$ are valid group elements with respect to Π. Thus, in the following, we may assume that all these group elements are valid and have a unique encoding.

- In Step 3, it is checked whether $X \in \{0,1\}^n$, $Y \in \mathbb{G}_T$ and π is in the proper form. In Step 4 and Step 5 it inductively checks whether all the equality holds. Now, since the bilinear group is satisfied, i.e., each group element has a unique encoding and the bilinear map is non-degenerate, there exists only one unique π such that correctness holds.

Therefore, the value of (Y, π) is uniquely determined by the input X and the verification key vk. This completes the proof. $\square$

Theorem 5.3 (Pseudorandomness)**.** *Our scheme satisfies pseudorandomness assuming L-*DDH *with $L = \eta\zeta = \omega(\log^2 \lambda)$.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the pseudorandomness of the scheme with non-negligible advantage. Let $\epsilon = \epsilon(\lambda)$ be its advantage and $Q = Q(\lambda)$ be the upper bound on the number of evaluation queries it makes. Here, since $\mathcal{A}$ is a valid adversary, Q is a polynomially bounded function and there exists a noticeable function $\epsilon_0 = \epsilon_0(\lambda)$ such that $\epsilon(\lambda) \ge \epsilon_0(\lambda)$ holds for infinitely many λ. Then combining Definition 5.9 and Theorem 5.1 together, for $\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$, we have $\mathsf{T} \subseteq [2\ell]$ and $|\mathsf{T}| < \eta$ with probability 1 for all sufficiently large λ. Therefore, in the following, we assume this condition always holds. We show security of the scheme through a sequence of games. In each game, a value $\mathsf{coin}' \in \{0,1\}$ is defined. While it is set $\mathsf{coin}' = \widehat{\mathsf{coin}}$ in the first game, these values might be different in the later games. In the following we define E_i to be the event that $\mathsf{coin}' = \mathsf{coin}$ in Game_i.

Game_0 : This is the actual security game. Since $\mathcal{Y} = \mathbb{G}_T$, when $\mathsf{coin} = 1$, a random element $Y_1^* \leftarrow \mathbb{G}_T$ is returned to $\mathcal{A}$ as the challenge query. At the end of the game, $\mathcal{A}$ outputs a guess $\widehat{\mathsf{coin}}$ for coin. Finally, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By assumption on the adversary $\mathcal{A}$, we have

$$\left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\widehat{\mathsf{coin}} = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon.$$

Game_1 : In this game, we change Game_0 so that the challenger performs an additional step at the end of the game. Namely, the challenger first runs $\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$ from Theorem 5.1. As noted earlier, we have $|\mathsf{T}| \subseteq [2\ell]$ and $|\mathsf{T}| < \eta$. Then, it checks whether the following condition holds:

$$\begin{aligned} & \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X^{(1)}) = 1 \ \wedge \ \cdots = \ \wedge \ \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X^{(Q)}) = 1 \ \wedge \ \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X^*) = 0 \\ \iff & \left(\mathsf{T} \not\subseteq \mathsf{S}(X^{(1)})\right) \ \wedge \ \cdots \ \wedge \ \left(\mathsf{T} \not\subseteq \mathsf{S}(X^{(Q)})\right) \ \wedge \ \left(\mathsf{T} \subseteq \mathsf{S}(X^*)\right) \end{aligned} \tag{5.15}$$

where X^* is the challenge input and $\{X^{(i)}\}_{i\in[Q]}$ are the inputs for which $\mathcal{A}$ has queried the evaluation of the function. If it does not hold, the challenger ignores the output $\widehat{\mathsf{coin}}$ of $\mathcal{A}$ and sets $\mathsf{coin}' \leftarrow \{0,1\}$. In this case, we say that the challenger aborts. If condition (5.15) holds, the challenger sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. By Lemma 5.1 and Theorem 5.1 (See also Definition 5.9, Item 2), the following holds for infinitely many λ:

$$\left|\Pr[\mathsf{E}_1] - \frac{1}{2}\right| \ge \gamma_{\min} \cdot \epsilon - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$

$$\geq \gamma_{\min} \cdot \epsilon_0 - \frac{\gamma_{\max} - \gamma_{\min}}{2}$$
$$\geq \tau,$$

where $\tau = \tau(\lambda)$ is a noticeable function. Recall that $\gamma_{\max}, \gamma_{\min}, \tau$ are functions specified by Q, ϵ_0 and the underlying partitioning function $\mathsf{F}_{\mathsf{MAH}}$.

Game_2 : In this game, we change the way $w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ are chosen. First, at the beginning of the game, the challenger picks $\mathsf{T} \leftarrow \mathsf{PrtSmp}_{\mathsf{MAH}}(1^\lambda, Q(\lambda), \epsilon_0(\lambda))$ and parses it as $\mathsf{T} = \{t_1, \cdots, t_{\eta'}\} \subset [2\ell]$. Note that changing the time on which the adversary runs the algorithm is only conceptual. Now, recalling that by our assumption $\eta' < \eta$, it sets $t_i = 0$ for $i \in [\eta'+1, \eta]$. Next, it samples $\alpha \leftarrow \mathbb{Z}_p^*$ and $\tilde{w}_0, \tilde{w}_{i,k} \leftarrow \mathbb{Z}_p$ for $(i,k) \in [\eta] \times [\zeta]$. Finally, the challenger sets

$$w_0 = \tilde{w}_0 \cdot \alpha, \quad w_{i,k} = \tilde{w}_{i,k} \cdot \alpha + t_{i,k} \quad \text{for} \quad (i,k) \in [\eta] \times [\zeta], \tag{5.16}$$

where $t_{i,k}$ is the k-th bit of the binary representation of t_i. The rest of the game is identical to Game_1. Here, the statistical distance of the distributions of $w_0, (w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ in Game_1 and Game_2 is at most $(\eta\zeta + 1)/p$, which is negligible. Therefore, we have

$$|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda).$$

Before, getting into Game_3, we introduce polynomials (associated with each input X) that implicitly embeds the information on the partitioning function $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X)$, i.e., the form of the polynomials depend on whether $\mathsf{T} \subseteq \mathsf{S}(X)$ or not. For any $\mathsf{T} \subseteq [2\ell]$ with $|\mathsf{T}| = \eta' < \eta$ and $X \in \{0,1\}^n$ (i.e., for any $\mathsf{S}(X)$), we define the polynomial $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) : \mathbb{Z}_p \to \mathbb{Z}_p$ as

$$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \prod_{i=1}^{\eta} \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot (\tilde{w}_{i,k}\mathsf{Z} + t_{i,k}) \Big), \tag{5.17}$$

where $\{s_{j,k}\}_{(j,k)\in[\ell]\times[\zeta]}$ and $\{t_{i,k}\}_{(i,k)\in[\eta]\times[\zeta]}$ are defined as in Game_2. Note that $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha) = \theta$. Our security proof is built upon the following lemma on the partitioning function.

Lemma 5.4. *There exists $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) : \mathbb{Z}_p \to \mathbb{Z}_p$ such that*

$$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \begin{cases} 1 + \mathsf{Z} \cdot \mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}), & \text{if } \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 0 \\ \mathsf{Z} \cdot \mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}), & \text{if } \mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 1 \end{cases}.$$

In other words, $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ is not divisible by Z if and only if $\mathsf{T} \subseteq \mathsf{S}(X)$.

So as not to interrupt the proof of Theorem 5.3, we intentionally skip the proof of Lemma 5.4 for the time being. Furthermore, with an abuse of notation, for all $X \in \{0,1\}^n$, we define the following polynomials that map $\mathbb{Z}_p$ to $\mathbb{Z}_p$, which are defined analogously to the values computed during Eval:

$$\begin{cases} \theta_{i'}^X(\mathsf{Z}) = \prod_{i=1}^{i'} \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k})(\tilde{w}_{i,k}\mathsf{Z} + t_{i,k}) \Big) \\ \theta_{i,j,k'}^X(\mathsf{Z}) = \prod_{k=1}^{k'} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k})(\tilde{w}_{i,k}\mathsf{Z} + t_{i,k}) \Big) \end{cases},$$

for $i' \in [\eta]$ and $(i,j,k') \in [\eta]\times[\ell]\times[\zeta]$, and define $\theta^X(\mathsf{Z}) := \theta_\eta^X(\mathsf{Z})$. Note that we have $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \theta^X(\mathsf{Z}), \theta_{i'} = \theta_{i'}^X(\alpha), \theta_{i,j,k'} = \theta_{i,j,k'}^X(\alpha)$, and $\theta = \theta^X(\alpha)$.

Game_3 : Recall that in the previous game, the challenger aborts at the end of the game if condition (5.15) is not satisfied. In this game, we change the game so that the challenger aborts as soon as the abort condition becomes true. Since this is only a conceptual change, we have

$$\Pr[\mathsf{E}_2] = \Pr[\mathsf{E}_3].$$

Game_4 : In this game, we change the way the evaluation queries are answered. When the adversary $\mathcal{A}$ queries an input X to be evaluated, it first checks whether $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 1$, i.e., it checks if condition (5.15) is satisfied. If it does not hold, it aborts as in Game_3. Otherwise, it computes the polynomial $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{Z}]$ such that $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z}) = \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$, and returns

$$Y = e(g^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)/\tilde{w}_0}, h), \tag{5.18}$$

$$\pi = \left(\pi_0 = g^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)/\tilde{w}_0}, \left(\pi_{i'} = g^{\theta^X_{i'}(\alpha)}\right)_{i'\in[\eta]}, \left(\pi_{i,j,k'} = g^{\theta^X_{i,j,k'}(\alpha)}\right)_{(i,j,k')\in[\eta]\times[\ell]\times[\zeta]}\right). \tag{5.19}$$

Note that existence of such a polynomial $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ is guaranteed by Lemma 5.4. By the definition of $\theta^X_{i'}(\mathsf{Z})$ and $\theta^X_{i,j,k'}(\mathsf{Z})$, the components $\pi_{i'}$ and $\pi_{i,j,k'}$ are correctly generated. Furthermore, we have

$$\frac{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{\tilde{w}_0} = \frac{\alpha\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{\alpha\cdot\tilde{w}_0} = \frac{\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\alpha)}{w_0} = \frac{\theta}{w_0}.$$

Therefore, Y and π_0 are also correctly generated, and the challenger simulates the evaluation queries perfectly. Hence,

$$\Pr[\mathsf{E}_3] = \Pr[\mathsf{E}_4].$$

Game_5 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. Recall in the previous games when $\mathsf{coin} = 0$, we created a valid $Y^*_0 = \mathsf{Eval}(\mathsf{sk}, X^*)$ as in the real scheme. If $\mathsf{coin} = 0$ and $\mathsf{F}_{\mathsf{MAH}}(X^*) = 0$ (i.e., if it does not abort), to create Y^*_0, the challenger first computes the polynomial $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z}) \in \mathbb{Z}_p[\mathsf{X}]$ such that $\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z}) = 1 + \mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z})$, whose existence is guaranteed by Lemma 5.4. It then sets,

$$Y^*_0 = \left(e(g,h)^{1/\alpha}\cdot e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}\right)^{1/\tilde{w}_0}$$

and returns it to $\mathcal{A}$. Here, the above term can be written equivalently as

$$\left(e(g,h)^{1/\alpha}\cdot e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}\right)^{1/\tilde{w}_0} = e(g^{(1+\alpha\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha))/\alpha\tilde{w}_0}, h) = e(g^{\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)/w_0}, h) = e(g^{\theta/w_0}, h).$$

Therefore, the view of the adversary in unchanged. Hence,

$$\Pr[\mathsf{E}_4] = \Pr[\mathsf{E}_5].$$

Game_6 : In this game, we change the challenge value to be a random value in $\mathbb{G}_T$ regardless of whether $\mathsf{coin} = 0$ of $\mathsf{coin} = 1$. Namely, the challenger sets $Y^* \leftarrow \mathbb{G}_T$. As we will show in Lemma 5.5, assuming *L-DDH* is hard for $L = \eta\zeta$, we have $|\Pr[\mathsf{E}_5] = \Pr[\mathsf{E}_6]| = \mathsf{negl}(\lambda)$.

Analysis. From the above, we have

$$\begin{aligned}\left|\Pr[\mathsf{E}_6]-\frac{1}{2}\right| &= \left|\Pr[\mathsf{E}_1]-\frac{1}{2}+\sum_{i=1}^{5}\left(\Pr[\mathsf{E}_{i+1}]-\Pr[\mathsf{E}_i]\right)\right| \\ &\geq \left|\Pr[\mathsf{E}_1]-\frac{1}{2}\right|-\sum_{i=1}^{5}\left|\Pr[\mathsf{E}_{i+1}]-\Pr[\mathsf{E}_i]\right| \\ &\geq \tau(\lambda)-\mathsf{negl}(\lambda), \end{aligned} \tag{5.20}$$

for infinitely many λ. Since $\Pr[\mathsf{E}_6]=1/2$, this implies $\tau(\lambda)\leq\mathsf{negl}(\lambda)$ for infinitely many λ, which is a contradiction. □

To complete the proof of Theorem 5.3, it remains to prove Lemma 5.4 and 5.5.

Proof of Lemma 5.4. First, we can rewrite Eq.(5.17) as

$$\mathsf{P}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z})=\mathsf{Z}\cdot\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z})+\underbrace{\prod_{i=1}^{\eta}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k})+(-1+2s_{j,k})\cdot t_{i,k}\Big)}_{=C},$$

for some polynomial $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(\mathsf{X})}(\mathsf{Z})$ with degree at most $\eta\zeta$. Observe the constant term C is computing the circuit outputted by the functionality preserving encoding scheme $\mathsf{PES}_{\mathsf{FP}}$ for the subset predicate (See Lemma 5.2). Therefore, we have $C=1$ if $\mathsf{T}\subseteq\mathsf{S}\Leftrightarrow\mathsf{F}_{\mathsf{MAH}}(\mathsf{T},X)=0$, and $C=0$ otherwise, as desired. □

Lemma 5.5. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[\mathsf{E}_5]-\Pr[\mathsf{E}_6]|\leq\mathsf{Adv}_{\mathcal{B}}^{L\text{-}DDH},$$

where $L=\eta\zeta$. In particular, under the L-DDH assumption, we have $|\Pr[\mathsf{E}_5]-\Pr[\mathsf{E}_6]|=\mathsf{negl}(n)$.

Proof. Suppose an adversary $\mathcal{A}$ that has non-negligible advantage in distinguishing Game_5 and Game_6. We use $\mathcal{A}$ to construct an *L-DDH* algorithm denoted $\mathcal{B}$, which proceeds as follows.

Instance. $\mathcal{B}$ is given the problem instance of $L\text{-}DDH(\Pi,g,h,\{g^{\alpha^i}\}_{i\in L},\Psi)$ for $L=\eta\zeta$, where $\Psi=e(g,h)^{1/\alpha}$ or $\Psi\leftarrow\mathbb{G}_T$.

Setup. To construct the verification key vk, it samples $\tilde{w}_0,\tilde{w}_{i,k}\leftarrow\mathbb{Z}_p$ for $(i,k)\in[\eta]\times[\zeta]$ as in Game_2, and implicitly sets w_0 and $(w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ as in Eq. (5.16). Then, since $w_0,(w_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ are all at most degree 1 polynomials in α, $\mathcal{B}$ can efficiently compute $g_0,(g_{i,k})_{(i,k)\in[\eta]\times[\zeta]}$ from the problem instance. Finally, it returns $\mathsf{vk}=(\Pi,g,h,g_0,(g_{i,k})_{(i,k)\in[\eta]\times[\zeta]})$ to $\mathcal{A}$. $\mathcal{B}$ also picks a random bit $\mathsf{coin}\leftarrow\{0,1\}$ and keeps it secret.

Phase 1 and Phase 2. The evaluation queries made by $\mathcal{A}$ are answered as in Eq. (5.18) of Game_4. Observe that this can be done efficiently, since $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X)}(\mathsf{Z})$ is degree at most $L=\eta\zeta$.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge input X^*, $\mathcal{B}$ first computes $\mathsf{F}_{\mathsf{MAH}}(X^*)$. Then, it aborts and sets $\mathsf{coin}'\leftarrow\{0,1\}$ if $\mathsf{F}_{\mathsf{MAH}}(X^*)=1$. Otherwise, it proceeds as follows. If $\mathsf{coin}=0$, it computes

$$Y_0^*=\left(\Psi\cdot e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}\right)^{1/\tilde{w}_0}.$$

Note that $e(g,h)^{\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\alpha)}$ can be efficiently computed from the problem instance, since $\mathsf{R}_{\mathsf{T}\subseteq\mathsf{S}(X^*)}(\mathsf{Z})$ is of degree at most $L = \eta\zeta$. In the case of $\mathsf{coin} = 1$, $\mathcal{B}$ sets $Y^* \leftarrow \mathbb{G}_T$. In both cases, $\mathcal{B}$ returns Y^* to $\mathcal{A}$.

Guess. At last, $\mathcal{A}$ outputs its guess $\widehat{\mathsf{coin}}$ (if the abort condition has not been satisfied). Then, $\mathcal{B}$ sets $\mathsf{coin}' = \widehat{\mathsf{coin}}$. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise.

Analysis. It can be seen that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in Game_5 if $\Psi = e(g,h)^{1/\alpha}$ and Game_6 if $\Psi \leftarrow \mathbb{G}_T$. We therefore conclude that

$$\mathsf{Adv}_{\mathcal{B}}^{L\text{-}DDH} = |\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]|$$

for $L = \eta\zeta$ as desired.

□

5.5.4 Achieving Smaller Proof Size

In this section, we propose a variant of the VRF presented in Section 5.5.2 with a much shorter proof size. Recall the VRF we constructed in the previous section had a very small verification key size $|\mathsf{vk}| \approx \eta\zeta = \omega(\log^2 \lambda)$ with a rather large proof size $|\pi| \approx \eta\ell\zeta = \omega(\lambda \log^2 \lambda)$, where we count the number of group elements for size. A first attempt is to use a similar trick used in [Yam17] to add some helper terms in the verification key to make the proof size smaller. In particular, we can convert our VRF to have a very small proof size $|\pi| = \omega(\log \lambda)$ by allowing the verification key size to grow quasi-linearly in the security parameter, i.e., $|\mathsf{vk}| = \omega(\lambda \log \lambda)$.

However, we can do much better with an additional idea; we obtain a VRF with proof size $|\pi| = \omega(\log \lambda)$ and verification key size $|\mathsf{vk}| = \omega(\sqrt{\lambda} \log \lambda)$, which is now sublinear.

Preparation. We define *power tuples* $\mathcal{P}(W)$ for a tuple W, analogously to power sets. Namely, we create a tuple that contains all the subsequence of W in lexicographical order, i.e., $\mathcal{P}(W) = (w_1, w_2, w_3, w_1w_2, w_1w_3, w_2w_3, w_1w_2w_3)$ for $W = (w_1, w_2, w_3)$. Here, we do not consider the empty string as a subsequence of W. For a group element $g \in \mathbb{G}$ or $\mathbb{G}_T$ and a tuple W with elements in $\mathbb{Z}_p$, we denote $g^{\mathcal{P}(W)}$ as the tuple $(g^w \mid w \in \mathcal{P}(W))$. Furthermore, for tuples W, W' with elements in $\mathbb{Z}_p$ we define $e(g^{\mathcal{P}(W)}, g^{\mathcal{P}(W')})$ to be the tuple $(e(g,g)^{ww'} \mid w \in W, w' \in W')$. Assume all the tuples are sorted in the lexicographical order.

Construction. Below, we provide a VRF with small proof size.

$\mathsf{Gen}(1^\lambda)$: On input 1^λ, it runs $\Pi \leftarrow \mathsf{GrpGen}(1^\lambda)$ to obtain a group description. It then chooses random generators $g, h \leftarrow \mathbb{G}^*$, $w_0, w_{i,k} \leftarrow \mathbb{Z}_p$ for $(i,k) \in [\eta] \times [\zeta]$ and sets $L_i = (w_{i,k})_{k \in [\lfloor \zeta/2 \rfloor]}$ and $R_i = (w_{i,k})_{k \in [\lfloor \zeta/2 \rfloor + 1, \zeta]}$. Finally, it outputs

$$\mathsf{vk} = \left(\Pi, g, h, g_0 := g^{w_0}, (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i \in [\eta]} \right), \quad \text{and} \quad \mathsf{sk} = \left(w_0, (w_{i,k})_{(i,k) \in [\eta] \times [\zeta]} \right).$$

Note that we have $e(g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)}) = e(g,g)^{\mathcal{P}(W_i)}$ where $W_i = (w_{i,k})_{k \in [\zeta]}$.

$\mathsf{Eval}(\mathsf{sk}, X)$: On input $X \in \{0,1\}^n$, it first computes $\mathsf{S}(X) = \{s_1, \cdots, s_\ell\} \in [2\ell]$. In the following, let $s_{j,k}$ be the k-th bit of the binary representation of s_j, where $k \in [\zeta]$. It then computes

$$\begin{cases} \theta_i = \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot w_{i,k} \Big) \\ \theta_{[1:i']} = \prod_{i=1}^{i'} \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot w_{i,k} \Big) \end{cases},$$

for $i \in [\eta], i' \in [2, \eta]$ and sets $\theta := \theta_{[1:\eta]}$. Note that we do not require $i' = 1$ since $\theta_1 = \theta_{[1:1]}$. Finally, it outputs

$$Y = e(g,h)^{\theta/w_0}, \quad \text{and} \quad \pi = \left(\pi_0 := g^{\theta/w_0}, \left(\pi_i := g^{\theta_i} \right)_{i \in [\eta]}, \left(\pi_{[1:i']} := g^{\theta_{[1:i']}} \right)_{i' \in [2,\eta]} \right).$$

$\mathsf{Verify}(\mathsf{vk}, X, (Y, \pi))$: First, it checks the validity of vk. It outputs 0 if any of the following properties are not satisfied.

1. vk is of the form $\left(\Pi, g, h, g_0, (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i \in [\eta]} \right)$.
2. $\mathsf{GrpVfy}(\Pi) = 1$ and $\mathsf{GrpVfy}(\Pi, s) = 1$ for all $s \in (g, h, g_0) \cup (g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i \in [\eta]}$.

Then, it checks the validity of X, Y and π. In doing so, it first computes the coefficients $(\alpha_S)_{S \subseteq [\zeta]}$ of the multi-variate polynomial

$$p(\mathsf{Z}_1, \cdots, \mathsf{Z}_\zeta) = \sum_{j=1}^{\ell} \prod_{k=1}^{\zeta} \Big((1 - s_{j,k}) + (-1 + 2s_{j,k}) \cdot \mathsf{Z}_k \Big) = \sum_{S \subseteq [\zeta]} \alpha_S \prod_{k \in S} \mathsf{Z}_k.$$

Next, for all $i \in [\eta]$ and $S \subseteq [\zeta]$, it sets $L_S = S \cap [\lfloor \zeta/2 \rfloor]$ and $R_S = S \cap [\lfloor \zeta/2 \rfloor + 1, \zeta]$, and computes $\Phi_{i,S}$ as

$$\Phi_{i,S} = e(g^{\prod_{k \in L_S} w_{i,k}}, g^{\prod_{k \in R_S} w_{i,k}}).$$

Here, in case $L_S = \phi$ (resp. $R_S = \phi$), we define $\prod_{k \in L_S} w_{i,k}$ (resp. $\prod_{k \in R_S} w_{i,k}$) to be 1. Note that these values can be computed efficiently, since $g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)}$ are given as part of the verification key. It outputs 0 if any of the following properties are not satisfied.

3. $X \in \{0,1\}^n, Y \in \mathbb{G}_T$, π is of the form $\pi = (\pi_0, (\pi_i)_{i \in [\eta]}, (\pi_{[1:i']})_{i' \in [2,\eta]})$.
4. It holds that for all $i \in [\eta]$ and $i' \in [3, \eta]$,

$$e(\pi_i, g) = \prod_{S \subseteq [\zeta]} \Phi_{i,S}^{\alpha_S}, \quad \text{and} \quad e(\pi_{[1:2]}, g) = e(\pi_1, \pi_2), \quad \text{and} \quad e(\pi_{[1:i']}, g) = e(\pi_{[1:i'-1]}, \pi_{i'}).$$

5. It holds that $e(\pi_{[1:\eta]}, g) = e(\pi_0, g_0)$ and $e(\pi_0, h) = Y$.

If all the above checks are passed, it outputs 1.

The correctness, unique provability and pseudorandomness of the above VRF can be proven in a similar manner to the VRF in Section 5.5.2. For completeness, we show correctness and a proof sketch for pseudorandomness of our VRF. The proofs follow closely to the ones given in Section 5.5.3. We omit the proof for the unique provability, since it is the same as the one given in Section 5.5.3.

Theorem 5.4 (Correctness). *Our VRF from Section 5.5.4 forms a correct verifiable random function.*

Proof. We first prove the correctness of the scheme. It is easily seen that when Gen and Eval are properly run, then it passes Step 1, 2, 3 of the verification algorithm. Next, observe that for all $i \in [\eta]$ we have

$$\begin{aligned}\prod_{S \subseteq [\zeta]} \Phi_{i,S}^{\alpha_S} &= \prod_{S \subseteq [\zeta]} \left(e(g^{\prod_{k \in L_S} w_{i,k}}, g^{\prod_{k \in R_S} w_{i,k}})\right)^{\alpha_S} \\ &= \prod_{S \subseteq [\zeta]} \left(e(g,g)^{\prod_{k \in S} w_{i,k}}\right)^{\alpha_S} \\ &= e(g,g)^{\sum_{S \subseteq [\zeta]} \alpha_S \prod_{k \in S} w_{i,k}} \\ &= e(g,g)^{p(w_{i,1}, \cdots, w_{i,\zeta})}\end{aligned}$$

Since $\theta_i = p(w_{i,1}, \cdots, w_{i,\zeta})$, the first equation in Step 4 holds. The equality of the rest of the equations in Step 4 follow using the additional observation that $\theta_{[1:i']} \cdot \theta_{i'+1} = \theta_{[1:i'+1]}$ for $i' \in [\eta - 1]$, where $\theta_{[1:1]} = \theta_1$. Finally, since by definition $\pi_{[1:\eta]} = g^{\theta_{[1:\eta]}} = g^{\theta}$, Step 5 holds. This completes the proof of the correctness of the scheme. □

The proof of pseudorandomness follows very closely to the proof given in Section 5.5.3. Notably, the VRF is proven under the L-DDH assumption where $L = \eta\zeta = \omega(\log^2 \lambda)$. Therefore, to avoid being redundant, we point out the main differences between the proof in Section 5.5.3 and restrict ourselves to an overview of the security proof.

Proof Sketch. At a high level, the strategy of the proof is the same; we show that we can simulate all the components in the verification key and a valid output Y^* for the challenge input X^* using the L-DDH instance $\{g^{\alpha^i}\}_{i \in [\eta\zeta]}$. Here, note that if we can simulate a valid output Y^*, we can also simulate a valid proof for any input X such that $\mathsf{T} \not\subseteq \mathsf{S}(X)$. We first show that the challenger can correctly simulate the verification key. As in Game_2, Eq. (5.16) of the previous proof, the challenger sets

$$w_0 = \tilde{w}_0 \cdot \alpha, \quad w_{i,k} = \tilde{w}_{i,k} \cdot \alpha + t_{i,k} \quad \text{for} \quad (i,k) \in [\eta] \times [\zeta].$$

To create the rest of $(g^{\mathcal{P}(L_i)}, g^{\mathcal{P}(R_i)})_{i \in [\eta]}$, it can simply use $\{g^{\alpha^i}\}_{i \in [\eta\zeta]}$ since the terms in $\mathcal{P}(L_i), \mathcal{P}(R_i)$ are at most degree ζ. Recall $L_i = (w_{i,k})_{k \in [\lfloor \zeta/2 \rfloor]}$ and $R_i = (w_{i,k})_{k \in [\lfloor \zeta/2 \rfloor + 1 : \zeta]}$. Furthermore, since we use the same degree $\eta\zeta$ polynomial $\mathsf{P}_{\mathsf{T} \subseteq \mathsf{S}(X)}(\mathsf{Z})$ as in Eq. (5.17) to embed the partitioning function $\mathsf{F}_{\mathsf{MAH}}$, we can correctly simulate the proof as in Game_5 using $\{g^{\alpha^i}\}_{i \in [\eta\zeta]}$. Thus, we have that our VRF is adaptively pesudorandom. □

Combining everything together, our second VRF satisfies all the desired properties under the L-DDH assumption where $L = \omega(\log^2 \lambda)$. Finally, we end this section by discussing the efficiency of the above construction.

Remark 5.3. *Our second VRF has verification key size $|\mathsf{vk}| = \omega(\sqrt{\lambda} \log \lambda)$ and proof size $|\pi| = \omega(\log \lambda)$. To see this, observe that for all $i \in [\eta]$ we have $|\mathcal{P}(L_i)|, |\mathcal{P}(R_i)| \leq 2^{\lceil \zeta/2 \rceil}$, which follows from $|L_i|, |R_i| \leq \lceil \zeta/2 \rceil$. Next, since $\ell = \Theta(\lambda)$, there exists some positive constant c such that $\ell(\lambda) \leq c\lambda$ for large enough $\lambda \in \mathbb{N}$. Then, since $\zeta = \lfloor \log \ell \rfloor + 1$, we have $\zeta(\lambda) \leq \log \lambda + \log c + 1$. Therefore, $2^{\lceil \zeta/2 \rceil} \leq 2^{\zeta/2+1} = c' \lambda^{1/2}$ for some positive constant c'. Thus, we obtain the upper bound $|g^{\mathcal{P}(L_i)}|, |g^{\mathcal{P}(R_i)}| = O(\sqrt{\lambda})$. Since we consider this for all $i \in [\eta]$ where $\eta = \omega(\log \lambda)$, we conclude $|\mathsf{vk}| = \omega(\sqrt{\lambda} \log \lambda)$. Note that this means that we can take vk for example as small as $|\mathsf{vk}| = O(\sqrt{\lambda} \log^2 \lambda)$. A detailed comparison is provided in Section 5.1.1, Table 5.1.*

5.6 Predicate Encryption for $\mathsf{MultD\text{-}Eq}$ Predicates

In this section, we show how to construct a predicate encryption scheme for the multi-dimensional equality predicates $\mathsf{MultD\text{-}Eq}$. This directly yields predicate encryption schemes for all the predicates presented in Section 5.4.3. Due to the symmetry of the $\mathsf{MultD\text{-}Eq}$ predicate and the compatible domains $(\mathcal{X}, \mathcal{Y})$, we obtain both key-policy and ciphertext-policy predicate encryption schemes.

5.6.1 Embedding Predicate Encoding Schemes into Matrices

[BGG+14a] provides us with a generic way of constructing a lattice-based attribute-based encryption (ABE) scheme from three deterministic algorithms $(\mathsf{Eval}_{\mathrm{pk}}, \mathsf{Eval}_{\mathrm{ct}}, \mathsf{Eval}_{\mathrm{sim}})$. In this chapter, we slightly modify the syntax of the $\mathsf{Eval}_{\mathrm{ct}}$ algorithm so that the three deterministic algorithms yield a predicate encryption (equivalently, a predicate hiding ABE) scheme.

Definition 5.11. *We say that the* deterministic *algorithms* $(\mathsf{Eval}_{\mathrm{pk}}, \mathsf{Eval}_{\text{ct-priv}}, \mathsf{Eval}_{\mathrm{sim}})$ *are* $\alpha_{\mathcal{C}}$*-predicate encryption (PE) enabling for a family of arithmetic circuits* $\mathcal{C} = \{C : \mathbb{Z}_q^t \to \mathbb{Z}_q\}$ *if they are efficient and satisfy the following properties:*

- $\mathsf{Eval}_{\mathrm{pk}}\left(C \in \mathcal{C},\ \ \mathbf{B}_0, (\mathbf{B}_i)_{i\in[t]} \in \mathbb{Z}_q^{n\times m}\right) \to \mathbf{B}_C \in \mathbb{Z}_q^{n\times m}$
- $\mathsf{Eval}_{\text{ct-priv}}\left(C \in \mathcal{C},\ \ \mathbf{c}_0, (\mathbf{c}_i)_{i\in[t]} \in \mathbb{Z}_q^{n}\right) \to \mathbf{c}_C \in \mathbb{Z}_q^{m}$
- $\mathsf{Eval}_{\mathrm{sim}}\left(C \in \mathcal{C},\ \ \mathbf{R}_0, (\mathbf{R}_i)_{i\in[t]} \in \mathbb{Z}^{m\times m}\right) \to \mathbf{R}_C \in \mathbb{Z}^{m\times m}$

We further require that the following holds:

1. $\mathsf{Eval}_{\mathrm{pk}}(C, (\mathbf{A}\mathbf{R}_0 - \mathbf{G}), (\mathbf{A}\mathbf{R}_i - x_i\mathbf{G})_{i\in[t]}) = \mathbf{A}\cdot \mathsf{Eval}_{\mathrm{sim}}(C, \mathbf{R}_0, (\mathbf{R}_i)_{i\in[t]}) - C(\mathbf{x})\mathbf{G}$ *for any* $\mathbf{x} = (x_1, \cdots, x_t) \in \{0,1\}^t$*.*
2. *If* $\mathbf{c}_0 = (\mathbf{B}_0 + \mathbf{G})^\top \mathbf{s} + \mathbf{z}_0$ *and* $\mathbf{c}_i = (\mathbf{B}_i + x_i\mathbf{G})^\top \mathbf{s} + \mathbf{z}_i$ *for some* $\mathbf{s} \in \mathbb{Z}_q^n$*, and* $\mathbf{z}_0, \mathbf{z}_i \leftarrow D_{\mathbb{Z}^m,\beta}, x_i \in \{0,1\}$ *for all* $i \in [t]$*, then* $\|\mathbf{c}_C - (\mathbf{B}_C + C(\mathbf{x})\mathbf{G})^\top \mathbf{s}\|_2 < \alpha_{\mathcal{C}} \cdot \beta\sqrt{m}$ *with all but negligible probability.*
3. *If* $\mathbf{R}_i \leftarrow \{-1,1\}^{m\times m}$ *for all* $i \in [0,t]$*, then* $s_1(\mathbf{R}_C) < \alpha_{\mathcal{C}}$ *with all but negligible probability.*

There are two major differences between the notions from [BGG+14a]. First, $\mathsf{Eval}_{\text{ct-priv}}$ does not take $(x_i)_{i\in[t]} \in \{0,1\}^t$ as input to the homomorphic evaluation of the ciphertexts. On one hand this limits us to perform only linear operations over the ciphertexts, however, on the other hand this will allow the decryptor to create $\mathbf{c}_C$ without knowledge of the predicate associated to the ciphertext (See also [AFV11]). Second, we loosen the condition on $\mathbf{z}, \mathbf{R}$ in Requirement 1, 2 to hold with overwhelming probability. This allows us to obtain tighter bounds on the behavior of the random matrices and error vectors. Finally, we make a minor change by additionally including $(\mathbf{B}_0, \mathbf{c}_0, \mathbf{R}_0)$ as inputs to the algorithms to cope with the constant terms of the polynomials being evaluated.

$\alpha_{\mathcal{C}}$-PE enabling algorithms for $\mathsf{MultD\text{-}Eq}$ predicates. We show that the linear predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}}$ for the $\mathsf{MultD\text{-}Eq}$ predicates (Section 5.4.2, Lemma 5.3) provides us with a family of arithmetic circuits $\hat{\mathcal{C}}$ that allows for $\alpha_{\hat{\mathcal{C}}}$-PE enabling algorithms $(\mathsf{Eval}_{\mathrm{pk}}, \mathsf{Eval}_{\text{ct-priv}}, \mathsf{Eval}_{\mathrm{sim}})$.

Let all the parameters be defined as in Lemma 5.3 and denote $\hat{\mathcal{C}}$ as the set $\{\hat{C}_{\mathsf{Y}} \mid \hat{C}_{\mathsf{Y}} \leftarrow \mathsf{EncPred}_{\mathsf{Lin}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}), \forall \mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}\}$. The three algorithms are defined as follows:[15]

$\mathsf{Eval}_{\mathrm{pk}}\big(\hat{C}_{\mathsf{Y}} \in \hat{\mathcal{C}},\ \mathbf{B}_0, (\mathbf{B}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\big)$: It outputs

$$\mathbf{B}_{\mathsf{Y}} = \mathbf{B}_0 \cdot \mathbf{G}^{-1}(D\mathbf{G}) - \sum_{i=1}^{D}\sum_{j=1}^{\ell}\sum_{w=1}^{L} a_{i,j,w} \cdot \mathbf{B}_{i,j,w} \in \mathbb{Z}_q^{n\times m}.$$

$\mathsf{Eval}_{\text{ct-priv}}\Big(\hat{C}_{\mathsf{Y}} \in \hat{\mathcal{C}},\ \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big)$: It outputs

$$\mathbf{c} = \big(\mathbf{G}^{-1}(D\mathbf{G})\big)^{\top}\mathbf{c}_0 - \sum_{i=1}^{D}\sum_{j=1}^{\ell}\sum_{w=1}^{L} a_{i,j,w} \cdot \mathbf{c}_{i,j,w} \in \mathbb{Z}_q^{m}.$$

$\mathsf{Eval}_{\mathrm{sim}}\Big(\hat{C}_{\mathsf{Y}} \in \hat{\mathcal{C}},\ \mathbf{R}_0, (\mathbf{R}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big)$: It outputs

$$\mathbf{R}_{\mathsf{Y}} = \mathbf{R}_0 \cdot \mathbf{G}^{-1}(D\mathbf{G}) - \sum_{i=1}^{D}\sum_{j=1}^{\ell}\sum_{w=1}^{L} a_{i,j,w} \cdot \mathbf{R}_{i,j,w} \in \mathbb{Z}_q^{n\times m}.$$

Lemma 5.6. *The above algorithms* $(\mathsf{Eval}_{\mathrm{pk}}, \mathsf{Eval}_{\text{ct-priv}}, \mathsf{Eval}_{\mathrm{sim}})$ *are* $\alpha_{\hat{\mathcal{C}}}$*-PE enabling algorithms for the family of arithmetic circuits* $\hat{\mathcal{C}}$ *defined by the predicate encoding scheme* $\mathsf{PES}_{\mathsf{Lin}}$ *for the* $\mathsf{MultD\text{-}Eq}$ *predicates defined over* $\mathbb{Z}_p^{D\times\ell}$*, where* $\alpha_{\hat{\mathcal{C}}} = C\cdot\max\{m\sqrt{m/n}, \sqrt{D\ell pm}\}$ *for some absolute constant* $C > 0$.

Proof. We check that all the requirements of Definition 5.11 are satisfied. First, by the property of the gadget matrix $\mathbf{G}$ and the fact that $a_{i,j,w} \in \{-1, 0, 1\}$, we have $a_{i,j,w}\mathbf{I}_m = \mathbf{G}^{-1}(a_{i,j,w}\cdot\mathbf{G})$. Then, by plugging in $\mathbf{B}_0 = \mathbf{A}\mathbf{R}_0 - \mathbf{G}$ and $\mathbf{B}_{i,j,w} = \mathbf{A}\mathbf{R}_{i,j,w} - \hat{\mathsf{X}}_{i,j,w}\mathbf{G}$, we can see that $\mathbf{B}_{\mathsf{Y}} = \mathbf{A}\mathbf{R}_{\mathsf{Y}} - \hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}})\mathbf{G}$. Hence, Requirement 1 holds. Furthermore, simple calculation shows that in case $\mathbf{c}_0 = (\mathbf{B}_0 + \mathbf{G})^{\top}\mathbf{s} + \mathbf{z}_0$ and $\mathbf{c}_{i,j,w} = (\mathbf{B}_{i,j,w} + \hat{\mathsf{X}}_{i,j,w}\mathbf{G})^{\top}\mathbf{s} + \mathbf{z}_{i,j,w}$, we have

$$\mathbf{c} = (\mathbf{B}_{\mathsf{Y}} + \hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}})\mathbf{G})^{\top}\mathbf{s} + \Big(\underbrace{\big(\mathbf{G}^{-1}(D\mathbf{G})\big)^{\top}\mathbf{z}_0}_{:=\mathbf{e}_1\ \text{(noise)}} - \underbrace{\sum_{i=1}^{D}\sum_{j=1}^{\ell}\sum_{w=1}^{L} a_{i,j,w}\cdot\mathbf{z}_{i,j,w}}_{:=\mathbf{e}_2\ \text{(noise)}}\Big). \tag{5.21}$$

Recall that the discrete Gaussian distribution $D_{\mathbb{Z}^m,\beta}$ is subgaussian with parameter $C\cdot\beta$ for some absolute constant C. In the following, with an abuse of notation, we will denote any absolute constant as C. Then by the property of $\mathbf{G}^{-1}$, we can use Lemma 2.10 with $B = C\beta$ and $\ell = 1$ to obtain $\|\mathbf{e}_1\|_2 \le C\cdot\beta m\sqrt{m/n}$. Note that we assume $n|m$ without loss of generality. Next, from Lemma 2.3 and the linearity of subgaussian variables, we have $\|\mathbf{e}_2\|_2 \le C\cdot\sqrt{mD\ell L}\beta$. Combining this together with the fact $L = 2^{\lfloor\log p\rfloor+1}$, we obtain $\|\mathbf{e}_1 - \mathbf{e}_2\|_2 \le C\cdot(m/\sqrt{n} + \sqrt{D\ell p})\cdot\sqrt{m}\beta \le \alpha_{\hat{\mathcal{C}}}\cdot\sqrt{m}\beta$ with all but negligible probability. This shows that Requirement 2 holds.

Finally, we show that Requirement 3 holds. First, since the absolute values of each element is bounded by 1, every entry of $\mathbf{R}_0, \mathbf{R}_{i,j,w}$ are subgaussian variables with parameter C. Then, following a similar argument as above, $\mathbf{R}_{\mathsf{Y}}$ is a subgaussian matrix with parameter $C\cdot(m/\sqrt{n} + \sqrt{D\ell p})\cdot\sqrt{m}$. Then using the Lemma 2.9 of [MP12], we have that $s_1(\mathbf{R}_{\mathsf{Y}}) \le C\cdot(m\sqrt{m/n} + \sqrt{D\ell pm}) \le C\cdot\max\{m\sqrt{m/n}, \sqrt{D\ell pm}\} \le \alpha_{\hat{\mathcal{C}}}$ with all but negligible probability. $\square$

[15] Recall that when we use the notation $(A_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}$, we assume the elements are sorted in the lexicographical order.

5.6.2 Construction

Given $\alpha_{\hat{\mathcal{C}}}$-PE enabling algorithms $(\mathsf{Eval}_{\rm pk}, \mathsf{Eval}_{\rm ct\text{-}priv}, \mathsf{Eval}_{\rm sim})$ for a family of arithmetic circuits defined by the predicate encoding scheme $\mathsf{PES}_{\mathsf{Lin}} = (\mathsf{EncInpt}_{\mathsf{Lin}}, \mathsf{EncPred}_{\mathsf{Lin}})$ for the $\mathsf{MultD\text{-}Eq}$ predicates with compatible domains $(\mathcal{X}, \mathcal{Y})$, we build a predicate encryption scheme for the same family of predicates.

Parameters. In the following, let n, m, q, p, D, ℓ be positive integers such that q is a prime and $q > D$, and let σ, α, α' be positive reals denoting the Gaussian parameters. Furthermore, let $(\mathcal{X}, \mathcal{Y}) \in \mathbb{Z}_p^{D\times\ell} \times \mathbb{Z}_p^{D\times\ell}$ be any compatible domains for the $\mathsf{MultD\text{-}Eq}$ predicates, let $\mathcal{P} = \{\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} : \mathcal{X} \to \{0,1\} \mid \mathsf{Y} \in \mathcal{Y}\}$ be the set of multi-dimensional predicates and $\hat{\mathcal{C}} = \{\hat{C}_{\mathsf{Y}} \mid \hat{C}_{\mathsf{Y}} \leftarrow \mathsf{EncPred}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}), \forall \mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}\}$ be the set of polynomials representing the multi-dimensional predicates. Finally, let $\zeta = \lfloor \log p \rfloor + 1$ and $L = 2^{\zeta}$. Here, we assume that all of the parameters are a function of the security parameter $\lambda \in \mathbb{N}$. We provide a concrete parameter selection of the scheme in Section 5.6.3. The following is our PE scheme.

$\mathsf{Setup}(1^{\lambda})$: It first runs $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$ to obtain $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$. It also picks $\mathbf{u} \leftarrow \mathbb{Z}_q^n$, $\mathbf{B}_0, \mathbf{B}_{i,j,w} \leftarrow \mathbb{Z}_q^{n\times m}$ for $(i,j,w) \in [D]\times[\ell]\times[L]$ and outputs

$$\mathsf{mpk} = \Big(\mathbf{A}, \mathbf{B}_0, (\mathbf{B}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}, \mathbf{u}\Big) \quad \text{and} \quad \mathsf{msk} = \mathbf{T_A}.$$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathsf{MultD\text{-}Eq}_{\mathsf{Y}})$: Given a predicate $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}$ for $\mathsf{Y} \in \mathbb{Z}_p^{D\times\ell}$ as input, it runs $\hat{C}_{\mathsf{Y}} \leftarrow \mathsf{EncPred}_{\mathsf{Lin}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}})$ and computes

$$\mathsf{Eval}_{\rm pk}\Big(\hat{C}_{\mathsf{Y}}, \mathbf{B}_0, (\mathbf{B}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big) \to \mathbf{B}_{\mathsf{Y}} \in \mathbb{Z}_q^{n\times m}.$$

Then, it runs

$$\mathbf{e} \leftarrow \mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_{\mathsf{Y}}, \mathbf{u}, \mathbf{T_A}, \sigma),$$

where $[\mathbf{A}|\mathbf{B}_{\mathsf{Y}}]\mathbf{e} = \mathbf{u} \mod q$, and finally returns $\mathsf{sk}_{\mathsf{Y}} = \mathbf{e} \in \mathbb{Z}^{2m}$.

$\mathsf{Enc}(\mathsf{mpk}, \mathsf{X}, \mathsf{M})$: Given an attribute $\mathsf{X} \in \mathbb{Z}_p^{D\times\ell}$ as input, it first runs $\hat{\mathsf{X}} \leftarrow \mathsf{EncInpt}_{\mathsf{Lin}}(\mathsf{X})$ where $\hat{\mathsf{X}} \in \{0,1\}^{D\ell L}$. Then it samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $z \leftarrow D_{\mathbb{Z},\alpha q}$, $\mathbf{z}, \mathbf{z}_0, \mathbf{z}_{i,j,w} \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ for $(i,j,w) \in [D]\times[\ell]\times[L]$, and computes

$$\mathbf{c}_{\mathsf{X}} = \begin{cases} c &= \mathbf{u}^{\top}\mathbf{s} + z + \mathsf{M}\cdot\lfloor q/2 \rceil, \\ \mathbf{c} &= \mathbf{A}^{\top}\mathbf{s} + \mathbf{z}, \\ \mathbf{c}_0 &= (\mathbf{B}_0 + \mathbf{G})^{\top}\mathbf{s} + \mathbf{z}_0, \\ \mathbf{c}_{i,j,w} &= \big(\mathbf{B}_{i,j,w} + \hat{\mathsf{X}}_{i,j,w}\mathbf{G}\big)^{\top}\mathbf{s} + \mathbf{z}_{i,j,w} \quad \text{for} \quad (i,j,w) \in [D]\times[\ell]\times[L], \end{cases}$$

where $\hat{\mathsf{X}}_{i,j,w}$ is the (i,j,w)-th element of $\hat{\mathsf{X}}$. Finally, it returns the ciphertext $\mathbf{c}_{\mathsf{X}} \in \mathbb{Z}_q \times (\mathbb{Z}_q^m)^{D\ell L+2}$.

$\mathsf{Dec}(\mathsf{mpk}, (\hat{C}_{\mathsf{Y}}, \mathsf{sk}_{\mathsf{Y}}), \mathbf{c}_{\mathsf{X}})$: To decrypt the ciphertext $\mathbf{c}_{\mathsf{X}} = (c, \mathbf{c}, \mathbf{c}_0, (\mathbf{c}_{i,j,w}))$ given a predicate and a secret key $(\hat{C}_{\mathsf{Y}}, \mathsf{sk}_{\mathsf{Y}})$, it computes

$$\mathsf{Eval}_{\rm ct\text{-}priv}\Big(\hat{C}_{\mathsf{Y}}, \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big) \to \bar{\mathbf{c}} \in \mathbb{Z}_q^m.$$

Then using the secret key $\mathsf{sk}_{\mathsf{Y}} = \mathbf{e} \in \mathbb{Z}^{2m}$, it computes

$$d = c - [\mathbf{c}^{\top}|\bar{\mathbf{c}}^{\top}]^{\top}\mathbf{e} \in \mathbb{Z}_q.$$

Finally, it returns $|d - \lfloor q/2 \rceil| < q/4$ and 0 otherwise.

5.6.3 Correctness and Parameter Selection

Lemma 5.7 (correctness)**.** *If the predicate is satisfied, assuming $\alpha' > \alpha$, the error term on the decrypted values are bounded by $O(\sqrt{m}\alpha'\alpha_{\hat{C}}\sigma q)$ with overwhelming probability.*

Proof. By the definition of $\alpha_{\hat{C}}$-PE enabling algorithms, when the cryptosystem is operated as specified, we have during decryption

$$d = c - [\mathbf{c}^\top | \bar{\mathbf{c}}^\top]^\top \mathbf{e} = \mathsf{M} \cdot \lfloor q/2 \rceil + z - (\mathbf{z}_0 + \mathbf{z})^\top \mathbf{e},$$

where $\mathbf{z}$ is defined as in Eq. (5.21) (i.e., $\mathbf{z} := \mathbf{e}_1 - \mathbf{e}_2$). Further, we have the following upper bound on the noise.

$$\begin{aligned} \|z - (\mathbf{z}_0 + \mathbf{z})^\top \mathbf{e}\|_2 &\le |z| + (\|\mathbf{z}_0^\top \mathbf{e}\|_2 + \|\mathbf{z}^\top \mathbf{e}\|_2) \\ &\le \sqrt{m}\alpha q + 2\sqrt{m}\alpha'\alpha_{\hat{C}}\sigma q \\ &= O(\sqrt{m}\alpha'\alpha_{\hat{C}}\sigma q). \end{aligned}$$

The first inequality follows from the CauchySchwarz inequality and the second inequality follows from Lemma 2.3, Requirement 2 of the $\alpha_{\hat{C}}$-PE enabling algorithms and the linearity of subgaussian variables. □

Parameter selection. To satisfy the correctness requirement and make the security proof follow through, we need the following:

- the error term is less than $q/5$ with overwhelming probability (i.e., $O(\sqrt{m}\alpha'\alpha_{\hat{C}}\sigma q) < q/5$. See Lemma 5.7),
- the correctness of $\mathsf{PES}_{\mathsf{Lin}}$ holds. (i.e., $q > D$. See Lemma 5.3),
- the $\mathsf{TrapGen}$ algorithm works as specified during Setup. (i.e., $m \ge 2n\lceil \log q \rceil$. See Lemma 2.12),
- the leftover hash lemma can be applied in the security proof (i.e., $m > (n+1)\log q + \omega(\log n)$. See. Lemma 2.11),
- the $\mathsf{SampleLeft}$ algorithm works as specified during KeyGen. (i.e., $\sigma > \|\mathbf{T}_\mathbf{A}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m})$. See Lemma 2.12),
- the $\mathsf{SampleRight}$ algorithm in the security proof works as specified. (i.e., $\sigma > s_1(\mathbf{R}_\mathsf{Y}) \cdot \|\mathbf{T}_\mathbf{G}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m}) \Leftrightarrow \sigma > \alpha_{\hat{C}} \cdot \omega(\sqrt{\log m})$. See Lemma 2.12, Definition 5.11),
- the ReRand algorithm in the security proof works as specified (i.e., $\alpha'/2\alpha > s_1(\mathbf{R}^*)$, $\alpha q > \omega(\sqrt{\log mD\ell L})$ where $\mathbf{R}^*$ is defined as in Game_4. See Lemma 2.6),
- the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{n}$).

To satisfy the above requirements, one way to set the parameters are as follows:

$$\begin{aligned} &m = O(n \log q), && q = \sqrt{m} \cdot (\sqrt{D\ell p})^{-1} \cdot \alpha_{\hat{C}}^2 \cdot \omega(\log m), && \sigma = \alpha_{\hat{C}} \cdot \omega(\sqrt{\log m}), \\ &\alpha_{\hat{C}} = O(\max\{m\sqrt{\log q}, \sqrt{D\ell pm}\}) && \alpha = (\sqrt{D\ell p}) \cdot \alpha_{\hat{C}}^{-2} \cdot \omega(\log m)^{-1}, && \alpha' = O(\sqrt{D\ell pm} \cdot \alpha), \end{aligned}$$

and round up q to the nearest larger prime. Here, D, ℓ, p are chosen accordingly to the types of $\mathsf{MultD\text{-}Eq}$ predicates one wants to use.

5.6.4 Security Proof

Theorem 5.5. *Given the PE enabling algorithms* $(\mathsf{Eval}_{\mathrm{pk}}, \mathsf{Eval}_{\text{ct-priv}}, \mathsf{Eval}_{\mathrm{sim}})$ *for the family of arithmetic circuits* $\hat{\mathcal{C}}$ *defined above, our predicate encryption scheme is selectively secure and weakly attribute hiding with respect to the* $\mathsf{MultD\text{-}Eq}$ *predicates, assuming the hardness of* $\mathsf{LWE}_{n,m+1,q,D_{\mathbb{Z},\alpha q}}$.

Proof. The proof proceeds in a sequence of games where the first game is identical to the real predicate encryption security game from Definition 5.2. In the last game in the sequence, the adversary has zero advantage. In the following, let $\mathcal{A}$ be a PPT adversary that breaks the security of the scheme with advantage ϵ, and define E_i to be the event that $\mathcal{A}$ wins in Game_i.

Game_0 : This is the real security game between the attacker $\mathcal{A}$ against our scheme. By definition, we have $|\Pr[\mathsf{E}_0] - 1/2| = \epsilon$. In the following, let $\mathsf{X}^* \in \mathbb{Z}_p^{D\times \ell}$ denote the challenge attribute $\mathcal{A}$ submits.

Game_1 : In this game, we change the way $\mathbf{B}_0, \mathbf{B}_{i,j,w}$ are chosen. At the beginning of the game, the challenger samples $\mathbf{R}_0, \mathbf{R}_{i,j,w} \leftarrow \{-1,1\}^{m\times m}$ for $(i,j,w) \in [D] \times [\ell] \times [L]$. Then, we define $\mathbf{B}_0$ and $\mathbf{B}_{i,j,w}$ as

$$\mathbf{B}_0 = \mathbf{A}\mathbf{R}_0 - \mathbf{G} \quad \text{and} \quad \mathbf{B}_{i,j,w} = \mathbf{A}\mathbf{R}_{i,j,w} - \hat{\mathsf{X}}^*_{i,j,w}\mathbf{G}, \tag{5.22}$$

where $\hat{\mathsf{X}}^* \leftarrow \mathsf{EncInpt}_{\mathsf{Lin}}(\mathsf{X}^*)$ and $\hat{\mathsf{X}}^*_{i,j,w}$ is the (i,j,w)-th element of $\hat{\mathsf{X}}^* \in \{0,1\}^{D\ell L}$. By Lemma 2.11, the distributions

$$\Big(\mathbf{A}, \mathbf{B}_0, (\mathbf{B}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big) \quad \text{and} \quad \Big(\mathbf{A}, \mathbf{A}\mathbf{R}_0, (\mathbf{A}\mathbf{R}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big)$$

are negligibly close, where $\mathbf{B}_0, \mathbf{B}_{i,j,w} \leftarrow \mathbb{Z}_q^{n\times m}$. Therefore, we have

$$|\Pr[\mathsf{E}_0] - \Pr[\mathsf{E}_1]| = \mathsf{negl}(\lambda).$$

Before continuing to our next game, we make the following observation. From Requirement 1 of Definition 5.11, for all $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}$ and $\hat{C}_{\mathsf{Y}} \leftarrow \mathsf{EncPred}_{\mathsf{Lin}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}})$, we have

$$\mathsf{Eval}_{\mathrm{pk}}\Big(\hat{C}_{\mathsf{Y}}, (\mathbf{A}\mathbf{R}_0 - \mathbf{G}), (\mathbf{A}\mathbf{R}_{i,j,w} - \hat{\mathsf{X}}^*_{i,j,w}\mathbf{G})_{(i,j,w)}\Big) = \mathbf{A}\mathbf{R}_{\mathsf{Y}} - \hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}}^*)\mathbf{G}$$

where $\mathbf{R}_{\mathsf{Y}} = \mathsf{Eval}_{\mathrm{sim}}(\hat{C}_{\mathsf{Y}}, \mathbf{R}_0, (\mathbf{R}_{i,j,w})_{(i,j,w)})$. Furthermore, we have $\|\mathbf{R}_{\mathsf{Y}}\|_2 < \alpha_{\hat{\mathcal{C}}} < \sigma$ from Requirement 3 and our parameter selection. Now by the correctness of the $\mathsf{PES}_{\mathsf{Lin}}$ scheme (Lemma 5.3), we have

$$\mathbf{A}\mathbf{R}_{\mathsf{Y}} - \hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}})\mathbf{G} = \begin{cases} \mathbf{A}\mathbf{R}_{\mathsf{Y}} & \text{if} \quad \mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}^*) = 1 \\ \mathbf{A}\mathbf{R}_{\mathsf{Y}} - t\mathbf{G} \quad \text{for} \quad \exists t \in \{1, \cdots D\} & \text{if} \quad \mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}^*) = 0 \end{cases}. \tag{5.23}$$

Note that since $q > D$ and q a prime, t is always invertible in $\mathbb{Z}_q$.

Game_2 : In this game, we change how $\mathbf{A}$ is sampled. Namely, in Game_2, we generate $\mathbf{A}$ as a random matrix in $\mathbb{Z}_q^{n\times m}$ instead of generating it with a trapdoor. By Lemma 2.12, this makes only negligible difference. To respond to a key extraction query for a predicate $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}} \in \mathcal{P}$ made by $\mathcal{A}$, it first runs $\hat{C}_{\mathsf{Y}} \leftarrow \mathsf{EncPred}_{\mathsf{Lin}}(\mathsf{MultD\text{-}Eq}_{\mathsf{Y}})$ and computes

$$\mathsf{Eval}_{\mathrm{sim}}\Big(\hat{C}_{\mathsf{Y}}, \mathbf{R}_0, (\mathbf{R}_{i,j,w})_{(i,j,w)\in[D]\times[\ell]\times[L]}\Big) \to \mathbf{R}_{\mathsf{Y}}.$$

If $\mathcal{A}$ is a valid adversary, then all the predicates submitted as a key extraction query satisfies $\mathsf{MultD\text{-}Eq}_{\mathsf{Y}}(\mathsf{X}^*) = 0$, and by Eq. (5.23) we have $t = \hat{C}_{\mathsf{Y}}(\hat{\mathsf{X}}^*)$ for some invertible element $t \in \{1, \cdots, D\} \subset \mathbb{Z}_q$. Then, using the SampleRight algorithm from Lemma 2.12, it samples the secret key

$$\mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{\mathsf{Y}}, t, \mathbf{u}, \mathbf{T}_{\mathbf{G}}, \sigma) \to \mathbf{e}.$$

Note that in the previous game the key was sampled as

$$\mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}_{\mathsf{Y}}, \mathbf{u}, \mathbf{T}_{\mathbf{A}}, \sigma) \to \mathbf{e}.$$

By the definition of $\mathsf{Eval}_{\mathrm{sim}}, \mathsf{SampleRight}, \mathsf{SampleLeft}$ and for our choice of σ, the distribution of the secret key is negligibly close to the distribution of that in the previous game. Therefore, the above alters the view of $\mathcal{A}$ only negligibly. Thus, we have

$$|\Pr[\mathsf{E}_1] - \Pr[\mathsf{E}_2]| = \mathsf{negl}(\lambda).$$

Game_3 : In this game, we change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. Recall in the previous games when $\mathsf{coin} = 0$, we created a valid challenge ciphertext as in the real scheme. If $\mathsf{coin} = 0$, to create the challenge ciphertext, the challenger first picks $\mathbf{s} \leftarrow \mathbb{Z}_q^n, z \leftarrow D_{\mathbb{Z},\alpha q}, \bar{\mathbf{z}} \leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $v = \mathbf{u}^\top \mathbf{s} + z \in \mathbb{Z}_q$ and $\mathbf{v} = \mathbf{A}^\top \mathbf{s} + \bar{\mathbf{z}} \in \mathbb{Z}_q^m$. It then sets $\mathbf{R}^* = [\mathbf{R}_0 | \mathbf{R}_{1,1,1} | \cdots | \mathbf{R}_{D,\ell,L}] \in \mathbb{Z}^{m \times (D\ell L+1)m}$ and runs

$$\mathsf{ReRand}\Big([\mathbf{I}_m | \mathbf{R}^*], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}\Big) \to \mathbf{c}' \in \mathbb{Z}_q^{(D\ell L+2)m}$$

from Lemma 2.6, where $\mathbf{I}_m$ is the identity matrix with size m. Finally, it parses $\mathbf{c}'$ appropriately into $D\ell L + 2$ size m vectors $(\mathbf{c}, \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)})$ and outputs the challenge ciphertext as

$$\Big(c = v + \mathsf{M} \cdot \lfloor q/2 \rceil, \quad \mathbf{c}, \quad \mathbf{c}_0, \quad (\mathbf{c}_{i,j,w})_{(i,j,w) \in [D] \times [\ell] \times [L]}\Big). \tag{5.24}$$

We claim this change alters the view of $\mathcal{A}$ only negligibly. To see this, we apply the noise rerandomization lemma (Lemma 2.6) with $\mathbf{V} = [\mathbf{I}_m | \mathbf{R}^*], \mathbf{b} = \mathbf{A}^\top \mathbf{s}$ and $\mathbf{z} = \bar{\mathbf{z}}$ to obtain that the distribution of $\mathbf{c}'$ is negligibly close to the following:

$$\begin{aligned}
\mathbf{c}'^\top &= \mathbf{s}^\top \mathbf{A} [\mathbf{I}_m | \mathbf{R}^*] + \mathbf{z}'^\top \\
&= \mathbf{s}^\top [\mathbf{A} | \mathbf{A}\mathbf{R}_0 | \mathbf{A}\mathbf{R}_{1,1,1} | \cdots | \mathbf{A}\mathbf{R}_{D,\ell,L}] + \mathbf{z}'^\top \\
&= \mathbf{s}^\top [\mathbf{A} | \mathbf{B}_0 + \mathbf{G} | \mathbf{B}_{1,1,1} + \hat{\mathsf{X}}^*_{1,1,1}\mathbf{G} | \cdots | \mathbf{B}_{D,\ell,L} + \hat{\mathsf{X}}^*_{D,\ell,L}\mathbf{G}] + \mathbf{z}'^\top \in \mathbb{Z}^{(D\ell L+2)m}
\end{aligned}$$

where the last equality follows from Eq. (5.22), and $\mathbf{z}'$ is distributed negligibly close to $D_{\mathbb{Z}^{(D\ell L+2)m},\alpha' q}$. Here, we can apply the noise rerandomization lemma since

$$\alpha'/2\alpha > 20\sqrt{(D\ell L + 3)m} \geq s_1([\mathbf{I}_m | \mathbf{R}^*]),$$

for our parameter selection, where we use Lemma 2.9 for the second inequality. It can be seen that the challenge ciphertext is distributed statistically close to Game_2. Therefore, we may conclude that

$$|\Pr[\mathsf{E}_2] - \Pr[\mathsf{E}_3]| = \mathsf{negl}(\lambda).$$

Game_4 : In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, to create the challenge ciphertext the challenger first picks $w \leftarrow \mathbb{Z}_q, \mathbf{w} \leftarrow \mathbb{Z}_q^m$, $z \leftarrow D_{\mathbb{Z},\alpha q}$, and $\bar{\mathbf{z}} \leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $v = w + z \in \mathbb{Z}_q$ and $\mathbf{v} = \mathbf{w} + \bar{\mathbf{z}} \in \mathbb{Z}_q^m$. It then sets $\mathbf{R}^*$ and runs the ReRand algorithm as in Game_4. Finally, it sets the challenge ciphertext as in Eq. (5.24). We claim that $|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]|$ is negligible assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,D_{\mathbb{Z},\alpha q}}$ problem. To show this, we use $\mathcal{A}$ to construct an LWE adversary $\mathcal{B}$ as follows:

$\mathcal{B}$ is given the problem instance of LWE as $(\mathbf{A}', \mathbf{v}' = \mathbf{w}' + \bar{\mathbf{z}}') \in \mathbb{Z}_q^{n\times(m+1)} \times \mathbb{Z}_q^{m+1}$ where $\bar{\mathbf{z}}' \leftarrow D_{\mathbb{Z}^{m+1},\alpha q}$. The task of $\mathcal{B}$ is to distinguish whether $\mathbf{w}' = \mathbf{A}'^{\top}\mathbf{s}$ for $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ or $\mathbf{w}' \leftarrow \mathbb{Z}_q^{m+1}$. In the following, let the first column of $\mathbf{A}'$ be $\mathbf{u} \in \mathbb{Z}_q^n$ and the remaining columns be $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$. Further, let the first coefficient of $\mathbf{v}'$ be v and the remaining coefficients be $\mathbf{v} \in \mathbb{Z}_q^m$. Using these terms, $\mathcal{B}$ sets the master public keys as in Eq. (5.22). During the game, key extractions queries made by $\mathcal{A}$ are answered as in Game_2 without knowledge of the trapdoor of $\mathbf{A}$. To generate the challenge ciphertext, it first picks $\mathsf{coin} \leftarrow \{0,1\}$. If $\mathsf{coin} = 0$, it generates the challenge ciphertext as in Eq. (5.24) using $v, \mathbf{v}$, and returns it to $\mathcal{A}$. Note that all $\mathcal{B}$ needs to do to generate the ciphertext is to run the ReRand algorithm, which it can do without knowledge of the secret randomness $\mathbf{s}, \bar{\mathbf{z}}'$. If $\mathsf{coin} = 1$, $\mathcal{B}$ returns a random ciphertext. At the end of the game, coin' is defined. Finally, $\mathcal{B}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$ and 0 otherwise. It can be seen that if $\mathbf{A}', \mathbf{v}'$ is a valid LWE sample (i.e., $\mathbf{v}' = \mathbf{A}'^{\top}\mathbf{s}$), the view of the adversary corresponds to Game_3. Otherwise (i.e., $\mathbf{v}' \leftarrow \mathbb{Z}_q^{m+1}$), it corresponds to Game_4. We therefore conclude that assuming the hardness of the $\mathsf{LWE}_{n,m+1,q,\mathbb{Z}_{\alpha q}}$ problem we have

$$|\Pr[\mathsf{E}_3] - \Pr[\mathsf{E}_4]| = \mathsf{negl}(\lambda).$$

Game_5 : In this game, we further change the way the challenge ciphertext is created when $\mathsf{coin} = 0$. If $\mathsf{coin} = 0$, the challenger samples $w \leftarrow \mathbb{Z}_q, \mathbf{w} \leftarrow \mathbb{Z}_q^m$, $z \leftarrow D_{\mathbb{Z},\alpha q}$, $\mathbf{z}' \leftarrow D_{\mathbb{Z}^{(D\ell L+2)m},\alpha' q}$, sets $\mathbf{R}^*$ as in the previous games and computes $v = w + z \in \mathbb{Z}_q$. Then, it computes

$$\mathbf{c}'^{\top} = \mathbf{w}^{\top}[\mathbf{I}_m|\mathbf{R}^*] + \mathbf{z}'^{\top} \in \mathbb{Z}_q^{(D\ell L+2)m},$$

and parses $\mathbf{c}'$ appropriately into $D\ell L + 2$ size m vectors $(\mathbf{c}, \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)})$. Finally, it sets the challenge ciphertext as in Eq. (5.24). Using the same argument we made to move from Game_2 and Game_3 concerning the noise rerandomization lemma, we can check that the above change alters the distribution of the challenge ciphertext only negligibly. Thus, we have

$$|\Pr[\mathsf{E}_4] - \Pr[\mathsf{E}_5]| = \mathsf{negl}(\lambda).$$

Game_6 : In this game, we change the challenge ciphertext to be a random vector, regardless of the value of coin. Namely, the challenger creates the challenge ciphertext $(c, \mathbf{c}, \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)}) \in \mathbb{Z}_q \times \mathbb{Z}_q^{(D\ell L+2)m}$ by properly formatting $(D\ell L + 2)m + 1$ random elements from $\mathbb{Z}_q$. In this game, the value coin is independent from the view of $\mathcal{A}$. Therefore, $\Pr[\mathsf{E}_6] = 1/2$.

It remains to upper bound $|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]|$. Since Game_5 and Game_6 differs only in the creation of the challenge ciphertext when $\mathsf{coin} = 0$, we focus on this case. First, it is easy to see that c is uniformly random over $\mathbb{Z}_q$ and independent of the other terms of the ciphertext in both games. Therefore, we are left to show that the distribution of $\bar{\mathbf{c}} = (\mathbf{c}, \mathbf{c}_0, (\mathbf{c}_{i,j,w})_{(i,j,w)})$

in Game_5 is negligibly close to the uniform distribution over $\mathbb{Z}_q^{(D\ell L+2)m}$. First, observe that the following distributions are negligibly close:

$$(\mathbf{A}, \mathbf{A}\mathbf{R}^*, \mathbf{w}^\top, \mathbf{w}^\top\mathbf{R}^*) \approx (\mathbf{A}, \mathbf{A}', \mathbf{w}^\top, \mathbf{w}'^\top) \approx (\mathbf{A}, \mathbf{A}\mathbf{R}^*, \mathbf{w}^\top, \mathbf{w}'^\top)$$

where $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{A}' \leftarrow \mathbb{Z}_q^{n\times(D\ell L+1)m}$, $\mathbf{R}^* \leftarrow \{-1,1\}^{m\times(D\ell L+1)m}$, $\mathbf{w} \leftarrow \mathbb{Z}_q^m$ and $\mathbf{w}' \leftarrow \mathbb{Z}_q^{(D\ell L+1)m}$. It can be seen that the first and the second distributions are negligibly close, by applying Lemma 2.11 for $[\mathbf{A}^\top|\mathbf{w}]^\top \in \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{R}^*$. It can also be seen that the second and the third distributions are negligibly close, by applying the same lemma for $\mathbf{A}$ and $\mathbf{R}^*$. Adding a noise vector $\mathbf{z}'$ to the above $\mathbf{w}^\top\mathbf{R}^*$ does not change the statistical distance between the distributions. Therefore, we may conclude that

$$|\Pr[\mathsf{E}_5] - \Pr[\mathsf{E}_6]| = \mathsf{negl}(\lambda).$$

Analysis. Combining everything together, we have

$$\begin{aligned}\epsilon = \left|\Pr[\mathsf{E}_0] - \frac{1}{2}\right| &= \left|\sum_{i=0}^{5}(\Pr[\mathsf{E}_i] - \Pr[\mathsf{E}_{i+1}]) + \Pr[\mathsf{E}_6] - \frac{1}{2}\right| \\ &\le \sum_{i=0}^{5}|\Pr[\mathsf{E}_i] - \Pr[\mathsf{E}_{i+1}]| + \left|\Pr[\mathsf{E}_6] - \frac{1}{2}\right| \\ &\le \mathsf{negl}(\lambda).\end{aligned}$$

Therefore, the probability that $\mathcal{A}$ wins Game_0 is negligible. □

Remark 5.4. *As noted in Remark 5.2 and Section 5.4.4, in some cases we can compress the size of the ciphertext by taking advantage of the underlying compatible domains $(\mathcal{X}, \mathcal{Y})$. For example, in case we construct a predicate encryption scheme for the subset conjunction predicate, we can decrease the size of the ciphertext by a factor of D.*

5.7 Other Applications: Improving [Yam17] IBE

In this section, we give an (informal) overview on how to make the identity-based encryption (IBE) scheme of [Yam17] more efficient using the preicate encoding scheme of Eq. (5.3) in Section 5.2. Notably, we are able to lower the approximation factor of the LWE problem from $\tilde{O}(n^{11})$ to $\tilde{O}(n^{5.5})$ by exploiting the additive structure of our embedded polynomial and with some additional techniques concerning random matrices used in our proof of Lemma 2.10. Furthermore, we are able to parallelize the encryption and key generation algorithm, whereas the algorithms of [Yam17] are inherently unparallelizable since they rely heavily on the sequential matrix multiplication technique of [GV15].

Recall that [Yam17] provides a modular construction of IBEs. They first define the notion of *compatible algorithms* for partitioning functions (See Definition 5.9). Then, they propose a generic construction of IBE schemes from a partitioning function with its associating compatible algorithms. In particular, they obtain an IBE scheme by instantiating this framework with the compatible algorithms for the modified admissible hash function $\mathsf{F}_{\mathsf{MAH}}$ (See Definition 5.10). Below, we provide the definition of compatible algorithms.

Definition 5.12. ([Yam17], Definition 8) *We say that the deterministic algorithms* $(\mathsf{Encode}, \mathsf{PubEval}, \mathsf{TrapEval})$ *are δ-compatible with a function family* $\{\mathsf{F} : \mathcal{K} \times \mathcal{X} \to \{0,1\}\}$ *if they are efficient and satisfy the following properties:*

- $\mathsf{Encode}(K \in \mathcal{K}) \to \kappa \in \{0,1\}^u$
- $\mathsf{PubEval}(X \in \mathcal{X}, \{\mathbf{B}_i \in \mathbb{Z}_q^{n\times m}\}_{i\in[u]}) \to \mathbf{B}_X \in \mathbb{Z}_q^{n\times m}$
- $\mathsf{TrapEval}(K \in \mathcal{K}, X \in \mathcal{X}, \mathbf{A} \in \mathbb{Z}_q^{n\times m}, \{\mathbf{R}_i \in \mathbb{Z}^{m\times m}\}_{i\in[u]}) \to \mathbf{R}_X \in \mathbb{Z}^{m\times m}$

We require the following to hold:

$$\mathsf{PubEval}(K, X, \mathbf{A}, \{\mathbf{AR}_i + \kappa_i\mathbf{G}\}_{i\in[u]}) = \mathbf{AR}_X + \mathsf{F}(K,X)\mathbf{G},$$

where $\kappa_i \in \{0,1\}$ is the i-th bit of $\kappa = \mathsf{Encode}(K) \in \{0,1\}^u$. Furthermore, if $\mathbf{R}_i \in \{-1,0,1\}^{m\times m}$ for all $i \in [u]$, we have $\|\mathbf{R}_X\|_\infty \leq \delta$.

At a high level, $\mathsf{PubEval}$ is a public algorithm used to compute the hash of an $\mathsf{ID} \in \mathcal{X}$ and $\mathsf{TrapEval}$ is a secret algorithm used by the simulator to recover the $\mathbf{G}$-trapdoor $\mathbf{R}_X$. Therefore, since $\mathbf{R}_X$ is used as a trapdoor to sample a secret key for user X, the quality of $\mathbf{R}_X$ has a direct effect on the efficiency and required hardness assumption for LWE. In particular, the value of δ has a *quadratic* effect on the approximation factor of the LWE problem used in the underlying IBE scheme. Thus, compatible algorithms for $\mathsf{F}_{\mathsf{MAH}}$ with a smaller δ will directly yield a more efficient IBE construction.

In thier work, they (basically) used Eq. (5.1) to compute $\mathsf{F}_{\mathsf{MAH}}$ (See Eq. (5.13)) and obtained δ-compatible algorithms $(\mathsf{Encode}_{\mathsf{Yam}}, \mathsf{PubEval}_{\mathsf{Yam}}, \mathsf{TrapEval}_{\mathsf{Yam}})$ for $\mathsf{F}_{\mathsf{MAH}}$ where $\delta = \tilde{O}(\lambda^4)$, which can be obtained by plugging in the values from Theorem 5.1. Furthermore, due to the multiplicative structure of Eq. (5.1), they heavily rely on the sequential matrix multiplication technique of [GV15] in order to control the growth of δ. This is the reason why their scheme is inherently unparallelizable.

We provide two ideas to improve their scheme. First, we can do much better by using Eq. (5.3) to compute $\mathsf{F}_{\mathsf{MAH}}$. Namely, we use the following polynomial defined over $\mathbb{Z}_q$, which is a slight modification of Eq. (5.3):

$$\eta - \sum_{i=1}^{\eta}\sum_{j=1}^{\ell}\prod_{k=1}^{\zeta}\Big((1-s_{j,k}) + (-1+2s_{j,k})\cdot t_{i,k}\Big) = \begin{cases} 0 & \text{if } \mathsf{T} \subseteq \mathsf{S}(X) \\ \in \{1,\cdots\eta\} & \text{if } \mathsf{T} \not\subseteq \mathsf{S}(\mathsf{X}) \end{cases}, \tag{5.25}$$

A subtle point here is that we have to alter Definition 5.12 so that the function family can take output over $\mathbb{Z}_q$. Recall that in the above we required $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) \in \{0,1\}$. However, we can easily show that for the security proof for the IBE scheme to follow through, we do not necessarily need the output to be in $\{0,1\}$, as long as we have $\mathsf{F}_{\mathsf{MAH}}(\mathsf{T}, X) = 0$ iff $\mathsf{T} \subseteq \mathsf{S}(X)$.

For completeness, we provide the algorithms for $\mathsf{PubEval}, \mathsf{TrapEval}$ following the notations of [Yam17], Section 5.1. The Encode algorithm is defined as in [Yam17].

$\mathsf{PubEval}(X, \{\mathbf{B}_{i,k}\}_{(i,k)\in[\eta]\times[\zeta]} \in \mathbb{Z}_q^{n\times m})$: It first computes $S(X) = \{s_1, \cdots, s_\ell\} \subset [2\ell]$. Let $s_{j,k} \in \{0,1\}$ be the k-th bit of the binary representation of s_j. It then proceeds as follows:

1. For $(i,j,k) \in [\eta]\times[\ell]\times[\zeta]$, it sets $\mathbf{V}_{i,j,k} = (1-s_{j,k})\cdot\mathbf{G} + (-1+2s_{j,k})\cdot\mathbf{B}_{i,k}$
2. For $(i,j) \in [\eta]\times[\ell]$, set $\mathbf{V}_{i,j,[1:1]} := \mathbf{V}_{i,j,1}$ and compute $\mathbf{V}_{i,j,[1:k+1]} := \mathbf{V}_{i,j,k+1} \cdot \mathbf{G}^{-1}(\mathbf{V}_{i,j,[1:k]})$ for $k \in [\zeta-1]$. Then set $\mathbf{V}_{i,j} = \mathbf{V}_{i,j,[1:\zeta]}$.

3. Finally, it outputs $\mathbf{B}_X = \eta \cdot \mathbf{G} - \sum_{i \in [\eta]} \sum_{j \in [\ell]} \mathbf{V}_{i,j}$

$\mathsf{TrapEval}(\mathbf{T}, X, \mathbf{A}, \{\mathbf{R}_{i,k}\}_{(i,k)\in[\eta]\times[\zeta]} \in \mathbb{Z}_q^{n\times m})$: It first computes $S(X) = \{s_1, \cdots, s_\ell\} \subset [2\ell]$ and parses $\mathbf{T} \to (t_1, \cdots, t_{\eta'}) \subset [2\ell]$, where $\eta' < \eta$. It then sets $t_{\eta'+1} = \cdots = t_\eta = 0$. In the following, let $t_{i,k}, s_{j,k} \in \{0,1\}$ be the k-th bit of the binary representation of t_i, s_j, respectively. It then proceeds as follows:

1. For $(i,j,k) \in [\eta] \times [\ell] \times [\zeta]$, it sets $\mathbf{S}_{i,j,k} = (-1 + 2s_{j,k}) \cdot \mathbf{R}_{i,k}$
2. For $(i,j) \in [\eta]\times[\ell]$, set $\mathbf{S}_{i,j,[1:1]} := \mathbf{S}_{i,j,1}$ and compute $\mathbf{S}_{i,j,[1:k+1]} := \mathbf{R}_{i,k+1} \cdot \mathbf{G}^{-1}(\mathbf{V}_{i,j,[1:k]}) + (-1 + 2s_{j,k+1}) \cdot \mathbf{S}_{i,j,[1:k]}$ for $k \in [\zeta - 1]$, where $\mathbf{V}_{i,j,[1:k]}$ is defined as above. Then set $\mathbf{S}_{i,j} = \mathbf{S}_{i,j,[1:\zeta]}$.
3. Finally, it outputs $\mathbf{R}_X = \sum_{i\in[\eta]} \sum_{j\in[\ell]} \mathbf{S}_{i,j}$

Lemma 5.8. *The above algorithms* (Encode, PubEval, TrapEval) *are $m\zeta\eta\ell$-compatible algorithms for $\mathsf{F}_{\mathsf{MAH}}$. In particular, if we instantiate $\mathsf{F}_{\mathsf{MAH}}$ using Definition 5.10, $m\zeta\eta\ell = \tilde{O}(\lambda^2)$.*

Proof. First observe the inequality

$$\begin{aligned}
\|\mathbf{S}_{i,j,[k+1]}\|_\infty &= \|\mathbf{R}_{i,k+1} \cdot \mathbf{G}^{-1}(\mathbf{V}_{i,j,[1:k]}) + (-1 + 2s_{j,k+1}) \cdot \mathbf{S}_{i,j,[1:k]}\|_\infty \\
&\le m \cdot \|\mathbf{R}_{i,k+1}\|_\infty + \|\mathbf{S}_{i,j,[1:k]}\|_\infty \\
&\le m + \|\mathbf{S}_{i,j,[1:k]}\|_\infty,
\end{aligned}$$

where we use $(-1 + 2s_{j,k+1}) \in \{0,1\}, \mathbf{R}_{i,k+1} \in \{-1,0,1\}^{m\times m}$. Therefore by induction, we have $\|\mathbf{S}_{i,j}\|_\infty \le m\zeta$. Hence, we obtain the bound. □

Note that the poly-log factor hidden in the $\tilde{O}(\cdot)$ notation is the same as [Yam17]. Our δ is a $O(\lambda^2)$ factor smaller than the scheme of [Yam17], and since δ has a quadratic effect on the approximation factor of the LWE problem, we are able to lower the approximation factor down by $O(\lambda^4)$. Finally, we make the following subtle observations:

- Using subgaussian arguments, the error term can be bounded by $O(\alpha'\sigma\sqrt{mq})$ instead of $O(\alpha'\sigma mq)$. (See [Yam17], Lemma 13).

- Since $\mathbf{R}_X \in \mathbb{Z}^{m\times m}$ is subgaussian with parameter δ (which follows from $\|\mathbf{R}\|_\infty \le \delta$), we have $s_1(\mathbf{R}_X) \le C \cdot \sqrt{m}\delta$ with overwhelming probability for some positive constant C ([MP12], Lemma 2.9). Therefore, we can use $\alpha' > O(\alpha\sqrt{m}\delta)$, instead of $\alpha' > O(\alpha m\delta)$. (See [Yam17], Section 6.2).

- Use the sampling algorithm of [MP12] to obtain $\sigma > \tilde{\Omega}(\sqrt{m}\delta)$ instead of $\sigma > \tilde{\Omega}(m\delta)$ (See. [Yam17], Lemma 3).

Combining this together, we obtain a candidate parameter selection as follows:

$$\begin{aligned}
&m = O(n \log q), \qquad && q = n^2 \cdot \delta^2 \cdot \omega(\log^2 n), \qquad && \sigma = m \cdot \delta \cdot \omega(\sqrt{\log m}), \\
&\alpha q = 3\sqrt{n} && \alpha' q = 5\sqrt{n} \cdot m \cdot \delta.
\end{aligned}$$

Plugging in our δ-compatible algorithm for the $\mathsf{F}_{\mathsf{MAH}}$ function, we obtain an approximation factor of $\tilde{O}(n^{5.5})$ for the LWE problem. Recall that the approximation factor of [Yam17] was $\tilde{O}(n^{11})$. Finally, we are also able to improve significantly on the parallel complexity of the IBE scheme. Notably, our compatible algorithms (Encode, PubEval, TrapEval) for the modified admissible hash function $\mathsf{F}_{\mathsf{MAH}}$ allows for high parallelization of the encryption and key generation algorithm. (Recall $\mathsf{PubEval}$ is used to compute the hash of an $\mathsf{ID} \in \mathcal{X}$.) We obtain parallel speed up because our encoded polynomial of $\mathsf{F}_{\mathsf{MAH}}$ has an additive structure, and we do not have to rely on the sequential matrix multiplication technique of [GV15] to control the growth of $\mathbf{R}_X$.

Chapter 6

Non-Zero Inner Product Encryption Schemes from Various Assumptions

6.1 Introduction

An attribute-based encryption (ABE) scheme is an advanced form of public key encryption where an access control over encrypted data is possible. In an ABE scheme, a ciphertext and a secret key are associated with attributes X and Y, respectively, and the decryption is possible only when they satisfy $R(X, Y) = 1$ for a certain relation R. The concept of ABE was first proposed by Sahai and Waters [SW05]. Since then, many study followed in order to improve the scheme in many aspects: security [LOS+10, OT10], expressibility [GPSW06, LW11, GVW13], and efficiency [ALDP11]. While the early constructions of ABE schemes are based on bilinear maps, some of the more recent schemes are based on lattices.

In this chapter, we focus on a special form of an ABE scheme called non-zero inner product encryption (NIPE) scheme. In an NIPE scheme, a ciphertext attribute is a vector $\vec{x}$ and a secret key attribute is a vector $\vec{y}$, and the relation is defined as $R(\vec{x}, \vec{y}) = 1$ iff $\langle \vec{x}, \vec{y} \rangle \neq 0$. The notion of NIPE was first introduced in [KSW08]. It was not until Attrapadung and Libert [AL10] who gave the first construction of an NIPE scheme using bilinear maps. In their work, they provided interesting applications of NIPE schemes such as identity-based revocation (IBR) schemes, where an IBR scheme is a type of broadcast encryption scheme that allows for efficient revocation of small member size. Since then, many efficient NIPE schemes have been proposed [AL10, ALDP11, OT10, OT15, YAHK14, CW14, CLR16]. They are all based on number theoretic assumptions on bilinear maps.

On the other hand, the constructions of NIPE schemes without bilinear maps are much less investigated. The only known other constructions are based on lattices. However, unlike in the bilinear map setting, we do not know of any direct constructions of a NIPE scheme in the lattice setting. In more detail, we have ABE schemes for any circuit (i.e. the relation R being general circuits) [GVW13, BGG+14b] and any branching programs [GVW13, GV15] from the learning with errors (LWE) assumption. Here, the expressibility of the latter constructions are more limited, however, these schemes can be proven secure under the LWE assumption with polynomial approximation factors unlike the former schemes that require sub-exponential approximation factors, i.e., the required hardness assumption is much weaker. Although we have two lines of works

[0]The contents of this chapter is based on the work presented at PKC 2019 under the title "Non-Zero Inner Product Encryption Schemes from Various Assumptions: LWE, DDH and DCR" [KY19b].

that allow us to indirectly construct lattice-based NIPE schemes, they are both highly inefficient. In particular, we can use the former constructions from circuits to implement an NIPE scheme, however, this would require us to express the computation of the non-zero-inner-product predicates as a circuit, which would result in a highly inefficient scheme. Furthermore, it would require us to base security under a sub-exponential LWE assumption, which is not desirable both from the efficiency and security stand points. Alternatively, we can use the latter construction for branching programs. To do so, we would first represent the non-zero inner product predicate as an NC^1 circuit, which is possible because arithmetic operations are known to be in NC^1 [BCH86], and then convert it into a branching program using the Barrington's theorem. Using [GVW13] or[GV15], the construction by this approach enjoys security from the standard polynomial LWE assumption. However, the approach is still highly inefficient due to the large overhead incurred by the invocation of the Barrington's theorem [Bar89].

More on NIPEs. Although NIPE schemes allows us to construct other cryptographic primitives such as IBR schemes as explained above, it may be more helpful to understand the usefulness of the primitive through its "negating" feature. As the name suggests, NIPE scheme is the counterpart of inner-product encryption (IPE) schemes. It is well known that IPE schemes can be used to construct functional encryption schemes that can handle many practical predicates such as polynomial evaluations, disjunction and/or conjunctions of equality tests, membership tests and so on (for concrete applications see for example [BW07, KSW08]). In brief, NIPE schemes are primitives that can handle the exact opposite of all these predicates. As negating policies are useful in practice, the importance of negated policies in the area of ABE has been highlighted in prior works [OSW07, AL10, ABS17].

Furthermore, aside from its practical interest, NIPE schemes are theoretically interesting in its own right, since as we show as one of our results, NIPE schemes can be constructed from much weaker assumptions than one would expect. In particular, we construct NIPE schemes from the DDH or DCR assumption, whereas it currently seems that stronger assumptions such as the DBDH or DLIN assumption is required to construct IPE schemes. Therefore, although an NIPE scheme may be simply understood as an IPE scheme in the opposite flavor, our result indicates a distinct gap between the two primitives when it comes to concrete constructions. Considering the recent breakthrough in constructing identity-based encryption schemes [DG17] and functional encryption schemes for inner products [ABDCP15, ALS16] from weak assumptions, we hope our work to spark interest to finding the minimum assumption for other ABE-related primitives.

6.1.1 Our Contributions

To remedy our rather poor understanding regarding NIPE schemes without bilinear maps, we provide two methods for constructing NIPE schemes: a direct construction from lattices and a generic construction from functional encryption schemes for inner products (LinFE)[1] . For the first direct lattice-based approach, we propose two NIPE constructions where the differences lie in where the inner products between attribute and predicate vectors are taken. The first scheme is over $\mathbb{Z}$ whereas the second scheme is over $\mathbb{Z}_p$. For the second generic approach, we show how to generically construct NIPE schemes from any LinFE scheme. In particular, we can use the recent works of [ABDCP15, ALS16] to instantiate various types of NIPE schemes. Concretely, since [ALS16] provides us with LinFE schemes from the LWE assumption, the DDH assumption and

[1] The term LinFE is borrowed from [ALS16]. It is named as such, since it is a special type of functional encryption scheme restricted to the class of linear functions.

the DCR assumption, we obtain NIPE schemes secure under all of these assumptions. Notably, we obtain the first NIPE constructions *without* bilinear maps or lattices.

We give a brief overview on the properties that our NIPE schemes satisfy. As for the first direct approach, we obtain two NIPE schemes with different properties: a selectively secure *stateless* NIPE scheme over $\mathbb{Z}$ and a selectively secure *stateful* NIPE scheme over $\mathbb{Z}_p$. As for the second generic approach, by using the LinFE schemes provided in [ALS16], which subsumes the work of [ABDCP15], we obtain an adaptively secure *stateless* or *stateful* NIPE scheme over $\mathbb{Z}$ or $\mathbb{Z}_p$, depending on what we use as the underlying LinFE scheme. The main advantage of the first approach is that it leads to a more efficient NIPE scheme in the amortized sense compared with the second approach instantiated with a lattice-based LinFE scheme. In more detail, to encrypt a message of ℓ_M-bit length, the first approach requires $(\ell_M + m + m\ell)$ elements of $\mathbb{Z}_q$ in a ciphertext and the second requires $(m + \ell)\ell_M$. Here, ℓ is the dimension of the predicate vectors in the NIPE scheme and q and m are the modulus size and the number of columns of the LWE matrix involved in the scheme, respectively. The first approach is more efficient than the second one when we encrypt more than $m\ell/(m + \ell)$ bits at once. For a natural setting of $\ell < m, \lambda$ where λ is the security parameter, this encompasses the most interesting case of KEM-DEM settings where one encrypts λ bits of session key. In fact, when we are in the ring setting, since m is $O(\log \lambda)$, the first approach will be more efficient regardless of the size ℓ. Furthermore, for NIPE schemes over $\mathbb{Z}_p$, the first approach would require smaller LWE modulus. Indeed, in certain regime of parameters such as $\ell = \log n/ \log\log\log n$ and $p = \log\log n$, the first approach would yield a scheme with polynomial modulus whereas the second requires super-polynomial modulus. However, on the other hand, the advantage of the second approach is that it achieves adaptive security and allows us to instantiate the NIPE scheme with different types of hardness assumptions such as the DDH and DCR assumptions. Below, we give an outline of the techniques we used for constructing lattice-based NIPE schemes and the generic construction of NIPE schemes from LinFE. In particular, we believe the techniques we utilized for the lattice-based direct NIPE construction to be of independent interest.

6.2 Technical Overview

Lattice-Based Constructions. We propose two NIPE schemes built directly from lattices. At a high level, our two NIPE constructions share many similarities; both constructions highly depart from the previous lattice-based ABE constructions [GVW13, BGG+14b, GV15] and they rely heavily on the tools of Gaussian measures over *multi-dimensional lattices* during the security proof. Notably, for both of our constructions: a trapdoor $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m \times m}$ for the public matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is not required, a secret key for a user is simply a linear combination of the master secret keys, and the algorithm $\mathsf{SampleRight}$ of [ABB10] is used during decryption. To the careful readers, this may seem somewhat peculiar, since $\mathsf{SampleRight}$ is an algorithm that customary appears in the security proof for allowing the simulator to sample a short vector $\mathbf{e}$ such that $[\mathbf{A}|\mathbf{B}]\mathbf{e} = \mathbf{u}$ without knowledge of the trapdoor of $\mathbf{A}$, in case $\mathbf{B}$ is in the special form $\mathbf{AR} + t \cdot \mathbf{G} \mod q$, where $t \in \mathbb{Z}_q$ is some invertible element and $\mathbf{G}$ is a special matrix with a publicly known trapdoor $\mathbf{T}_\mathbf{G}$ [MP12].

Below we sketch our construction. We set the master public key MPK and the master secret key MSK as follows:

$$\mathsf{MPK} = (\mathbf{A}, \mathbf{B}_1, \cdots, \mathbf{B}_\ell, \mathbf{u}) \quad \text{and} \quad \mathsf{MSK} = (\mathbf{R}_1, \cdots, \mathbf{R}_\ell),$$

where ℓ denotes the dimension of the vectors, $\{\mathbf{R}_i\}_{i\in[\ell]}$ are random matrices whose columns are sampled from the discrete Gaussian distribution and $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i \mod q$. In the following, we focus on the overview of our first NIPE scheme with inner product space $\mathbb{Z}$. Although the high level construction is the same for our second NIPE scheme with inner product space $\mathbb{Z}_p$, we require some additional technicalities during key generation, which we describe later.

Given the master secret key MSK, our secret key generation algorithm is very simple and does not require any Gaussian sampling as in prior works. Concretely, given a predicate vector $\vec{y} = (y_1, \cdots, y_\ell) \in \mathbb{Z}^\ell$, we simply return $\mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} y_i \mathbf{R}_i \in \mathbb{Z}^{m\times m}$ as the secret key. To embed an attribute vector $\vec{x} = (x_1, \cdots, x_\ell) \in \mathbb{Z}^\ell$ into the ciphertext, we use the techniques of [AFV11, BGG$^+$14b], and create vectors $\{\mathbf{c}_i = \mathbf{s}^\top(\mathbf{B}_i + x_i \cdot \mathbf{G}) + \mathbf{z}_i\}_{i\in[\ell]}$ along with $\mathbf{c}_0 = \mathbf{s}^\top \mathbf{A} + \mathbf{z}_0$. Then, for decryption, a user with predicate vector $\vec{y}$ computes the following:

$$\sum_{i=1}^{\ell} y_i \cdot \mathbf{c}_i = \mathbf{s}^\top(\sum_{i=1}^{\ell} y_i \mathbf{B}_i + \langle \vec{x}, \vec{y}\rangle \cdot \mathbf{G}) + \mathsf{noise} = \mathbf{s}^\top(\mathbf{A}\mathbf{R}_{\vec{y}} + \langle \vec{x}, \vec{y}\rangle \cdot \mathbf{G}) + \mathsf{noise}.$$

Therefore, if $\langle \vec{x}, \vec{y}\rangle \neq 0$ (over $\mathbb{Z}$), we can use the algorithm $\mathsf{SampleRight}$ to sample a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$ such that $[\mathbf{A}|\mathbf{A}\mathbf{R}_{\vec{y}} + \langle \vec{x}, \vec{y}\rangle \cdot \mathbf{G}]\mathbf{e} = \mathbf{u} \mod q$. Here, to take care of the subtle problem that $\langle \vec{x}, \vec{y}\rangle$ has to be invertible over $\mathbb{Z}_q$, we require the attribute and predicate vectors to be in some restricted domains.

However, despite the simplicity of our construction, the security proof requires a rather sensitive and technical analysis that calls new techniques. In particular, building upon the prior works of [BF11], we prepare new tools concerning Gaussian measures over *mulit-dimensional lattices*, which we believe to be of independent interest. Using these tools, we are able to provide a rigorous treatment on the distribution of the secret keys $\mathbf{R}_{\vec{y}}$ of the real world and the simulated world. In more detail, given a challenge attribute $\vec{x}^* \in \mathbb{Z}^\ell$ at the outset of the game, the simulator samples random matrices $\{\mathbf{R}_i^{\mathsf{SIM}}\}_{i\in[\ell]}$ as in the real world and sets the public matrices $\mathbf{B}_i$ as $\mathbf{A}\mathbf{R}_i^{\mathsf{SIM}} - x_i^* \cdot \mathbf{G}$. We answer the secret key queries as in the real world, i.e., given a predicate vector $\vec{y} = (y_1, \cdots, y_\ell) \in \mathbb{Z}^\ell$, we simply return $\mathbf{R}_{\vec{y}}^{\mathsf{SIM}} = \sum_{i=1}^{\ell} y_i \mathbf{R}_i^{\mathsf{SIM}} \in \mathbb{Z}^{m\times m}$. At first glance this seems completely insecure, since an adversary may query $\vec{y} = (1, 0, \cdots, 0) \in \mathbb{Z}^\ell$ and recover $\mathbf{R}_1$ or $\mathbf{R}_1^{\mathsf{SIM}}$ depending on which world it is in. Then, the adversary can check whether $\mathbf{B}_1 = \mathbf{A}\mathbf{R}_1$ or $\mathbf{B}_1 = \mathbf{A}\mathbf{R}_1^{\mathsf{SIM}} - x_1^* \cdot \mathbf{G}$ to distinguish between the real world and the simulated world. However, this seemingly acute tactic cannot be used to attack our NIPE scheme. The main observation is that, if $\vec{y} = (1, 0, \cdots, 0) \in \mathbb{Z}^\ell$ is a valid predicate for the key extraction query, then we must have $\langle \vec{x}^*, \vec{y}\rangle = 0$, or in other words $x_1^* y_1 = x_1^* = 0$. Therefore, since $\mathbf{R}_1$ and $\mathbf{R}_1^{\mathsf{SIM}}$ are distributed statistically close, the above attack cannot be used to distinguish between the two worlds. Our security analysis builds on this idea and proves that the distribution of the secret keys the adversary obtains in the two worlds $\{\mathbf{R}_{\vec{y}^{(j)}}\}_{j\in[Q]}$ and $\{\mathbf{R}_{\vec{y}^{(j)}}^{\mathsf{SIM}}\}_{j\in[Q]}$ are indeed statistically indistinguishable. The main technical contribution is developing new tools for Gaussian measures over multi-dimensional lattices, and analyzing the (set of) linear combinations of Gaussian distributions $\{\mathbf{R}_{\vec{y}^{(j)}} = \sum_{i=1}^{\ell} y_i^{(j)} \mathbf{R}_i\}_{j\in[Q]}$.

Finally, we briefly note on the aforementioned technical issue that arises for our second NIPE construction with inner product space $\mathbb{Z}_p$. Notably, we require our NIPE scheme to be *stateful*. This is similar to an issue that came up in the works of [ALS16] for their LinFE scheme over $\mathbb{Z}_p$. Unlike in the NIPE construction with inner product space $\mathbb{Z}$, the linear dependency of the predicate vectors $\vec{y} \in \mathbb{Z}_p^\ell$ and the secret keys $\mathbf{R}_{\vec{y}} \in \mathbb{Z}^{m\times m}$ are no longer consistent. In other words, even when an adversary queries for secret keys corresponding to predicate vectors that are linearly dependent over $\mathbb{Z}_p$, the corresponding secret keys may no longer be linearly dependent over $\mathbb{Z}$. Therefore, the adversary can recover the full master secret key $\{\mathbf{R}_i\}_{i\in[\ell]}$ by querying the right predicate vectors. To prevent this from happening, we make the key generation algorithm stateful and pay special attention so as not to give out linearly independent secret keys for linearly dependent predicate vectors. In addition, we also specify how to maintain the state in a clever way. This is because the representation of the state has a direct effect on the required LWE assumption, and if we maintain the state naively, we would have to base our security on the subexponential LWE assumption.

Generic Construction from LinFE. Besides the direct constructions from lattices, we also propose a generic construction of a NIPE scheme from a LinFE scheme. The idea for the generic conversion is inspired by the works of [ABP+17] and is surprisingly simple. To explain the idea, let us first recall that in a LinFE scheme, a ciphertext and a private key are associated with vectors $\vec{x}$ and $\vec{y}$, and when we decrypt the ciphertext using the private key, we recover $\langle \vec{x}, \vec{y} \rangle$. Given a LinFE scheme, we construct a NIPE scheme as follows. To encrypt a message M for a vector $\vec{x}$, we encrypt a vector $\mathsf{M} \cdot \vec{x}$ using the underlying LinFE scheme to obtain a ciphertext. A private key for a vector $\vec{y}$ in the NIPE scheme is exactly the same as a private key for $\vec{y}$ in the underlying LinFE scheme. Observe that when we decrypt the ciphertext using the private key, we recover $\langle \mathsf{M} \cdot \vec{x}, \vec{y} \rangle = \mathsf{M} \cdot \langle \vec{x}, \vec{y} \rangle$. This value corresponds to 0 when $\langle \vec{x}, \vec{y} \rangle = 0$ regardless of the value of the message. On the other hand, when $\vec{x}$ and $\vec{y}$ are known, M can be recovered by computing $\mathsf{M} \cdot \langle \vec{x}, \vec{y} \rangle / \langle \vec{x}, \vec{y} \rangle = \mathsf{M}$. That is, the message is recovered if and only if $\langle \vec{x}, \vec{y} \rangle \neq 0$. Indeed, this functionality exactly matches that of NIPE schemes.

While the idea is very simple, it leads to interesting consequences. By applying our LinFE-to-NIPE conversion to existing LinFE constructions [ABDCP15, ALS16], we obtain several new NIPE schemes. Notably, we obtain the first NIPE constructions from the DDH and DCR assumptions. In other words, we obtain NIPE constructions without relying on bilinear maps or lattices. This result may be somewhat surprising, since we do not know any other similar primitives to inner product encryption (IPE)[2] schemes that can be constructed without bilinear maps or lattices. In particular, it was not until recently for even a simple primitive such as an identity-based encryption scheme (in the standard model) to be constructed without relying on bilinear maps or lattices [DG17]. Therefore, our result indicates that NIPE schemes may be a primitive quite different from other ABE type primitives in nature.

6.3 Preparation

6.3.1 Non-Zero Inner Product Encryption

Syntax. Let $\mathcal{P}$ and $\mathcal{I}$ denote the predicate space and attribute space, where the inner product between elements (i.e., vectors) from $\mathcal{P}$ and $\mathcal{I}$ are well-defined. Furthermore, let $\mathcal{S}$ denote the space where the inner product is taken. A *stateful* non-zero inner product encryption (NIPE) scheme over $\mathcal{S}$ consists of the following four algorithms:

$\mathsf{Setup}(1^\lambda, 1^\ell) \to (\mathsf{MPK}, \mathsf{MSK}, \mathsf{st})$: The setup algorithm takes as input a security parameter 1^λ and the length ℓ of the vectors in the predicate and attribute spaces, and outputs a master public key MPK, a master secret key MSK and an initial state st.

$\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \vec{y}) \to (\mathsf{sk}_{\vec{y}}, \mathsf{st})$: The key generation algorithm takes as input the master public key MPK, the master secret key MSK, the state st and a predicate vector $\vec{y} \in \mathcal{P}$. It

[2]IPE is a special kind of ABE where decryption is possible iff the inner product of the vectors corresponding to a ciphertext and a private key is 0. This should not be confused with LinFE, where the decryption is always possible and the decryption result is the inner product itself.

outputs a private key $\mathsf{sk}_{\vec{y}}$ and a updated state st. We assume that $\vec{y}$ is implicitly included in $\mathsf{sk}_{\vec{y}}$.

$\mathsf{Encrypt}(\mathsf{MPK}, \vec{x}, \mathsf{M}) \to \mathsf{ct}$: The encryption algorithm takes as input a master public key MPK, an attribute vector $\vec{x} \in \mathcal{I}$ and a message M. It outputs a ciphertext ct.

$\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, (\vec{x}, \mathsf{ct})) \to \mathsf{M}$ or $\perp$: The decryption algorithm takes as input the master public key MPK, a private key $\mathsf{sk}_{\vec{y}}$, and a ciphertext ct with an associating attribute vector $\vec{x}$. It outputs the message M or $\perp$, which means that the ciphertext is not in a valid form.

Correctness. We require correctness of decryption: that is, for all $\lambda, \ell \in \mathbb{N}$, all $\vec{x} \in \mathcal{I}, \vec{y} \in \mathcal{P}$, and all M in the specified message space, the following holds:

- if $\langle \vec{x}, \vec{y} \rangle \neq 0$, then $\Pr[\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \mathsf{M}] = 1 - \mathsf{negl}(\lambda)$

- if $\langle \vec{x}, \vec{y} \rangle = 0$, then $\Pr[\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \perp] = 1 - \mathsf{negl}(\lambda)$,

where the inner products are taken over $\mathcal{S}$ and the probability is taken over the randomness used in all the algorithms.

We also define a *stateless* non-zero inner product encryption, where we do not require any state information in the above algorithms.

Security. We define the security of a (stateful) NIPE scheme over $\mathcal{S}$ with predicate space $\mathcal{P}$ and attribute space $\mathcal{I}$ by the following game between a challenger and an adversary $\mathcal{A}$.

- Setup. At the outset of the game, the challenger runs $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives the public parameter MPK to $\mathcal{A}$.

- Phase 1. $\mathcal{A}$ may adaptively make key-extraction queries. If $\mathcal{A}$ submits a predicate vector $\vec{y} \in \mathcal{P}$ to the challenger, the challenger runs $(\mathsf{sk}_{\vec{y}}, \mathsf{st}) \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \vec{y})$ and returns $\mathsf{sk}_{\vec{y}}$.

- Challenge Phase. At some point, $\mathcal{A}$ outputs messages $\mathsf{M}_0, \mathsf{M}_1$ and an attribute vector $\vec{x}^* \in \mathcal{I}$ on which it wishes to be challenged, with the restriction that $\langle \vec{x}^*, \vec{y} \rangle = 0$ (over $\mathcal{S}$) for all $\vec{y}$ queried during Phase 1. Then, the challenger picks a random bit $b \in \{0, 1\}$ and returns $C^* \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ to $\mathcal{A}$.

- Phase 2. After the challenge query, $\mathcal{A}$ may continue to make key-extraction queries for predicate vectors $\vec{y} \in \mathcal{P}$, with the added restriction that $\langle \vec{x}^*, \vec{y} \rangle = 0$ (over $\mathcal{S}$).

- Guess. Finally, $\mathcal{A}$ outputs a guess b' for b.

The advantage of $\mathcal{A}$ is defined as $\mathsf{Adv}^{\mathsf{NIPE}}_{\mathcal{A},\mathcal{S}} = \left|\Pr[b' = b] - \frac{1}{2}\right|$. We say that a stateful NIPE scheme with inner product space $\mathcal{S}$ is *adaptively secure*, if the advantage of any PPT $\mathcal{A}$ is negligible. Similarly, we define *selective security* for a stateful NIPE scheme with inner product space $\mathcal{S}$, by modifying the above game so that the adversary $\mathcal{A}$ is forced to declare its challenge attribute vector $\vec{x}^*$ before **Setup**. Therefore, we also add the restriction that $\langle \vec{x}^*, \vec{y} \rangle = 0$ (over $\mathcal{S}$) during **Phase 1**. Finally, we define an analogous security notion for stateless NIPE schemes, where we do not require any state information during the above game.

Remark on the Security Model. In the stateful setting, it may be more natural to consider a security model where the adversary is allowed to request the challenger to create a secret key without actually seeing it. Such a query will change the internal state of KeyGen in a possibly malicious way. In our work, we follow the stateful functional encryption formalization of [ALS16] and do not consider this stronger security model. We leave it open the problem of constructing efficient NIPE scheme satisfying this security notion.

6.3.2 Sampling Algorithm SampleSkewed

We introduce a new sampling algorithm SampleSkewed which is a slight modification of the algorithm SampleRight (See 2.2.2). Recall that even if we are in possession of a "nice" trapdoor matrix $\mathbf{R}$, we can not use the SampleRight algorithm in case t is not invertible over $\mathbb{Z}_q$. Below we consider the case where $q = p^d$ for some prime p and positive integer d, and slightly modify SampleRight so that we can sample short vectors from some shifted lattice of $\Lambda^{\perp}([\mathbf{A}|\mathbf{AR} + p^{d-1}t'\mathbf{G}])$ for an invertible element $t' \in \mathbb{Z}_q$. Note that $t = p^{d-1}t'$ is no longer invertible over $\mathbb{Z}_q$.

Lemma 6.1 (Algorithm SampleSkewed)**.** *Let $q = p^d$ for a prime p and positive integer d. Then, there exists a polynomial time algorithm* SampleSkewed *with the following property.*

SampleSkewed$(\mathbf{A}, \mathbf{G}, \mathbf{R}, t, p^{d-1}\mathbf{u}, \mathbf{T}_\mathbf{G}) \to \mathbf{e}$*: a randomized algorithm that, given full-rank matrices $\mathbf{A}, \mathbf{G} \in \mathbb{Z}_q^{n\times m}$, a matrix $\mathbf{R} \in \mathbb{Z}^{m\times m}$, a vector $p^{d-1}\mathbf{u} \in \mathbb{Z}_q^n$, and an invertible element $t \in \mathbb{Z}_q$, outputs a vector $\mathbf{e} \in \mathbb{Z}^{2m}$ such that $[\mathbf{A}|\mathbf{AR} + p^{d-1} \cdot t \cdot \mathbf{G}]\mathbf{e} = p^{d-1}\mathbf{u} \mod q$ and $\|\mathbf{e}\| \le s_1(\mathbf{R})\sqrt{m} \cdot \omega(\sqrt{\log n})$ with all but negligible probability.*

Proof. The proof follows in a straight forward manner from the trapdoor technique used in [MP12]. We describe how algorithm SampleSkewed works. It first samples a vector $\mathbf{z} \in \mathbb{Z}^m$ such that $\mathbf{Gz} = t^{-1}\mathbf{u} \mod q$ by invoking SamplePre with trapdoor $\mathbf{T}_\mathbf{G}$ of $\mathbf{G}$, where t^{-1} is well-defined since t is invertible in $\mathbb{Z}_q$. Then it returns the vector $\mathbf{e} = \begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_m \end{bmatrix} \mathbf{z} \in \mathbb{Z}^{2m}$ as its output.

We show that vector $\mathbf{e}$ has the desired property. First, observe that

$$\begin{aligned}[]
[\mathbf{A}|\mathbf{AR} + p^{d-1} \cdot t \cdot \mathbf{G}]\mathbf{e} &= [\mathbf{A}|\mathbf{AR} + p^{d-1} \cdot t \cdot \mathbf{G}] \begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_m \end{bmatrix} \mathbf{z} \\
&= p^{d-1} \cdot t \cdot \mathbf{Gz} \\
&= p^{d-1}\mathbf{u} \mod q.
\end{aligned}$$

Finally, we have $\|\mathbf{e}\| \le (s_1(\mathbf{R})+1)\|z\| \le s_1(\mathbf{R})\sqrt{m}\cdot\omega(\sqrt{\log n})$, since we have $\|z\| \le \sqrt{m}\omega(\sqrt{\log n})$ from Lemma 2.3 and Lemma 2.12. This completes the proof. $\square$

6.3.3 Multi-Dimensional Lattices

In this chapter, we require a generalization of standard lattices which we call *multi-dimensional* lattices. We provide the minimum explanation of multi-dimensional lattices that are required to understand the main contribution of this chapter. We refer the interested readers to Section 6.7. Specifically, we introduce new techniques over lattices to prove the below key theorem which we believe to be of an independent interest.

For an m-dimensional lattice $\Lambda \subseteq \mathbb{Z}^m$, define the m-dimensional k-multi lattice Λ^k as $[\Lambda|\cdots|\Lambda] = \{[z_1|\cdots|z_k]|\forall z_i \in \Lambda, \forall i \in [k]\} \subseteq \mathbb{Z}^{m\times k}$. For a matrix $\mathbf{T} = [\mathbf{t}_1|\cdots|\mathbf{t}_k] \in \mathbb{Z}^{m\times k}$, denote $\Lambda^k + \mathbf{T}$ as $[\Lambda + \mathbf{t}_1|\cdots|\Lambda + \mathbf{t}_k] \subseteq \mathbb{Z}^{m\times k}$. For a matrix $\mathbf{M} \in \mathbb{Z}^{k\times\ell}$ define $\Lambda^k \cdot \mathbf{M}$ as the multi lattice $\{\mathbf{VM}|\mathbf{V} \in \Lambda^k\} \subseteq \mathbb{Z}^{m\times\ell}$.

Analogously to standard (one-dimensional) lattices, for an m-dimensional k-multi lattice Λ^k, we define the discrete Gaussian distribution over Λ^k with center $\mathbf{C} \in \mathbb{Z}^{m\times k}$ and parameter σ denoted as $D_{\Lambda^k,\sigma,\mathbf{C}}$ by the process of sampling a matrix whose i-th column is a sample from $D_{\Lambda,\sigma,\mathbf{C}_i}$ for $i \in [k]$, where $\mathbf{C}_i$ denotes the i-th column of $\mathbf{C}$. This definition extends naturally to shifted multi-lattices as well.

Key Theorem. The following theorem concerning on the distribution of the sum of discrete Gaussians plays a central roll in our security proof. The proof of the theorem is given in Section 6.7 with a more formal treatment on the output distribution.

Theorem 6.1. *Let q be a prime or some power of a prime p. Let n, m, ℓ, t be positive integers such that $m \geq 2n \log q$ and $\ell > t$, let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be a random matrix and $\mathbf{T} \in \mathbb{Z}^{m\times \ell}$ be an arbitrary matrix. Let $\mathbf{M} \in \mathbb{Z}^{\ell\times(\ell-t)}$ and $\mathbf{W} \in \mathbb{Z}^{\ell\times t}$ be full rank matrices satisfying $\mathbf{W}^\top\mathbf{M} = \mathbf{0} \in \mathbb{Z}^{t\times(\ell-t)}$. Finally, let σ be a positive real such that $\sigma > \sqrt{s_1(\mathbf{W}^\top\mathbf{W})} \cdot \omega(\sqrt{\log m})$.*

If, $\mathbf{X} \in \mathbb{Z}^{m\times\ell}$ is distributed as $D_{\Lambda^\perp(\mathbf{A})^\ell+\mathbf{T},\sigma}$, then $\mathbf{XM} \in \mathbb{Z}^{m\times(\ell-t)}$ is statistically close to to a distribution parameterized by $\Lambda^\perp(\mathbf{A}), \sigma, \mathbf{M}, (\mathbf{TM} \mod \Lambda^\perp(\mathbf{A})^\ell\mathbf{M})$.

Remark 6.1. *An important observation is that, if we independently sample $\mathbf{X}_0 \leftarrow D_{\Lambda^k+\mathbf{T}_0,\sigma}$ and $\mathbf{X}_1 \leftarrow D_{\Lambda^k+\mathbf{T}_1,\sigma}$, then the distributions of $\mathbf{X}_0\mathbf{M}$ and $\mathbf{X}_1\mathbf{M}$ are statistically close whenever $\mathbf{T}_0\mathbf{M} = \mathbf{T}_1\mathbf{M} \mod \Lambda^k\mathbf{M}$. This is the key insight used in our security proof; in the real world the secret components are sampled as $\mathbf{X}_0$ and in the simulated world they are sampled as $\mathbf{X}_1$. Furthermore, for any matrix $\bar{\mathbf{M}}$, if we let $\mathbf{M}$ be an arbitrary maximal independent subset of the columns of $\bar{\mathbf{M}}$, since all the columns of $\mathbf{X}\bar{\mathbf{M}}$ are linear combinations of the columns of $\mathbf{XM}$, the distribution of $\mathbf{X}\bar{\mathbf{M}}$ is parameterized solely by the distribution of $\Lambda, \sigma, \mathbf{M}, (\mathbf{TM} \mod \Lambda^k\mathbf{M})$.*

6.4 Construction from Lattices with Inner Product over $\mathbb{Z}$

6.4.1 Constructions

Here we construct a *stateless* NIPE scheme with inner product space $\mathbb{Z}$. We consider the predicate space $\mathcal{P} = \{-P+1, \dots, P-2, P-1\}^\ell \subset \mathbb{Z}^\ell$ and attribute space $\mathcal{I} = \{-I+1, \dots, I-2, I-1\}^\ell \subset \mathbb{Z}^\ell$ for some integers $P = P(n), I = I(n)$, where $\ell = \ell(n)$ is typically taken to be $\mathsf{poly}(n)$, and set the modulus size to be a prime $q = q(n)$ such that the inner products of the predicate and attribute vectors do not wrap around q, i.e., $\ell PI < q$. Other parameters including $m(n), \sigma(n), \alpha(n), \alpha'(n), s(n)$ are specified later. Here, we assume that the message space is $\{0, 1\}$. For the multi-bit variant, we refer Section 6.4.4.

$\mathsf{Setup}(1^n, 1^\ell)$: On input $1^n, 1^\ell$, it samples a random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, a random vector $\mathbf{u} \leftarrow \mathbb{Z}_q^n$ and random matrices $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m,\sigma}\right)^m$ for $i \in [\ell]$. It then sets $\mathbf{B}_i = \mathbf{AR}_i \mod q$. Finally, it outputs

$$\mathsf{MPK} = (\mathbf{A}, \mathbf{B}_1, \cdots, \mathbf{B}_\ell, \mathbf{u}) \quad \text{and} \quad \mathsf{MSK} = (\mathbf{R}_1, \cdots, \mathbf{R}_\ell).$$

$\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \vec{y} \in \mathcal{P})$: Given a predicate vector $\vec{y} = (y_1, \cdots, y_\ell) \in \mathcal{P}$, it computes

$$\mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} y_i\mathbf{R}_i \in \mathbb{Z}^{m\times m}.$$

Then, it returns the secret key $\mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}}$.

$\mathsf{Enc}(\mathsf{MPK}, \vec{x} \in \mathcal{I}, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0, 1\}$ for an attribute $\vec{x} = (x_1, \cdots, x_\ell) \in \mathcal{I}$, it samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $z \leftarrow D_{\mathbb{Z},\alpha q}$ and $\mathbf{z}_i \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ for $i \in [0, \ell]$, and computes

$$\begin{cases} c = \mathbf{u}^\top\mathbf{s} + z + \mathsf{M}\lfloor q/2 \rceil, \\ \mathbf{c}_0 = \mathbf{A}^\top\mathbf{s} + \mathbf{z}_0, \\ \mathbf{c}_i = (\mathbf{B}_i + x_i\mathbf{G})^\top\mathbf{s} + \mathbf{z}_i, \quad (i \in [\ell]). \end{cases}$$

Then, it returns the ciphertext $C = (c, (\mathbf{c}_i)_{i\in[0,\ell]}) \in \mathbb{Z}_q \times (\mathbb{Z}_q^m)^{(\ell+1)}$ with the corresponding attribute $\vec{x}$.

$\mathsf{Dec}(\mathsf{MPK}, (\vec{y}, \mathsf{sk}_{\vec{y}}), (\vec{x}, C))$: To decrypt a ciphertext $C = (c, (\mathbf{c}_i)_{i\in[0,\ell]})$ with an associating attribute $\vec{x} \in \mathcal{I}$ using a secret key $\mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} y_i \mathbf{R}_i$ with an associating predicate $\vec{y} \in \mathcal{P}$, it first computes

$$\mathbf{c}_{\vec{y}} = \sum_{i=1}^{\ell} y_i \mathbf{c}_i \in \mathbb{Z}_q^m.$$

Next, it samples a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$ by running $\mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}_{\vec{y}}, \langle \vec{x}, \vec{y} \rangle, \mathbf{u}, \mathbf{T}_{\mathbf{G}}, s)$. Then, it computes $w = c - \mathbf{e}^\top [\mathbf{c}_0^\top | \mathbf{c}_1^\top]^\top \in \mathbb{Z}_q$. Finally, it returns 1 if $|w - \lceil q/2 \rceil| < \lceil q/4 \rceil$ and 0 otherwise.

6.4.2 Correctness and Parameter Selection

The following lemma states the correctness of our proposed NIPE scheme.

Lemma 6.2 (correctness)**.** *Assume $\big(\alpha q + \ell P^2 \sigma m \alpha' q\big) \cdot \omega(\sqrt{\log n}) < q/5$ holds with overwhelming probability. Then the above scheme has negligible decryption error.*

Proof. To establish correctness of decryption, we only need to consider the case $\langle \vec{x}, \vec{y} \rangle \neq 0 \in \mathbb{Z}$. Note that due to our parameter selection, we have $|\langle \vec{x}, \vec{y} \rangle| < q$, hence $\langle \vec{x}, \vec{y} \rangle$ is invertible in $\mathbb{Z}_q$ for q a prime. First, notice that

$$\begin{aligned}
\mathbf{c}_{\vec{y}} = \sum_{i=1}^{\ell} y_i \mathbf{c}_i &= \sum_{i=1}^{\ell} y_i \Big((\mathbf{B}_i + x_i \mathbf{G})^\top \mathbf{s} + \mathbf{z}_i \Big) \\
&= \Big(\mathbf{A} \sum_{i=1}^{\ell} y_i \mathbf{R}_i + \langle \vec{x}, \vec{y} \rangle \mathbf{G} \Big)^\top \mathbf{s} + \sum_{i=1}^{\ell} y_i \mathbf{z}_i \\
&= \Big(\mathbf{A}\mathbf{R}_{\vec{y}} + \langle \vec{x}, \vec{y} \rangle \mathbf{G} \Big)^\top \mathbf{s} + \mathbf{z}',
\end{aligned}$$

where we set $\mathbf{z}' = \sum_{i=1}^{\ell} y_i \mathbf{z}_i$ and recall $\mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}}$. Now, since each row of $\mathbf{R}_i$ are independent, each row of $\mathbf{R}_{\vec{y}}$ are distributed according to $D_{\mathbb{Z}^m, \|\vec{y}\|\sigma}$ from the linear structure of subgaussian random variables. Therefore,

$$s_1(\mathbf{R}_{\vec{y}}) \; = \; s_1\big(\sum_{i=1}^{\ell} y_i \mathbf{R}_i\big) \; \leq \; C \cdot \sqrt{\ell} P \sigma \cdot \sqrt{m} \tag{6.1}$$

where, the inequality follows from Lemma 2.2 and the fact that $\vec{y} \in \mathcal{P}$.

Next, since $\langle \vec{x}, \vec{y} \rangle$ is invertible in $\mathbb{Z}_q$, algorithm $\mathsf{SampleRight}$ work as specified, i.e., it outputs a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$ such that $[\mathbf{A} | \mathbf{A}\mathbf{R}_{\vec{y}} + \langle \vec{x}, \vec{y} \rangle \mathbf{G}]\mathbf{e} = \mathbf{u}$. Therefore,

$$\mathbf{e}^\top \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_{\vec{y}} \end{bmatrix} = \mathbf{e}^\top [\mathbf{A} | \mathbf{A}\mathbf{R}_{\vec{y}} + \langle \vec{x}, \vec{y} \rangle \mathbf{G}]^\top \mathbf{s} + \mathbf{e}^\top [\mathbf{z}_0^\top | \mathbf{z}'^\top]^\top = \mathbf{u}^\top \mathbf{s} + z'' \in \mathbb{Z}_q,$$

where we set $z'' = \mathbf{e}^\top [\mathbf{z}_0^\top | \mathbf{z}'^\top]^\top$. Then, we have $w = \mathsf{M} \lfloor q/2 \rceil + z - z''$. Finally,

$$\begin{aligned}
|z - z''| &\leq |z| + |\mathbf{e}^\top [\mathbf{z}_0^\top | \mathbf{z}'^\top]^\top| \\
&\leq |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\mathbf{e}^\top \mathbf{z}'\|
\end{aligned} \tag{6.2}$$

$$
\begin{aligned}
&= |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\sum_{i=1}^{\ell} y_i \cdot \mathbf{e}^\top \mathbf{z}_i\| \\
&\leq \Big(\alpha q + P \cdot s_1(\mathbf{R}_{\vec{y}})\sqrt{m\ell}\alpha' q\Big)\omega(\sqrt{\log n}) \qquad (6.3)\\
&\leq \Big(\alpha q + \ell P^2 \sigma m \alpha' q\Big)\omega(\sqrt{\log n}) \qquad (6.4)
\end{aligned}
$$

where Eq.(6.2) follows from the sub-additivity of the square root function $\sqrt{\cdot}$, Eq.(6.3) follows from the linear structure of subgaussian random variables, Lemma 2.12, Lemma 2.3 and the fact that $\vec{y} \in \mathcal{P}$, Eq.(6.4) follows from Eq.(6.1). Note that we hide the constant factors inside $\omega(\cdot)$.

By assumption this is smaller than $q/5$ with overwhelming probability. Hence, the error probability of the Decrypt algorithm is negligible. □

Parameter Selection. To satisfy the correctness requirement and make the security proof follow through, we need the following:

- the inner product between any attribute vector $\vec{x} \in \mathcal{I}$ and predicate vector $\vec{y} \in \mathcal{P}$ satisfies $|\langle \vec{x}, \vec{y} \rangle| < q$ (i.e., $\ell P I < q$),
- the error term is less than $q/5$ with overwhelming probability (i.e., $\big(\alpha q + \ell P^2 \sigma m \alpha' q\big)\omega(\sqrt{\log n}) < q/5$. See Lemma 6.2),
- the gadget matrix $\mathbf{G}$ is well defined (i.e., $m \geq n\lceil \log q \rceil$. See Lemma 2.12.),
- σ is sufficiently large so that $\mathbf{R}_i$'s are samplable, and Theorem 6.1 is applicable during the security proof. (i.e., $\sigma > \omega(\sqrt{\log n})$ and $\sigma > \sqrt{\ell I} \cdot \omega(\sqrt{\log n})$). See Lemma 2.12),
- the SampleRight algorithm works as specified (i.e., $s > s_1(\vec{\mathbf{R}}_{\vec{y}}) \cdot \omega(\sqrt{\log m})$ for all predicate vector $\vec{y} \in \mathcal{P}$. See Lemma 2.12),
- the ReRand algorithm in the security proof works as specified (i.e., $\alpha' > 2\alpha(s_1(\vec{\mathbf{R}}) + 1)$, $\alpha q > \omega(\sqrt{\log m\ell})$ where $\vec{\mathbf{R}} \in \mathbb{Z}^{m \times m(\ell+1)}$ is the concatenation of the $\mathbf{R}_i$'s. See Lemma 2.6),
- the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{n}$). See Section 2.2.3.).

Recall that $P(n)$ and $I(n)$ is the bound on the size of the predicate and attribute vectors and $\ell(n)$ is the dimension of the attribute/predicate vectors, where ℓ is set as $\mathsf{poly}(n)$ in a typical setting. To satisfy the above requirements, we propose a candidate parameter selections as follows:

$$
\begin{aligned}
&m = n\lceil \log q \rceil, && q = \ell^2 P^2 I m^2 \cdot \omega(\log n)^{1.5}, && \sigma = \sqrt{\ell I} \cdot \omega(\sqrt{\log n}),\\
&\alpha = (\ell^2 P^2 I m^{1.5} \cdot \omega(\log n)^{1.5})^{-1}, && \alpha' = (\ell^{1.5} P^2 I m \cdot \omega(\log n))^{-1}, && s = \ell P I \sqrt{m} \cdot \omega(\log n),
\end{aligned}
$$

and round up q to the nearest larger prime.

6.4.3 Security Proof

Theorem 6.2. *The above NIPE scheme with inner product space $\mathbb{Z}$ is selectively secure assuming* $\mathsf{LWE}_{n,m+1,q,\chi}$ *is hard, where $\chi = D_{\mathbb{Z},\alpha q}$.*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the selective security of the NIPE scheme. In addition, let $Q = Q(n)$ be the number of key extraction queries $\mathcal{A}$ makes, and denote $\vec{y}^{(k)} \in \mathcal{P}$ as the k-th predicate vector $\mathcal{A}$ queries, where $k \in [Q]$. Here, we assume that $\mathcal{A}$ always queries for $\ell - 1$ linearly independent predicate vectors, which are all orthogonal to the challenge attribute vector $\vec{x}^*$ over $\mathbb{Z}$. This can be done without loss of generality, since $\mathcal{A}$ can simply ignore these additional queries. The proof proceeds with a sequence of games that starts with the real game and ends with a game in which $\mathcal{A}$ has negligible advantage. For each game Game_i denote S_i the event that $\mathcal{A}$ wins the game.

Game_0 : This is the real security game. Namely, adversary $\mathcal{A}$ declares its challenge attribute vector $\vec{x}^* \in \mathcal{I}$ at the beginning of the game. Note that any predicate vector $\vec{y} \in \mathcal{P}$ queried by $\mathcal{A}$ to the challenger as a key extraction query must satisfy $\langle \vec{x}^*, \vec{y} \rangle = 0$ over $\mathbb{Z}$ if $\mathcal{A}$ is a legitimate adversary.

Game_1 : In this game, we change the way the public matrices $\mathbf{B}_1, \cdots, \mathbf{B}_\ell$ are created. On receiving the challenge attribute vector $\vec{x}^* = (x_1^*, \cdots, x_\ell^*) \in \mathcal{I}$ from adversary $\mathcal{A}$ at the beginning of the game, the challenger samples random matrices $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m,\sigma}\right)^m$ and sets $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G} \bmod q$ for $i \in [\ell]$. Otherwise, the behavior of the challenger is identical as in Game_0. Namely, the challenger remains to answer the key extraction query for a predicate vector $\vec{y} \in \mathcal{P}$ as $\mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} y_i \mathbf{R}_i$ where $\vec{y} = (y_1, \cdots, y_\ell)$, and creates the challenge ciphertext as in Game_0.

Before continuing to Game_2, we show that Game_0 is statistically indistinguishable from Game_1; this is the crux of our proof. In particular, we show that the view of the adversary in both games is statistically close. Here, the view of the adversary is completely determined by

$$\left\{ \mathsf{MPK} = \left\{ \mathbf{A}, \{\mathbf{B}_i\}_{i\in[\ell]}, \mathbf{u} \right\}, \quad \{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[Q]}, \quad C^* \right\}$$

where $\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[Q]}$ is the set of secret keys returned by the challenger during the key extraction query and $C^* \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ is the challenge ciphertext, where b is the random bit chosen by the challenger. Observe that in both games $\mathbf{A}, \mathbf{u}$ are distributed identically. Furthermore, the challenge ciphertext C^* is created using only the terms in MPK (with some extra randomness that are identical in both games). Furthermore, from our assumption on $\mathcal{A}$, we assume that $\{\vec{y}^{(k)}\}_{k\in[\ell-1]}$ is the set of the $\ell - 1$ linearly independent vectors that $\mathcal{A}$ queries. Then, what we need to consider are only the $\ell - 1$ secret keys $\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]}$, since all the other secret keys can be created by the linear combinations of $\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]}$. Therefore, the difference in the views of the adversary in Game_0 and Game_1 is determined solely by the difference in the distribution of

$$\left\{ \{\mathbf{B}_i\}_{i\in[\ell]}, \quad \{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} \right\}. \tag{6.5}$$

Hence, we aim at proving that the view of Eq.(6.5) for the adversary is statistically close in both games. More strictly, we compare the following probability of each game:

$$\begin{aligned}
&\Pr\left[\left\{ \{\mathbf{B}_i\}_{i\in[\ell]}, \quad \{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} \right\} = \left\{ \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]}, \quad \{\widehat{\mathbf{R}}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} \right\} \right] \\
=&\underbrace{\Pr\left[\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} = \{\widehat{\mathbf{R}}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} \;\middle|\; \{\mathbf{B}_i\}_{i\in[\ell]} = \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} \right]}_{\text{(A)}} \cdot \underbrace{\Pr\left[\{\mathbf{B}_i\}_{i\in[\ell]} = \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} \right]}_{\text{(B)}},
\end{aligned}$$

where the probability is taken over the randomness of $\{\mathbf{R}_i\}_{i\in[\ell]}$ during Setup; recall each $\mathbf{R}_i$ is distributed according to $\left(D_{\mathbb{Z}^m,\sigma}\right)^m$ in both games. Note that in the above we abuse the notation for sets by implicitly assigning an order over the elements, i.e., $\{\mathbf{X},\mathbf{Y}\} \neq \{\mathbf{Y},\mathbf{X}\}$.

We first prove that the value of (B) is negligibly close in both games. Observe that for all $i \in [\ell]$, $\mathbf{A}\mathbf{R}_i$ is distributed uniformly at random over $\mathbb{Z}_q^{n\times m}$ with all but negligible probability where $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m,\sigma}\right)^m$, which follows from Lemma 2.1 and our parameter selections. Concretely, since $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i$ and $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G}$ for Game_0 and Game_1, respectively, we have that in both games $\{\mathbf{B}_i\}_{i\in[\ell]}$ is distributed statistically close to uniform over $\left(\mathbb{Z}_q^{n\times m}\right)^\ell$.

We now proceed to prove that the value of (A) is negligibly close in both games. We first analyze the case for Game_0. Let $\vec{\mathbf{B}}_{\mathsf{view}} \in \mathbb{Z}_q^{n\times m\ell}$ and $\vec{\mathbf{R}} \in \mathbb{Z}^{m\times m\ell}$ denote the matrices $[\mathbf{B}_1|\cdots|\mathbf{B}_\ell]$ and $[\mathbf{R}_1|\cdots|\mathbf{R}_\ell]$, respectively. Then we have $\vec{\mathbf{B}}_{\mathsf{view}} = \mathbf{A}\vec{\mathbf{R}} \mod q$. Furthermore, let $\vec{\mathbf{T}} = [\mathbf{T}_1|\cdots|\mathbf{T}_\ell] \in \mathbb{Z}^{m\times m\ell}$ be an arbitrary solution to $\vec{\mathbf{B}}_{\mathsf{view}} = \mathbf{A}\vec{\mathbf{T}} \mod q$. Then, due to Lemma 2.1, conditioned on $\{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} = \{\mathbf{A}\mathbf{R}_i\}_{i\in[\ell]} \pmod q$, the conditional distribution of $\vec{\mathbf{R}}$ is $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}},\sigma}$. Now, we are ready to determine the conditional distribution of the secret keys $\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]}$ obtained by the adversary $\mathcal{A}$. Observe the following equation:

$$\underbrace{[\mathbf{R}_{\vec{y}^{(1)}}|\mathbf{R}_{\vec{y}^{(2)}}|\cdots|\mathbf{R}_{\vec{y}^{(\ell-1)}}]}_{:=\vec{\mathbf{R}}_{\mathsf{sk}}\ \in\mathbb{Z}^{m\times m(\ell-1)}} = \underbrace{[\mathbf{R}_1|\mathbf{R}_2|\cdots|\mathbf{R}_\ell]}_{=\vec{\mathbf{R}}\ \in\mathbb{Z}^{m\times m\ell}} \underbrace{\left[\begin{array}{c|c|c|c} y_1^{(1)}\mathbf{I}_m & y_1^{(2)}\mathbf{I}_m & \cdots & y_1^{(\ell-1)}\mathbf{I}_m \\ y_2^{(1)}\mathbf{I}_m & y_2^{(2)}\mathbf{I}_m & & y_2^{(\ell-1)}\mathbf{I}_m \\ \vdots & \vdots & \cdots & \vdots \\ y_\ell^{(1)}\mathbf{I}_m & y_\ell^{(2)}\mathbf{I}_m & & y_\ell^{(\ell-1)}\mathbf{I}_m \end{array}\right]}_{:=\mathbf{M}=\mathbf{Y}\otimes\mathbf{I}_m\ \in\mathbb{Z}^{m\ell\times m(\ell-1)}}, \tag{6.6}$$

where $y_j^{(k)}$ is the j-th entry of the k-th predicate vector $\vec{y}^{(k)}$ and $\mathbf{Y} \in \mathbb{Z}^{\ell\times(\ell-1)}$ is a full rank matrix whose k-th column is $\vec{y}^{(k)}$. We also denote the left and right hand matrices as $\vec{\mathbf{R}}_{\mathsf{sk}}$ and $\mathbf{M} \in \mathbb{Z}^{m\ell\times m(\ell-1)}$, respectively. Note that the equality is taken over $\mathbb{Z}$. Now, since $\vec{x}^{\star\top}\mathbf{Y} = \mathbf{0} \in \mathbb{Z}^{1\times(\ell-1)}$, we have $\mathbf{W}^\top\mathbf{M} = \mathbf{0} \in \mathbb{Z}^{m\times m(\ell-1)}$ where $\mathbf{W} = \vec{x}^\star \otimes \mathbf{I}_m \in \mathbb{Z}^{m\ell\times m}$ is a full rank matrix. Furthermore, by construction, we have $\sqrt{s_1(\mathbf{W}^\top\mathbf{W})} = \|\vec{x}^*\|$. Therefore, by Theorem 6.1 and from the fact that $\vec{\mathbf{R}}$ is distributed according to $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}},\sigma}$, for our parameter selection, we have that the distribution of $\vec{\mathbf{R}}_{\mathsf{sk}} = \vec{\mathbf{R}}\mathbf{M}$ is statistically close to a distribution parameterized by $\Lambda^\perp(\mathbf{A}), \sigma, \mathbf{M}$ and $(\vec{\mathbf{T}}\mathbf{M} \mod \Lambda^\perp(\mathbf{A})^{m\ell}\mathbf{M})$.

We now show that this holds in case for Game_1 as well. Similarly to above, we begin by determining the conditional distribution of $\vec{\mathbf{R}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]} = \{\mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G}\}_{i\in[\ell]}$. Let us denote $\vec{\mathbf{G}}_{\vec{x}^*} \in \mathbb{Z}_q^{n\times m\ell}$ as the matrix $[x_1^*\mathbf{G}|x_2^*\mathbf{G}|\cdots|x_\ell^*\mathbf{G}]$. Then, $\vec{\mathbf{B}}_{\mathsf{view}} + \vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}\vec{\mathbf{R}} \mod q$. Next, let us chose an arbitrary matrix $\mathbf{E} \in \mathbb{Z}^{m\times m}$ such that $\mathbf{G} = \mathbf{A}\mathbf{E} \mod q$, and define $\vec{\mathbf{E}}_{\vec{x}^*} \in \mathbb{Z}^{m\times m\ell}$ as the matrix $[x_1^*\mathbf{E}|x_2^*\mathbf{E}|\cdots|x_\ell^*\mathbf{E}]$. Then, we have $\vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}\vec{\mathbf{E}}_{\vec{x}^*} \mod q$. Combining this with the $\vec{\mathbf{T}}$ we have defined above in Game_0, we obtain $\vec{\mathbf{B}}_{\mathsf{view}} + \vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}(\vec{\mathbf{T}} + \vec{\mathbf{E}}_{\vec{x}^*}) \mod q$. Therefore, by Lemma 2.1, the conditional distribution of $\vec{\mathbf{R}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]}$ is $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}}+\vec{\mathbf{E}}_{\vec{x}^*},\sigma}$. Next, we determine the conditional distribution of the secret keys $\{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]}$ obtained by the adversary $\mathcal{A}$. Observe that equation Eq.(6.6) holds for Game_1 as well, since we do not change the way we answer the key extraction queries. Concretely, we have $\mathbf{M} = \mathbf{Y}\otimes\mathbf{I}_m$ and $\mathbf{W}^\top\mathbf{M} = \mathbf{0}$ where $\mathbf{W} = \vec{x}^\star\otimes\mathbf{I}_m$. Hence, by Theorem 6.1 and the fact that $\vec{\mathbf{R}}$ is distributed according to $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}}+\vec{\mathbf{E}}_{\vec{x}^*},\sigma}$, we have that the distribution of $\vec{\mathbf{R}}_{\mathsf{sk}} = \vec{\mathbf{R}}\mathbf{M}$ is statistically close to a distribution parameterized by $\Lambda^\perp(\mathbf{A}), \sigma, \mathbf{M}$ and $(\vec{\mathbf{T}}\mathbf{M} + \vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} \mod \Lambda^\perp(\mathbf{A})^{m\ell}\mathbf{M})$. Finally, it remains to prove that $\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} = \mathbf{0}$

(over $\mathbb{Z}$) in order to prove equivalence of (A) between Game_0 and Game_1. Observe that

$$\begin{aligned}\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} &= \mathbf{E}\cdot[x_1^*\mathbf{I}_m|x_2^*\mathbf{I}_m|\cdots|x_\ell^*\mathbf{I}_m]\left[\begin{array}{c|c|c|c} y_1^{(1)}\mathbf{I}_m & y_1^{(2)}\mathbf{I}_m & \cdots & y_1^{(\ell-1)}\mathbf{I}_m \\ y_2^{(1)}\mathbf{I}_m & y_2^{(2)}\mathbf{I}_m & & y_2^{(\ell-1)}\mathbf{I}_m \\ \vdots & \vdots & \cdots & \vdots \\ y_\ell^{(1)}\mathbf{I}_m & y_\ell^{(2)}\mathbf{I}_m & & y_\ell^{(\ell-1)}\mathbf{I}_m \end{array}\right] \\ &= \mathbf{E}\cdot[\langle\vec{x}^*,\vec{y}^{(1)}\rangle\mathbf{I}_m|\langle\vec{x}^*,\vec{y}^{(2)}\rangle\mathbf{I}_m|\cdots|\langle\vec{x}^*,\vec{y}^{(\ell-1)}\rangle\mathbf{I}_m] \\ &= \mathbf{0}\in\mathbb{Z}^{m\times m(\ell-1)},\end{aligned}$$

since we have $\langle\vec{x}^*,\vec{y}^{(k)}\rangle = 0$ over $\mathbb{Z}$ for $k\in[\ell-1]$. Hence, we conclude that the value of (A), i.e., the conditional probability of $\vec{\mathbf{R}}_{\mathsf{sk}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]}$, in Game_0 and Game_1 is negligibly close. Therefore, we have $|\Pr[S_0]-\Pr[S_1]| = \mathsf{negl}(n)$.

Game_2 : In this game, we change the way the challenge ciphertext is created. Recall that in the previous game, the challenge ciphertext was created as

$$c = \mathbf{u}^\top\mathbf{s}+z+\mathsf{M}_b\lfloor q/2\rceil,\quad \mathbf{c}_0 = \mathbf{A}^\top\mathbf{s}+\mathbf{z}_0,\quad (\mathbf{c}_i = (\mathbf{A}\mathbf{R}_i)^\top\mathbf{s}+\mathbf{z}_i)_{i\in[\ell]} \tag{6.7}$$

where $\mathbf{s}\leftarrow\mathbb{Z}_q^n$, $z\leftarrow D_{\mathbb{Z},\alpha q}$, $\mathbf{z}_i\leftarrow D_{\mathbb{Z}^m,\alpha' q}$ for $i\in[0,\ell]$, and $b\leftarrow\{0,1\}$, where the last term follows from the fact that in Game_1 we modified $\mathbf{B}_i$ so that $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G}$, and $\mathsf{M}_0,\mathsf{M}_1$ are the two messages sent by the adversary $\mathcal{A}$. To create the challenge ciphertext in Game_2, the challenger first picks $\mathbf{s}\leftarrow\mathbb{Z}_q^n$ and $\mathbf{z}\leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $\mathbf{v} = \mathbf{A}^\top\mathbf{s}+\mathbf{z}\in\mathbb{Z}_q^m$. It then runs the algorithm

$$\mathsf{ReRand}\left([\mathbf{I}_m|\vec{\mathbf{R}}],\mathbf{v},\alpha q,\frac{\alpha'}{2\alpha}\right)\to\mathbf{c}\in\mathbb{Z}_q^{m(\ell+1)}$$

from Lemma 2.6, and parses $\mathbf{c}$ into $\ell+1$ vectors $(\mathbf{c}_i)_{i\in[\ell+1]}$ in $\mathbb{Z}_q^m$ such that $\mathbf{c}^\top = [\mathbf{c}_0^\top|\mathbf{c}_1^\top|\cdots|\mathbf{c}_\ell^\top]\in\mathbb{Z}_q^{m(\ell+1)}$. Finally, it picks $z\leftarrow D_{\mathbb{Z},\alpha q}$, $b\leftarrow\{0,1\}$ and sets the challenge ciphertext as

$$C^* = \Big(c = v+\mathsf{M}_b\lfloor q/2\rceil,\quad \mathbf{c}_0,\quad (\mathbf{c}_i)_{i\in[\ell]}\Big)\in\mathbb{Z}_q\times\mathbb{Z}_q^m\times(\mathbb{Z}_q^m)^\ell, \tag{6.8}$$

where $v = \mathbf{u}^\top\mathbf{s}+z$.

We claim that this change alters the view of $\mathcal{A}$ only negligibly. First, the first term c is distributed identically as in Eq.(6.7). Next, observe that the input to ReRand is $[\mathbf{I}_m|\vec{\mathbf{R}}]\in\mathbb{Z}^{m\times m(\ell+1)}$ and $\mathbf{v} = \mathbf{A}^\top\mathbf{s}+\mathbf{z}\in\mathbb{Z}_q^m$. Therefore, due to Lemma 2.6, for our choices of α and α', the output of ReRand is

$$\begin{aligned}\mathbf{c}^\top &= \big(\mathbf{A}^\top\mathbf{s}\big)^\top[\mathbf{I}_m|\vec{\mathbf{R}}]+\mathbf{z}'^\top \\ &= \mathbf{s}^\top[\mathbf{A}|\mathbf{A}\vec{\mathbf{R}}]+\mathbf{z}'^\top \quad \in\mathbb{Z}_q^{m(\ell+1)},\end{aligned}$$

where the distribution of $\mathbf{z}'$ is within statistical distance from $\mathbf{z}'\leftarrow D_{\mathbb{Z}^{m(\ell+1)},\alpha' q}$. By parsing $\mathbf{c}$ appropriately as above, it can be seen that it is statistically close to $(\mathbf{c}_i)_{i\in[0,\ell]}$ of Eq.(6.7). Therefore, the challenge ciphertexts of Game_1 and Game_2 are statistically indistinguishable. Hence, we have $|\Pr[S_1]-\Pr[S_2]| = \mathsf{negl}(n)$.

Game_3 : In this game, we further change the way the challenge ciphertext is created. To create the challenge ciphertext, the challenger first samples $v\leftarrow\mathbb{Z}_q$, $\mathbf{v}'\leftarrow\mathbb{Z}_q^m$ and $\mathbf{z}\leftarrow D_{\mathbb{Z}^m,\alpha q}$, and runs $\mathsf{ReRand}\Big([\mathbf{I}_m|\vec{\mathbf{R}}], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}\Big) \to \mathbf{c} \in \mathbb{Z}_q^{m(\ell+1)}$, where $\mathbf{v} = \mathbf{v}' + \mathbf{z}$. Then, the challenge ciphertext is set as in Eq.(6.8). We show in Lemma 6.3 below that assuming $\mathsf{LWE}_{n,m+1,q,\chi}$ is hard, we have $|\Pr[S_2] - \Pr[S_3]| = \mathsf{negl}(n)$.

Furthermore, since v is uniformly random over $\mathbb{Z}_q$ and independent of the other values, the term in the challenge ciphertext $c = v + \mathsf{M}_b\lfloor q/2 \rceil$ that conveys the information on the message is distributed independently from the value of M_b. Therefore, we have $\Pr[S_3] = 1/2$. Combining everything together, we have

$$\begin{aligned}\left|\Pr[S_0] - \frac{1}{2}\right| &= \left|\sum_{i=0}^{2}(\Pr[S_i] - \Pr[S_{i+1}]) + \Pr[S_3] - \frac{1}{2}\right| \\ &\leq \sum_{i=0}^{2}|\Pr[S_i] - \Pr[S_{i+1}]| + \left|\Pr[S_3] - \frac{1}{2}\right| \leq \mathsf{negl}(n).\end{aligned}$$

Therefore, the probability that $\mathcal{A}$ wins Game_0 is negligible.

□

To complete the proof of Theorem 6.2, it remains to prove the following Lemma 6.3.

Lemma 6.3. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[S_2] - \Pr[S_3]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m+1,q,\chi}}.$$

In particular, under the $\mathsf{LWE}_{n,m+1,q,\chi}$ assumption, we have $|\Pr[S_2] - \Pr[S_3]| = \mathsf{negl}(n)$.

Proof. Suppose there exists an adversary $\mathcal{A}$ with non-negligible advantage in distinguishing between Game_2 and Game_3 that outputs a value $\mathsf{coin} \in \{0,1\}$, where $\mathsf{coin} = 1$ in case $\mathcal{A}$ decides its interacting with a Game_2 challenger. We use $\mathcal{A}$ to construct an LWE algorithm $\mathcal{B}$ as follows.

Instance. $\mathcal{B}$ is given $\{\mathbf{a}_i, v_i\}_{i=0}^m \in (\mathbb{Z}_q^n \times \mathbb{Z}_q)^{m+1}$ as the problem instance of $\mathsf{LWE}_{n,m+1,q,\chi}$, where recall that $\chi = D_{\mathbb{Z},\alpha q}$. We can assume without loss of generality that $v_i = v_i' + z_i$ for $z_i \leftarrow D_{\mathbb{Z},\alpha q}$ and restate the LWE problem so that $\mathcal{B}$'s task is now to distinguish whether $v_i' = \mathbf{a}_i^\top \mathbf{s}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ or $v_i' \leftarrow \mathbb{Z}_q$ for $i \in [0,m]$. We note this subtle change from the standard LWE problem is only a syntactical change made for the convenience of the proof.

Setup. To construct the master public key MPK, $\mathcal{B}$ first sets the random vector $\mathbf{u}$ as $\mathbf{a}_0$, and assembles the random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ from the remaining LWE samples $\{\mathbf{a}_i\}_{i=1}^m$ by letting the i-th column be the vector $\mathbf{a}_i$. It also samples ℓ random matrices $\mathbf{R}_i \leftarrow (D_{\mathbb{Z}^m,\sigma})^m$ and sets $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G} \mod q$ for $i \in [\ell]$. Finally, it returns $\mathsf{MPK} = (\mathbf{A}, \mathbf{B}_1, \cdots, \mathbf{B}_\ell, \mathbf{u})$ to $\mathcal{A}$.

Phase 1 and Phase 2. The key extraction queries made by $\mathcal{A}$ are answered as in Game_1 (which is equivalent to both Game_2 and Game_3), using the $\mathbf{R}_i$'s created during Setup.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge attribute vector $\vec{x}^*$ and challenge messages $\mathsf{M}_0, \mathsf{M}_1$, $\mathcal{B}$ sets the challenge ciphertext C^* as in Eq.(6.8) and returns C^* to $\mathcal{A}$.

Guess. At last, $\mathcal{A}$ outputs its guess coin. Then, $\mathcal{B}$ outputs 1 if $\mathsf{coin} = 1$ and 0 otherwise.

Analysis. It can be seen that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in Game_2 if $\{\mathbf{a}_i, v_i\}_{i=0}^m$ are valid LWE samples (i.e., $v_i' = \mathbf{a}_i^\top \mathbf{s}$) and Game_3 otherwise (i.e., $v_i' \leftarrow \mathbb{Z}_q$). We therefore conclude that $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m+1,q,\chi}} = |\Pr[S_2] - \Pr[S_3]|$ as desired.

□

6.4.4 Multi-bit Variant

Here, we explain how to extend our scheme to a multi-bit variant without increasing much the size of the master public keys, secret keys, and ciphertexts following the techniques of [PVW08, ABB10, Yam16]. To modify the scheme to deal with message space of length ℓ_M, we replace $\mathbf{u} \in \mathbb{Z}_q^n$ in MPK with $\mathbf{U} \in \mathbb{Z}_q^{n\times\ell_M}$. The component c in the ciphertext is replaced with $\mathbf{c} = \mathbf{U}^\top\mathbf{s} + \mathbf{z} + \mathsf{M}\lceil q/2\rceil$ where $\mathbf{z} \leftarrow D_{\mathbb{Z}^{\ell_M},\alpha q}$ and $\mathsf{M} \in \{0,1\}^{\ell_M}$ is the message to be encrypted. When decrypting the message, one samples a matrix $\mathbf{E} \in \mathbb{Z}^{2m\times\ell_M}$ such that $[\mathbf{A}|\mathbf{A}\mathbf{R}_{\vec{y}} + \langle\vec{x},\vec{y}\rangle\mathbf{G}]\mathbf{E} = \mathbf{U}$, which is possible given $\mathsf{sk}_{\vec{y}}$ by running $\mathsf{SampleRight}$ in a column wise manner. We can prove security for the multi-bit variant from $\mathsf{LWE}_{n,m+\ell_M,q,\chi}$ by naturally extending the proof of Theorem 6.2. We note that the same parameters as in Section 6.5.1 will also work for the multi-bit variant. By this change, the sizes of the master public keys, ciphertexts, and private keys become $\tilde{O}((n^2\ell+n\ell_M)\log q)$, $\tilde{O}((n+\ell+\ell_M)\log q)$, and $\tilde{O}(n^2\log q)$ from $\tilde{O}(n^2\ell\log q)$, $\tilde{O}((n+\ell)\log q)$, and $\tilde{O}(n^2\log q)$, respectively. The sizes of the master public keys and ciphertexts will be asymptotically the same as long as $\ell_M = \tilde{O}(n)$. To deal with longer messages, we employ a KEM-DEM approach as suggested in [Yam16]. Namely, we encrypt a random ephemeral key of sufficient length and then encrypt the message by using the ephemeral key.

6.5 Constructions from Lattices with Inner Product over $\mathbb{Z}_p$

In this section, we construct a *stateful* NIPE scheme with inner product space $\mathbb{Z}_p$ for $p = p(n)$ a prime, where the predicate and attribute spaces are $\mathbb{Z}_p^\ell$.

Overview. We give a more detailed overview on the intuition given in the introduction. First, we need the state to keep track of what kind of predicate vectors $\vec{y}$ we gave out secret keys to. Unlike in the NIPE construction of Section 6.4, for our NIPE scheme with predicate space $\mathbb{Z}_p$, the linear dependency of the predicate vectors (over $\mathbb{Z}_p$) and the secret keys (over $\mathbb{Z}$) are no longer consistent. Namely, when an adversary queries for linearly dependent predicate vectors over $\mathbb{Z}_p$, the corresponding secret keys may no longer be linearly dependent over $\mathbb{Z}$. For our particular construction, when an adversary obtains secret keys to a linearly independent predicate vectors over $\mathbb{Z}$, the scheme leads to a complete break in security. Therefore, we need to maintain information on the linear span of the predicate vectors (over $\mathbb{Z}_p$ and $\mathbb{Z}$) that it has generated secret keys to, and create a secret key for a new predicate vector $\vec{y}$ as a $\mathbb{Z}$-linear combination of the previously generated secret keys if $\vec{y}$ lies in the $\mathbb{Z}_p$-linear span maintained in the state.

Here, we also maintain our state in a unique way, which allows us to base security of our scheme under a weaker polynomial LWE assumption. As already mentioned, the state maintains the information of the linear span of the predicate vectors that it has generated secret keys to. In our scheme, this is expressed by a list of tuples of the form $(\vec{h}^{(i)}, \mathsf{h}^{(i)}, \mathsf{sk}_{\vec{h}^{(i)}}) \in \mathbb{Z}_p^\ell \times \mathbb{Z}^\ell \times \mathbb{Z}^{m\times m}$, where $i \in \mathsf{list} \subseteq [\ell]$. Informally, list indicates the distinctive indices that specifies the linear span of the so far queried predicate vectors, and $|\mathsf{list}|$ is the dimension of the linear span. Furthermore, $\vec{h}^{(i)} \in \mathbb{Z}_p^\ell$ are vectors specifying the linear span of the queried predicate vectors, $\mathsf{h}^{(i)}$ are vectors in $\mathbb{Z}^\ell$ that is in a sense encodings of $\vec{h}^{(i)}$ that maintain linear dependency over $\mathbb{Z}$, and $\mathsf{sk}_{\vec{h}^{(i)}}$ are the secret keys corresponding to the predicate vector $\vec{h}^{(i)}$. When queried a new predicate vector $\vec{y}$, the algorithm first checks if it lies in the $\mathbb{Z}_p$-linear span of $\{\vec{h}^{(i)}\}_{i\in\mathsf{list}}$. If so, (informally) it computes secret keys as a $\mathbb{Z}$-linear combination of $\{\mathsf{sk}_{\vec{h}^{(i)}}\}_{i\in\mathsf{list}}$. If not, it processes $\vec{y}$ into a new vector $\vec{h}^{(j)} \in \mathbb{Z}_p^\ell$ that does not lie in the $\mathbb{Z}_p$-linear span of $\{\vec{h}^{(i)}\}_{i\in\mathsf{list}}$ and adds j to list. Here, in order for us to base security on an LWE assumption with polynomial approximation factor, we need to process $\vec{y}$ in such a way that the matrix with columns $\{\vec{h}^{(i)}\}_{i\in\mathsf{list}}$ interpreted as vectors in $\mathbb{Z}^\ell$ has a small singular value. At a high level, this can be achieved by keeping the diagonal elements small, which we can do since we can store any factor of $\vec{h}^{(i)} \in \mathbb{Z}_p^\ell$ without altering the $\mathbb{Z}_p$-linear span. Here, the crucial observation is that the $\mathbb{Z}_p$-linear dependency of $\{\vec{h}^{(i)}\}_{i\in\mathsf{list}}$ and the size of the singular values of $\{\vec{h}^{(i)}\}_{i\in\mathsf{list}}$ interpreted as a matrix over $\mathbb{Z}$ are (almost completely) independent with each other.

Construction. Let $q = p^d$ for some positive integer $d \geq 3$ and let $m(n), \sigma(n), \alpha(n), \alpha'(n), s(n)$ be parameters that are specified later. Here, we assume that the message space is $\{0,1\}$. We can easily extend the scheme to the multi-bit variant similarly to Section 6.4.4.

$\mathsf{Setup}(1^n, 1^\ell)$: On input $1^n, 1^\ell$, it samples a random matrix $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, a random vector $\mathbf{u} \leftarrow \mathbb{Z}_q^n$, random matrices $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m,\sigma}\right)^m$ for $i \in [\ell]$ and sets $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i \mod q$. Furthermore, it initializes a state st that inculdes an empty list $\mathsf{list} \subseteq [\ell]$. Finally, it outputs

$$\mathsf{MPK} = \Big(\mathbf{A}, \{\mathbf{B}_i\}_{i\in[\ell]}, \mathbf{u}\Big) \quad \text{and} \quad \mathsf{MSK} = \Big(\mathsf{st}, \{\mathbf{R}_i\}_{i\in[\ell]}\Big).$$

$\mathsf{KeyGen}\ (\mathsf{MPK}, \mathsf{MSK}, \vec{y} \in \mathbb{Z}_p^\ell, \mathsf{st})$: Given a predicate vector $\vec{y} \in \mathbb{Z}_p^\ell$ and an internal state st, it computes the secret key $\mathsf{sk}_{\vec{y}}$ as follows. At any point of the execution, the internal state st contains a list of indices $\mathsf{list} \subseteq [\ell]$ and at most ℓ tuples of the form $(\vec{h}^{(i)}, \vec{\mathsf{h}}^{(i)}, \mathsf{sk}_{\vec{h}^{(i)}}) \in \mathbb{Z}_p^\ell \times \mathbb{Z}^\ell \times \mathbb{Z}^{m\times m}$, where the vectors $\{\vec{h}^{(i)}\}_{j\in\mathsf{list}}$ form a basis of the $\mathbb{Z}_p$-linear span of the predicate vectors which the key extraction queries has been made so far.

If $\vec{y} \in \mathbb{Z}_p^\ell$ is linearly independent modulo p from all the $\{\vec{h}^{(j)}\}_{j\in\mathsf{list}}$ in the state st, it first runs the following procedure. By construction, for all $j \in \mathsf{list}$, we will have $(j = \arg\min_{i\in[\ell]}\{h_i^{(j)} \neq 0\}) \wedge (h_j^{(j)} = 1)$, i.e., the smallest index for which the entry of $\vec{h}^{(j)}$ is non-zero is j, and at that index it holds that $h_j^{(j)} = 1$. It sets $\vec{h} = \vec{y}$, and starting with the smallest index $j \in \mathsf{list}$, it iterates through list in ascending order by updating $\vec{h} \leftarrow \vec{h} - h_j \cdot \vec{h}^{(j)} \mod p$ so that the updated $\vec{h}$ satisfies $h_j = 0 \mod p$, where h_j denotes the j-th element of $\vec{h}$. After it runs through all the element in list, it finds the smallest index j' such that $\vec{h}_{j'} \neq 0$. This always exists since $\vec{y}$ is linearly independent modulo p from $\{\vec{h}^{(j)}\}_{j\in\mathsf{list}}$. Then, it updates $\vec{h}$ once more by $\vec{h} \leftarrow (1/h_{j'}) \cdot \vec{h} \mod p$ and sets $\vec{h}^{(j')} = \vec{h} \in \mathbb{Z}_p^\ell$. It can be checked that $(j' = \arg\min_{i\in[\ell]}\{h_i^{(j')} \neq 0\}) \wedge (h_{j'}^{(j')} = 1)$. Finally, it sets $\vec{\mathsf{h}}^{(j')} = \vec{h}^{(j')}$, interpreted as a vector in $\mathbb{Z}^\ell$, and sets $\mathsf{sk}_{\vec{h}^{(j')}}$ as

$$\mathbf{R}_{\vec{h}^{(j')}} = \sum_{i=1}^{\ell} \mathsf{h}_i^{(j')} \mathbf{R}_i \in \mathbb{Z}^{m\times m}, \tag{6.9}$$

where $\mathsf{h}_i^{(j')}$ is the i-th entry of $\vec{\mathsf{h}}^{(j')}$. It then adds j' to list and the tuple $(\vec{h}^{(j')}, \vec{\mathsf{h}}^{(j')}, \mathsf{sk}_{\vec{h}^{(j')}})$ to st.[3] Note that after this procedure, the predicate vector $\vec{y}$ is linearly dependent modulo p with the vectors $\{\vec{h}^{(j)}\}_{j\in\mathsf{list}}$ in the state st. Furthermore, when ℓ linearly independent queries has been made, we have $\mathsf{list} = [\ell]$ and the set of vectors $\{\vec{h}^{(j)}\}_{j\in[\ell]}$ forms a lower triangular matrix with ones along the diagonal.

[3] Although $\vec{h}^{(j')} \in \mathbb{Z}_p^\ell$ and $\vec{\mathsf{h}}^{(j')} \in \mathbb{Z}^\ell$ are in some sense identical, we intentionally write it redundantly in this form for consistency with the other predicate vectors $\vec{y}$, i.e., $(\vec{\mathsf{h}}^{(j')}, \mathsf{sk}_{\vec{h}^{(j')}})$ acts as a valid secret key for the predicate vector $\vec{h}^{(j')}$.

Finally, to construct the secret key for $\vec{y}$, it sets $\vec{y} = \sum_{j \in \mathsf{list}} \lambda_j \vec{h}^{(j)} \mod p$ for some λ_j's in $\mathbb{Z}_p$ and sets $\vec{\mathsf{y}} = \sum_{j \in \mathsf{list}} \lambda_j \vec{\mathsf{h}}^{(j)} \in \mathbb{Z}^\ell$ where here λ_j is viewed as an element over $\mathbb{Z}$. Finally, it sets $\mathsf{sk}_{\vec{y}}$ as

$$\mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \in \mathbb{Z}^{m \times m},$$

where y_i is the i-th entry of $\vec{\mathsf{y}}$, and returns the tuple $(\vec{\mathsf{y}}, \mathsf{sk}_{\vec{y}}) \in \mathbb{Z}^\ell \times \mathbb{Z}^{m \times m}$ as the secret key.

$\mathsf{Enc}(\mathsf{MPK}, \vec{x} \in \mathbb{Z}_p^\ell, \mathsf{M})$: To encrypt a message $\mathsf{M} \in \{0,1\}$ for an attribute $\vec{x} = (x_1, \cdots, x_\ell) \in \mathbb{Z}_p^\ell$, it samples $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{z}_0, \mathbf{z}_i \leftarrow D_{\mathbb{Z}^m, \alpha' q}$ for $i \in [\ell]$, and computes

$$\begin{cases} c = p^{d-1} \cdot \left(\mathbf{u}^\top \mathbf{s} + \mathsf{M} \lfloor p/2 \rceil \right), \\ \mathbf{c}_0 = \mathbf{A}^\top \mathbf{s} + \mathbf{z}_0, \\ \mathbf{c}_i = (\mathbf{B}_i + p^{d-1} \cdot x_i \mathbf{G})^\top \mathbf{s} + \mathbf{z}_i, \quad (i \in [\ell]), \end{cases}$$

Then, it returns the ciphertext $C = (c, \mathbf{c}_0, (\mathbf{c}_i)_{i \in [\ell]}) \in \mathbb{Z}_q \times (\mathbb{Z}_q^m)^{\ell+1}$ with its corresponding attribute $\vec{x}$.

$\mathsf{Dec}(\mathsf{MPK}, (\vec{y}, \vec{\mathsf{y}}, \mathsf{sk}_{\vec{y}}), (\vec{x}, C))$: To decrypt a ciphertext $C = (c, \mathbf{c}_0, (\mathbf{c}_i)_{i \in [\ell]})$ with an associating attribute $\vec{x} \in \mathbb{Z}_p^\ell$, it first computes

$$\mathbf{c}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{c}_i \mod q \ \in \mathbb{Z}_q^m,$$

where y_i is the i-th entry of $\vec{\mathsf{y}}$. Next, it samples a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$ by running $\mathsf{SampleSkewed}(\mathbf{A}, \mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}}, \langle \vec{x}, \vec{y} \rangle, p^{d-1}\mathbf{u}, \mathbf{T_G})$. Then, it computes $t = c - \mathbf{e}^\top [\mathbf{c}_0^\top | \mathbf{c}_{\vec{y}}^\top]^\top \in \mathbb{Z}_q$ Finally, it returns 1 if $|t - \lceil q/2 \rceil| < \lceil q/4 \rceil$ and 0 otherwise.

6.5.1 Correctness and Parameter Selection

The following lemma states the correctness of our proposed NIPE scheme.

Lemma 6.4 (correctness). *Assume $\left(\alpha q + \ell p^2 \sigma m \alpha' q\right) \cdot \omega(\sqrt{\log n}) < q/5$ holds with overwhelming probability. Then the above scheme has negligible decryption error.*

Proof. To establish correctness of decryption, we only need to consider the case $\langle \vec{x}, \vec{y} \rangle \neq 0 \in \mathbb{Z}_p$. First, notice that

$$\begin{aligned} \mathbf{c}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{c}_i &= \sum_{i=1}^{\ell} \mathsf{y}_i \Big((\mathbf{B}_i + p^{d-1} \cdot x_i \mathbf{G})^\top \mathbf{s} + \mathbf{z}_i \Big) \\ &= \Bigg(\mathbf{A} \underbrace{\left(\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \right)}_{= \mathbf{R}_{\vec{y}} \ (= \mathsf{sk}_{\vec{y}})} + p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle \mathbf{G} \Bigg)^\top \mathbf{s} + \underbrace{\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{z}_i}_{:= \mathbf{z}' \ (\text{noise})} \\ &= \Big(\mathbf{A} \mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle \mathbf{G} \Big)^\top \mathbf{s} + \mathbf{z}' \mod q, \end{aligned} \tag{6.10}$$

where we set $\mathbf{z}' = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{z}_i$.

Next, we show that $p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle = p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mod q$. Recall that $\vec{y} = \sum_{j \in \mathsf{list}} \vec{h}^{(j)} \mod p$ and $\vec{\mathsf{y}} = \sum_{j \in \mathsf{list}} \lambda_j \vec{\mathsf{h}}^{(j)}$ for some λ_j's in $\mathbb{Z}_p$ (or view λ_j as an element in $\mathbb{Z}$ for the latter equality), where $\{\vec{h}^{(j)}\}_{j \in \mathsf{list}}$ are the vectors stored in the state st at the time of constructing the secret key for $\vec{y}$ and $\vec{h}^{(j)} = \vec{\mathsf{h}}^{(j)}$ over $\mathbb{Z}$. Therefore, we have $\langle \vec{x}, \vec{\mathsf{y}} \rangle = \langle \vec{x}, \vec{y} \rangle \mod p$, which implies $p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle = p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mod q$. Hence, Eq.(6.10) is equivalent to

$$\mathbf{c}_{\vec{y}} = \left(\mathbf{A}\mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mathbf{G} \right)^{\top} \mathbf{s} + \mathbf{z}' \in \mathbb{Z}_q^m.$$

Observe that since each rows of $\mathbf{R}_i$ are independent, each row of $\vec{\mathbf{R}}_{\vec{y}}$ are distributed according to $D_{\mathbb{Z}^m, \|\vec{\mathsf{y}}^{\top}\|\sigma}$ from the linear structure of subgaussian random variables. Therefore,

$$s_1(\vec{\mathbf{R}}_{\vec{y}}) \;=\; s_1\Big(\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \Big) \;\le\; C \cdot \sqrt{\ell} p \sigma \cdot \sqrt{m} \tag{6.11}$$

where, the inequality follows from Lemma 2.2 and the fact that $\vec{\mathsf{y}} \in \mathbb{Z}_p^{\ell}$.

Since p is a prime, $q = p^d$ and $\langle \vec{x}, \vec{y} \rangle \in \mathbb{Z}_p \backslash \{0\}$, we have that $\langle \vec{x}, \vec{y} \rangle$ is invertible in $\mathbb{Z}_q$. Therefore, algorithm $\mathsf{SampleSkewed}$ works as specified, i.e., it outputs a short vector $\mathbf{e} \in \mathbb{Z}^{2m}$ such that $[\mathbf{A} | \mathbf{A}\mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mathbf{G}]\mathbf{e} = p^{d-1} \cdot \mathbf{u} \mod q$. Hence,

$$\mathbf{e}^{\top} \begin{bmatrix} \mathbf{c}_0 \\ \mathbf{c}_{\vec{y}} \end{bmatrix} = \mathbf{e}^{\top} [\mathbf{A} | \mathbf{A}\mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mathbf{G}]^{\top} \mathbf{s} + \mathbf{e}^{\top} [\mathbf{z}_0^{\top} | \mathbf{z}'^{\top}]^{\top} = p^{d-1} \cdot \mathbf{u}^{\top} \mathbf{s} + z'' \in \mathbb{Z}_q,$$

where we set $z'' = \mathbf{e}^{\top} [\mathbf{z}_0^{\top} | \mathbf{z}'^{\top}]^{\top}$. Then, we have $w = \mathsf{M} \cdot p^{d-1} \lfloor p/2 \rceil + z - z''$. Finally,

$$\begin{aligned}
|z - z''| &\le |z| + |\mathbf{e}^{\top} [\mathbf{z}_0^{\top} | \mathbf{z}'^{\top}]^{\top}| \\
&\le |z| + \|\mathbf{e}^{\top} \mathbf{z}_0\| + \|\mathbf{e}^{\top} \mathbf{z}'\| \qquad (6.12) \\
&= |z| + \|\mathbf{e}^{\top} \mathbf{z}_0\| + \|\sum_{i=1}^{\ell} \mathsf{y}_i \cdot \mathbf{e}^{\top} \mathbf{z}_i\| \\
&\le \Big(\alpha q + p \cdot s_1(\vec{\mathbf{R}}_{\vec{y}}) \sqrt{m\ell} \alpha' q \Big) \omega(\sqrt{\log n}) \qquad (6.13) \\
&\le \Big(\alpha q + \ell p^2 \sigma m \alpha' q \Big) \omega(\sqrt{\log n}) \qquad (6.14)
\end{aligned}$$

where Eq.(6.12) follows from the sub-additivity of the square root function $\sqrt{\cdot}$, Eq.(6.13) follows from the linear structure of subgaussian random variables, Lemma 2.3, Lemma 2.12 and the fact that $\vec{\mathsf{y}} \in \mathbb{Z}_p^{\ell}$, Eq.(6.14) follows from Eq.(6.11). Note that we hide the constant factors inside $\omega(\cdot)$.

By assumption this is smaller than $q/5$ with overwhelming probability. Hence, from the fact that $q = p^d$, the error probability of the $\mathsf{Decrypt}$ algorithm is negligible. □

Parameter Selection. To satisfy the correctness requirement and make the security proof follow through, we need the following:

- the error term is less than $q/5$ with overwhelming probability (i.e., $\big(\alpha q + \ell p^2 \sigma m \alpha' q\big) \omega(\sqrt{\log n}) < q/5$. See Lemma 6.2),
- the gadget matrix $\mathbf{G}$ is well defined (i.e., $m \ge n \lceil \log q \rceil$. See Lemma 2.12.),

– σ is sufficiently large so that $\mathbf{R}_i$'s are samplable and Theorem 6.1 is applicable during the security proof (i.e., $\sigma > \omega(\sqrt{\log n})$ and $\sigma > (p+1)^{\ell+2} \cdot \omega(\sqrt{\log m})$. See Lemma 2.12 and Lemma 6.5),

– the ReRand algorithm in the security proof works as specified (i.e., $\alpha' > 2\alpha(s_1(\vec{\mathbf{R}}) + 1)$, $\alpha q > \omega(\sqrt{\log m\ell})$ where $\vec{\mathbf{R}} \in \mathbb{Z}^{m \times m(\ell+1)}$ is the concatenation of the $\mathbf{R}_i$'s. See Lemma 2.6),

– the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{n}$). (See Theorem 2.1).

Recall that $p(n)$ is the size of the predicate/attribute space and $\ell(n)$ is the dimension of the attribute/predicate vectors and the modulus size q is p^d for $d := d(n)$. To satisfy the above requirements, we propose a candidate parameter selections as follows:

$$m = n\lceil \log q \rceil, \qquad q = p^d, \qquad p^{d-2(\ell+1)} \geq \ell^{1.5} m^2 \omega(\log n)^{2.5},$$
$$\alpha = p^{-2(\ell+1)} \cdot (\ell m \omega(\log n))^{-1.5}, \quad \alpha' = p^{-(\ell+2)} \cdot (\ell m \omega(\log n))^{-1}, \qquad \sigma = p^{\ell} \cdot \omega(\sqrt{\log n}).$$

Therefore, to base the construction on the LWE problem with polynomial modulus q, for example we can set $\ell, d = O(\log n / \log\log n)$ and $p = O(\log n)$ or set $\ell, d = O(\log n)$ and p as some positive constant.

6.5.2 Security Proof

Theorem 6.3. *The above NIPE scheme with inner product space $\mathbb{Z}_p$ is selective secure assuming* $\mathsf{FE.LWE}_{n,m+1,q,\chi}$ *is hard, where $\chi = D_{\mathbb{Z},\alpha q}$*

Proof. Let $\mathcal{A}$ be a PPT adversary that breaks the selective security of the NIPE scheme. Here, assume that $\mathcal{A}$ makes key extraction queries in a way that at the end of the game the state st contains $\ell - 1$ linearly independent (modulo p) predicate vectors $\{\vec{h}^{(j)}\}_{j \in \mathsf{list}}$ where $|\mathsf{list}| = \ell - 1$ (which are all orthogonal modulo p to the challenge attribute vector $\vec{x}^*$). Note that this assumption can be made without loss of generality, since $\mathcal{A}$ may simply ignore unnecessary additional secret keys, and $\mathcal{A}$ can not obtain no more than $\ell - 1$ linearly independent (modulo p) vectors without violating the $\langle \vec{x}^*, \vec{y} \rangle = 0 \mod p$ condition. The proof proceeds with a sequence of games that starts with the real game and ends with a game in which $\mathcal{A}$ has negligible advantage. For each game Game_i denote S_i the event that $\mathcal{A}$ wins the game.

Game_0 : This is the real security game. Namely, adversary $\mathcal{A}$ declares its challenge attribute vector $\vec{x}^* \in \mathbb{Z}_p^\ell$ at the beginning of the game. Note that any predicate vector $\vec{y} \in \mathbb{Z}_p^\ell$ queried by $\mathcal{A}$ to the challenger as a key extraction query must satisfy $\langle \vec{x}^*, \vec{y} \rangle = 0 \mod p$ if $\mathcal{A}$ is a legitimate adversary.

Game_1 : In this game, we change the way the public matrices $\mathbf{B}_1, \cdots, \mathbf{B}_\ell$ are created. On receiving the challenge attribute vector $\vec{x}^* = (x_1^*, \cdots, x_\ell^*) \in \mathbb{Z}_p^\ell$ from adversary $\mathcal{A}$ at the beginning of the game, the challenger samples random matrices $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m,\sigma}\right)^m$ and sets $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - p^{d-1} \cdot x_i^* \mathbf{G} \mod q$ for $i \in [\ell]$. Otherwise, the behavior of the challenger is identical as in Game_0. Namely, the challenger remains to answer the key extraction query for a predicate vector $\vec{y} \in \mathbb{Z}_p^\ell$ and creates the challenge ciphertext as in Game_0.

Before moving on to Game_2, we show that Game_0 is statistically indistinguishable from Game_1. In particular, we prove that the view of the adversary in both games is statistically close. In doing so, we first show that every secret keys are $\mathbb{Z}$-linear combinations of the secret keys stored in the state st. Namely, let $\{\vec{h}^{(j)}\}_{j \in \mathsf{list}}$ denote the vectors stored in the state st on time of constructing the secret key for the queried predicate vector $\vec{y}$, where $\mathsf{list} \subseteq [\ell]$ is the index set contained in st. Then, we want to show that for a predicate vector $\vec{y}$ of the form $\sum_{j \in \mathsf{list}} \lambda_j \vec{h}^{(j)} \mod p$ for some λ_j's in $\mathbb{Z}_p$, the corresponding secret key $\mathsf{sk}_{\vec{y}}$ $(= \mathbf{R}_{\vec{y}})$ is a $\mathbb{Z}$-linear combination of $\{\mathsf{sk}_{\vec{h}^{(j)}} = \mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}}$. To see this let the tuples stored in st be $(\vec{h}^{(j)}, \vec{\mathsf{h}}^{(j)}, \mathsf{sk}_{\vec{h}^{(j)}} = \mathbf{R}_{\vec{h}^{(j)}}) \in \mathbb{Z}_p^{\ell} \times \mathbb{Z}^{\ell} \times \mathbb{Z}^{m \times m}$ for $j \in \mathsf{list}$. Then, we have the following:

$$\mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \overset{\text{(i)}}{=} \sum_{i=1}^{\ell} \left(\sum_{j \in \mathsf{list}} \lambda_j \mathsf{h}_i^{(j)} \right) \mathbf{R}_i = \sum_{j \in \mathsf{list}} \lambda_j \left(\sum_{i=1}^{\ell} \mathsf{h}_i^{(j)} \mathbf{R}_i \right) \overset{\text{(ii)}}{=} \sum_{j \in \mathsf{list}} \lambda_j \mathbf{R}_{\vec{h}^{(j)}} \in \mathbb{Z}^{m \times m},$$

where $\mathsf{h}_i^{(j)}$ is the i-th entry of $\vec{\mathsf{h}}^{(j)}$. Eq. (i) follows from the definition of y_i and Eq. (ii) follows from Eq. (6.9)

Therefore the distribution of the secret keys obtained by adversary $\mathcal{A}$ is completely determined by the distribution of the secret keys $\{\mathsf{sk}_{\vec{h}^{(j)}} = \mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}}$ stored in the state st at the end of the game. Therefore, the view of the adversary in both games is determined by

$$\left\{ \mathsf{MPK} = \left\{ \mathbf{A}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{u} \right\}, \quad \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}}, \quad C^* \right\},$$

where $C^* \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ is the challenge ciphertext, b is the random bit chosen by the challenger and $|\mathsf{list}| = \ell - 1$ by assumption. Observe that in both games $\mathbf{A}, \mathbf{u}$ are distributed identically and the challenge ciphertext C^* is created using only the terms in MPK (with some extra randomness that are identical in both games). Therefore, the differences in the views of the adversary in Game_0 and Game_1 is solely determined by the difference in the distribution of

$$\left\{ \{\mathbf{B}_i\}_{i \in [\ell]}, \quad \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}} \right\}. \tag{6.15}$$

Hence, we aim at proving that the view of Eq.(6.15) in both games are statistically close to the adversary. More specifically, we compare the following probability of each game:

$$\begin{aligned} &\Pr\left[\left\{ \{\mathbf{B}_i\}_{i \in [\ell]}, \ \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}} \right\} = \left\{ \{\widehat{\mathbf{B}}_i\}_{i \in [\ell]}, \ \{\widehat{\mathbf{R}}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}} \right\} \right] \\ = &\underbrace{\Pr\left[\{\mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}} = \{\widehat{\mathbf{R}}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}} \;\middle|\; \{\mathbf{B}_i\}_{i \in [\ell]} = \{\widehat{\mathbf{B}}_i\}_{i \in [\ell]} \right]}_{\text{(A)}} \cdot \underbrace{\Pr\left[\{\mathbf{B}_i\}_{i \in [\ell]} = \{\widehat{\mathbf{B}}_i\}_{i \in [\ell]} \right]}_{\text{(B)}}, \end{aligned}$$

where the probability is taken over the randomness of $\{\mathbf{R}_i\}_{i \in [\ell]}$ during Setup; recall each $\mathbf{R}_i$ is distributed according to $\left(D_{\mathbb{Z}^m, \sigma}\right)^m$ in both games. Note that in the above we abuse the notation for sets by implicitly assigning an order over the elements, i.e., $\{\mathbf{X}, \mathbf{Y}\} \neq \{\mathbf{Y}, \mathbf{X}\}$.

We first prove that the value of (B) is negligibly close in both games. Observe that for all $i \in [\ell]$, $\mathbf{A}\mathbf{R}_i$ is distributed uniformly at random over $\mathbb{Z}_q^{n \times m}$ with all but negligible probability where $\mathbf{R}_i \leftarrow \left(D_{\mathbb{Z}^m, \sigma}\right)^m$, which follows from Lemma 2.1 and our parameter selections. Concretely, since $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i$ and $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - p^{d-1} \cdot x_i^* \mathbf{G}$ for Game_0 and Game_1 respectively, we have that in both games $\{\mathbf{B}_i\}_{i \in [\ell]}$ is distributed statistically close to uniform over $\left(\mathbb{Z}_q^{n \times m}\right)^{\ell}$.

We now proceed to prove that the value of (A) is negligibly close in both games. We first analyze the case for Game_0. Let $\vec{\mathbf{B}}_{\mathsf{view}} \in \mathbb{Z}_q^{n \times m\ell}$ and $\vec{\mathbf{R}} \in \mathbb{Z}^{m \times m\ell}$ denote the matrices $[\mathbf{B}_1 | \cdots | \mathbf{B}_\ell]$ and $[\mathbf{R}_1 | \cdots | \mathbf{R}_\ell]$, respectively. Then, we have $\vec{\mathbf{B}}_{\mathsf{view}} = \mathbf{A}\vec{\mathbf{R}} \mod q$. Furthermore, let $\vec{\mathbf{T}} = [\mathbf{T}_1 | \cdots | \mathbf{T}_\ell] \in \mathbb{Z}^{m \times m\ell}$ be an arbitrary solution to $\vec{\mathbf{B}}_{\mathsf{view}} = \mathbf{A}\vec{\mathbf{T}} \mod q$. Then, due to

Lemma 2.1 and the conditions on $\{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} = \{\mathbf{A}\mathbf{R}_i\}_{i\in[\ell]}$, the conditional distribution of $\vec{\mathbf{R}}$ is given by $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}},\sigma}$. Now, we are ready to determine the conditional distribution of the secret keys $\{\mathbf{R}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}}$ obtained by the adversary $\mathcal{A}$. Here, let $j^*\in[\ell]$ denote the index $[\ell]\backslash\mathsf{list}$ where $|\mathsf{list}| = \ell-1$, and observe the following equation:

$$\underbrace{[\mathbf{R}_{\vec{h}^{(1)}}|\mathbf{R}_{\vec{h}^{(2)}}|\cdots|\mathbf{R}_{\vec{h}^{(\ell-1)}}]}_{:=\vec{\mathbf{R}}_{\mathsf{sk}}\ \in\mathbb{Z}^{m\times m(\ell-1)}} = \underbrace{[\mathbf{R}_1|\mathbf{R}_2|\cdots|\mathbf{R}_\ell]}_{=\vec{\mathbf{R}}\ \in\mathbb{Z}^{m\times m\ell}} \underbrace{\left[\begin{array}{c|c|c|c|c|c} \mathsf{h}_1^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_1^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(\ell-1)}\mathbf{I}_m \\ \mathsf{h}_2^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_2^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(\ell-1)}\mathbf{I}_m \\ \vdots & & \vdots & \vdots & & \vdots \\ \mathsf{h}_\ell^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_\ell^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(\ell-1)}\mathbf{I}_m \end{array}\right]}_{:=\mathbf{M}\ \in\mathbb{Z}^{m\ell\times m(\ell-1)}}, \tag{6.16}$$

where $\mathsf{h}_k^{(j)}$ is the k-th entry of $\vec{\mathsf{h}}^{(j)}$ that is associated with the j-th vector $\vec{h}^{(j)}$ in st for $j\in\mathsf{list}$. We denote the left and right hand matrices as $\vec{\mathbf{R}}_{\mathsf{sk}}\in\mathbb{Z}^{m\times m(\ell-1)}$ and $\mathbf{M}\in\mathbb{Z}^{m\ell\times m(\ell-1)}$ respectively. So as not to interrupt the proof of Theorem 6.3, we intentionally skip the proof for the time being. Later in Lemma 6.5, we show that there exists a matrix $\mathbf{W}\in\mathbb{Z}^{m\ell\times m}$ such that $\mathbf{W}^\top\mathbf{M} = \mathbf{0}$ over $\mathbb{Z}$ with a sufficiently small singular value. Therefore, for our parameter selection and the fact that $\vec{\mathbf{R}}$ is distributed according to $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}},\sigma}$ we can apply Theorem 6.1. Namely, the distribution of $\vec{\mathbf{R}}_{\mathsf{sk}} = \vec{\mathbf{R}}\mathbf{M}$ is statistically close to a distribution parameterized by $\Lambda^\perp(\mathbf{A}),\sigma,\mathbf{M}$ and ($\vec{\mathbf{T}}\mathbf{M}$ mod $\Lambda^\perp(\mathbf{A})^{m\ell}\mathbf{M}$).

We now show that this holds in case for Game_1 as well. We begin by determining the conditional distribution of $\vec{\mathbf{R}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]} = \{\mathbf{A}\mathbf{R}_i - p^{d-1}\cdot x_i^*\mathbf{G}\}_{i\in[\ell]}$. Let us denote $\vec{\mathbf{G}}_{\vec{x}^*}\in\mathbb{Z}_q^{n\times m\ell}$ as the matrix $p^{d-1}\cdot[x_1^*\mathbf{G}|x_2^*\mathbf{G}|\cdots|x_\ell^*\mathbf{G}]$. Then, $\vec{\mathbf{B}}_{\mathsf{view}} + \vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}\vec{\mathbf{R}} \mod q$. Next, let us chose an arbitrary matrix $\mathbf{E}\in\mathbb{Z}^{m\times m}$ such that $\mathbf{G} = \mathbf{A}\mathbf{E} \mod q$, and define $\vec{\mathbf{E}}_{\vec{x}^*}\in\mathbb{Z}^{m\times m\ell}$ as the matrix $p^{d-1}\cdot[x_1^*\mathbf{E}|x_2^*\mathbf{E}|\cdots|x_\ell^*\mathbf{E}]$. Then, we have $\vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}\vec{\mathbf{E}}_{\vec{x}^*} \mod q$. Combining this with the $\vec{\mathbf{T}}$ we have defined above in Game_0, we obtain $\vec{\mathbf{B}}_{\mathsf{view}} + \vec{\mathbf{G}}_{\vec{x}^*} = \mathbf{A}(\vec{\mathbf{T}} + \vec{\mathbf{E}}_{\vec{x}^*}) \mod q$. Therefore, by Lemma 2.1, the conditional distribution of $\vec{\mathbf{R}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]}$ is $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}}+\vec{\mathbf{E}}_{\vec{x}^*},\sigma}$. Next, we determine the conditional distribution of the secret keys $\{\mathbf{R}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}}$ obtained by the adversary $\mathcal{A}$. Observe that equation Eq.(6.16) holds for Game_1 as well, since we do not change the way we answer the key extraction query. Hence, following the same argument as above, by Theorem 6.1 and the fact that $\vec{\mathbf{R}}$ is distributed according to $D_{\Lambda^\perp(\mathbf{A})^{m\ell}+\vec{\mathbf{T}}+\vec{\mathbf{E}}_{\vec{x}^*},\sigma}$, we have that the distribution of $\vec{\mathbf{R}}_{\mathsf{sk}} = \vec{\mathbf{R}}\mathbf{M}$ is statistically close to a distribution parameterized by $\Lambda^\perp(\mathbf{A}),\sigma,\mathbf{M}$ and ($\vec{\mathbf{T}}\mathbf{M}+\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M}$ mod $\Lambda^\perp(\mathbf{A})^{m\ell}\mathbf{M}$).

Finally, we prove that $\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M}\in\Lambda^\perp(\mathbf{A})^{m\ell}\mathbf{M}$ to prove equivalence of the distributions between Game_0 and Game_1. Observe that

$$\begin{aligned}\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} &= p^{d-1}\cdot\mathbf{E}\cdot[x_1^*\mathbf{I}_m|x_2^*\mathbf{I}_m|\cdots|x_\ell^*\mathbf{I}_m]\cdot\left[\begin{array}{c|c|c|c|c|c} \mathsf{h}_1^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_1^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(\ell-1)}\mathbf{I}_m \\ \mathsf{h}_2^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_2^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(\ell-1)}\mathbf{I}_m \\ \vdots & & \vdots & \vdots & & \vdots \\ \mathsf{h}_\ell^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_\ell^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(\ell-1)}\mathbf{I}_m \end{array}\right], \\ &= p^{d-1}\cdot\mathbf{E}\cdot[\langle\vec{x}^*,\vec{\mathsf{h}}^{(1)}\rangle\mathbf{I}_m|\cdots|\langle\vec{x}^*,\vec{\mathsf{h}}^{(j^*-1)}\rangle\mathbf{I}_m|\langle\vec{x}^*,\vec{\mathsf{h}}^{(j^*+1)}\rangle\mathbf{I}_m|\cdots|\langle\vec{x}^*,\vec{\mathsf{h}}^{(\ell-1)}\rangle\mathbf{I}_m] \\ &= q\cdot\mathbf{E}\cdot[n_1\mathbf{I}_m|\cdots|n_{j^*-1}\mathbf{I}_m|n_{j^*+1}\mathbf{I}_m|\cdots|n_{\ell-1}\mathbf{I}_m]\in q\mathbb{Z}^{m\times m(\ell-1)},\end{aligned}$$

where we set $n_j = \langle \vec{x}^*, \vec{\mathsf{h}}^{(j)} \rangle / p \in \mathbb{N}$ for $j \in \mathsf{list}$. Note that this is well-defined since $\langle \vec{x}^*, \vec{\mathsf{h}}^{(j)} \rangle = \langle \vec{x}^*, \vec{h}^{(j)} \rangle = 0 \mod p$ (See Section 6.5.1) and $q = p^d$. Therefore, to prove $\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} \in \Lambda^{\perp}(\mathbf{A})^{m\ell}\mathbf{M}$, it suffices to prove that $q\mathbb{Z}^{m \times m(\ell-1)} \subset \Lambda^{\perp}(\mathbf{A})^{m\ell}\mathbf{M}$. Namely, we prove that for every $\mathbf{Z} \in q\mathbb{Z}^{m\times m(\ell-1)}$, there exists a matrix $\mathbf{V} \in \Lambda^{\perp}(\mathbf{A})^{m\ell} \subset \mathbb{Z}^{m \times m\ell}$ such that $\mathbf{VM} = \mathbf{Z}$ (over $\mathbb{Z}$). Here, recall that for the vectors $\{\vec{h}^{(j)}\}_{j \in \mathsf{list}}$ in the state st, we had $(j = \arg\min_{i \in [\ell]}\{h_i^{(j)} \neq 0\}) \wedge (h_j^{(j)} = 1)$. Namely, the smallest index with a non-zero entry for $\vec{h}^{(j)}$ is j, and at that index we have $h_j^{(j)} = 1$. Therefore, denoting $\mathbf{H} \in \mathbb{Z}^{\ell \times (\ell-1)}$ as the matrix whose columns are the vectors in $\{h^{(j)}\}_{j\in\mathsf{list}}$, we can properly rearrange the columns and rows of $\mathbf{H}$, or more concretely there exists a permutation matrix $\mathbf{P} \in \{0,1\}^{\ell\times\ell}, \mathbf{Q} \in \{0,1\}^{(\ell-1)\times(\ell-1)}$, such that $\mathbf{H}$ gets transformed into the following matrix:

$$\mathbf{PHQ} = \left[\begin{array}{ccccc} \star & & \cdots & \star & \star \\ \hline 1 & 0 & \cdots & \cdots & 0 \\ \star & 1 & \ddots & & \vdots \\ \vdots & \star & \ddots & \ddots & \vdots \\ \vdots & & \ddots & 1 & 0 \\ \star & \star & \cdots & \star & 1 \end{array}\right] = \begin{bmatrix} \mathbf{a}^{\top} \\ \mathbf{U} \end{bmatrix} \in \mathbb{Z}^{\ell\times(\ell-1)}, \tag{6.17}$$

where $\star$ denotes an arbitrary element in $\mathbb{Z}$, $\mathbf{a} \in \mathbb{Z}^{\ell-1}$ is some vector and $\mathbf{U} \in \mathbb{Z}^{(\ell-1)\times(\ell-1)}$ is unimodular. Recall that permutation matrices are orthogonal matrices: $\mathbf{Q}^{-1} = \mathbf{Q}^{\top}$, and that the inverse of a unitary matrix is also unitary: $\mathbf{U}^{-1} \in \mathbb{Z}^{(\ell-1)\times(\ell-1)}$. We now proceed to prove that $\mathbf{V} = [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{QU}^{-1} \otimes \mathbf{I}_m)]\cdot(\mathbf{P}\otimes\mathbf{I}_m) \in \mathbb{Z}^{m\times m\ell}$ satisfies the above condition, i.e., $\mathbf{V} \in \Lambda^{\perp}(\mathbf{A})^{m\ell}$ and $\mathbf{VM} = \mathbf{Z}$ (over $\mathbb{Z}$). First, it is easy to check that $\mathbf{V} \in \Lambda^{\perp}(\mathbf{A})^{m\ell}$, since $\mathbf{Z} \in q\mathbb{Z}^{m\times m(\ell-1)}$ and $q\mathbb{Z}^m \subset \Lambda^{\perp}(\mathbf{A})$. Then, recalling that $\mathbf{M} = \mathbf{H} \otimes \mathbf{I}_m$, we have

$$\begin{aligned} \mathbf{VM} &= \Big([\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{QU}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)\Big)\cdot(\mathbf{H}\otimes\mathbf{I}_m) \\ &= \Big([\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{QU}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)\Big)\cdot\left(\mathbf{P}^{\top}\begin{bmatrix}\mathbf{a}^{\top}\\ \mathbf{U}\end{bmatrix}\mathbf{Q}^{\top}\right)\otimes\mathbf{I}_m \qquad (6.18) \\ &= [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{QU}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)(\mathbf{P}^{\top}\otimes\mathbf{I}_m)\left(\begin{bmatrix}\mathbf{a}^{\top}\mathbf{Q}^{\top}\\ \mathbf{UQ}^{\top}\end{bmatrix}\otimes\mathbf{I}_m\right) \qquad (6.19) \\ &= [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{QU}^{-1}\otimes\mathbf{I}_m)]\begin{bmatrix}\mathbf{a}^{\top}\mathbf{Q}^{\top}\otimes\mathbf{I}_m\\ \mathbf{UQ}^{\top}\otimes\mathbf{I}_m\end{bmatrix} \qquad (6.20) \\ &= \mathbf{Z}, \qquad (6.21) \end{aligned}$$

where Eq. (6.18) follows from Eq. (6.17), Eq. (6.19) follows from the fact that $(\mathbf{AB}\otimes\mathbf{I}_m) = (\mathbf{A}\otimes\mathbf{I}_m)(\mathbf{B}\otimes\mathbf{I}_m)$ and Eq. (6.20),(6.21) follows from the fact that $\mathbf{P}, \mathbf{Q}$ are orthogonal matrices. Therefore, we have $\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} \in \Lambda^{\perp}(\mathbf{A})^{m\ell}\mathbf{M}$.

Hence, we conclude that the value of (A), i.e., the conditional probability of $\vec{\mathbf{R}}_{\mathsf{sk}}$ given $\{\mathbf{B}_i\}_{i\in[\ell]}$ in Game_0 and Game_1 are negligibly close. Therefore, we have $|\Pr[S_0] - \Pr[S_1]| = \mathsf{negl}(n)$.

Game_2 : In this game, we change the way the challenge ciphertext is created. Recall that in the previous game, the challenge ciphertext was created as

$$c = p^{d-1}\cdot\left(\mathbf{u}^{\top}\mathbf{s} + \mathsf{M}_b\lfloor p/2\rceil\right), \quad \mathbf{c}_0 = \mathbf{A}^{\top}\mathbf{s} + \mathbf{z}_0, \quad (\mathbf{c}_i = (\mathbf{AR}_i)^{\top}\mathbf{s} + \mathbf{z}_i)_{i\in[\ell]} \tag{6.22}$$

where $\mathbf{s} \leftarrow \mathbb{Z}_q^n$, $\mathbf{z}_0, \mathbf{z}_i \leftarrow D_{\mathbb{Z}^m,\alpha' q}$ for $i \in [\ell]$, and $b \leftarrow \{0,1\}$. Note the term $(\mathbf{c}_i)_{i\in[\ell]}$ follows from the fact that in Game_1 we modified $\mathbf{B}_i$ so that $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - p^{d-1} \cdot x_i^* \mathbf{G}$, and $\mathsf{M}_0, \mathsf{M}_1$ are the two messages sent by the adversary $\mathcal{A}$. To create the challenge ciphertext in Game_2, the challenger first picks $\mathbf{s} \leftarrow \mathbb{Z}_q^n$ and $\mathbf{z} \leftarrow D_{\mathbb{Z}^m,\alpha q}$ and computes $\mathbf{v} = \mathbf{A}^\top \mathbf{s} + \mathbf{z} \in \mathbb{Z}_q^m$. It then runs the algorithm

$$\mathsf{ReRand}\Big([\mathbf{I}_m|\vec{\mathbf{R}}], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}\Big) \to \mathbf{c} \in \mathbb{Z}_q^{m(\ell+1)}$$

from Lemma 2.6, and parses $\mathbf{c}$ into $\ell+1$ vectors $(\mathbf{c}_i)_{i\in[\ell+1]}$ in $\mathbb{Z}_q^m$ such that $\mathbf{c}^\top = [\mathbf{c}_0^\top|\mathbf{c}_1^\top|\cdots|\mathbf{c}_\ell^\top] \in \mathbb{Z}_q^{m(\ell+1)}$. Finally, it picks $b \leftarrow \{0,1\}$ and sets the challenge ciphertext as

$$C^* = \Big(c = v + \mathsf{M}_b \cdot p^{d-1}\lfloor p/2 \rceil, \quad \mathbf{c}_0, \quad (\mathbf{c}_i)_{i\in[\ell]}\Big) \in \mathbb{Z}_q \times \mathbb{Z}_q^m \times (\mathbb{Z}_q^m)^\ell, \tag{6.23}$$

where $v = p^{d-1} \cdot \mathbf{u}^\top \mathbf{s}$.

We claim that this change alters the view of $\mathcal{A}$ only negligibly. First, observe c is distributed identically as in Eq.(6.22). Next, observe that the input to ReRand is $[\mathbf{I}_m|\vec{\mathbf{R}}] \in \mathbb{Z}^{m\times m(\ell+1)}$ and $\mathbf{v} = \mathbf{A}^\top \mathbf{s} + \mathbf{z} \in \mathbb{Z}_q^m$. Therefore, due to Lemma 2.6, for our choices of α and α', the output of ReRand is

$$\begin{aligned} \mathbf{c}^\top &= \big(\mathbf{A}^\top \mathbf{s}\big)^\top [\mathbf{I}_m|\vec{\mathbf{R}}] + \mathbf{z}'^\top \\ &= \mathbf{s}^\top [\mathbf{A}|\mathbf{A}\vec{\mathbf{R}}] + \mathbf{z}'^\top \quad \in \mathbb{Z}_q^{m(\ell+1)}, \end{aligned}$$

where the distribution of $\mathbf{z}'$ is within statistical distance from $\mathbf{z}' \leftarrow D_{\mathbb{Z}^{m(\ell+1)},\alpha' q}$. By parsing $\mathbf{c}$ appropriately as above, it can be seen that it is statistically close to $\big(\mathbf{c}, (\mathbf{c}_i)_{i\in[\ell]}\big)$ of Eq.(6.22). Therefore, the challenge ciphertexts of Game_1 and Game_2 are statistically indistinguishable. Hence, we have $|\Pr[S_1] - \Pr[S_2]| = \mathsf{negl}(n)$.

Game_3 : In this game, we further change the way the challenge ciphertext is created. To create the challenge ciphertext, the challenger first samples $v \leftarrow p^{d-1}\mathbb{Z}/q\mathbb{Z}$ (i.e., $\{a \mid p^{d-1} \cdot a, \forall a \in \mathbb{Z}_q\}$), $\mathbf{v}' \leftarrow \mathbb{Z}_q^m$ and $\mathbf{z} \leftarrow D_{\mathbb{Z}^m,\alpha q}$, and runs

$$\mathsf{ReRand}\Big([\mathbf{I}_m|\vec{\mathbf{R}}], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}\Big) \to \mathbf{c} \in \mathbb{Z}_q^{m(\ell+1)},$$

where $\mathbf{v} = \mathbf{v}' + \mathbf{z}$. Then, the challenge ciphertext is set as in Eq.(6.23). We show below in Lemma 6.6 that by assuming $\mathsf{FE.LWE}_{n,m+1,q,\chi}$ is hard, we have $|\Pr[S_2] - \Pr[S_3]| = \mathsf{negl}(n)$.

Furthermore, since v is uniformly random over $p^{d-1}\mathbb{Z}/q\mathbb{Z}$ and independent of the other values, the term in the challenge ciphertext $c = v + \mathsf{M}_b \cdot p^{d-1}\lfloor p/2 \rceil \in p^{d-1}\mathbb{Z}/q\mathbb{Z}$ that conveys the information on the message is distributed independently from the value of M_b. Therefore, we have $\Pr[S_3] = 1/2$. Combining everything together, we have

$$\begin{aligned} \left|\Pr[S_0] - \frac{1}{2}\right| &= \left|\sum_{i=0}^{2} (\Pr[S_i] - \Pr[S_{i+1}]) + \Pr[S_3] - \frac{1}{2}\right| \\ &\le \sum_{i=0}^{2} |\Pr[S_i] - \Pr[S_{i+1}]| + \left|\Pr[S_3] - \frac{1}{2}\right| \le \mathsf{negl}(n). \end{aligned}$$

Therefore, the probability that $\mathcal{A}$ wins Game_0 is negligible. $\square$

To complete the proof of Theorem 6.3, it remains to prove Lemma 6.5 and Lemma 6.6.

Lemma 6.5. *There exits a full rank matrix* $\mathbf{W} \in \mathbb{Z}^{m\ell \times m}$ *such that* $\mathbf{W}^\top \mathbf{M} = \mathbf{0}$ *(over* $\mathbb{Z}$*) and* $\sqrt{s_1(\mathbf{W}^\top\mathbf{W})} \le (p+1)^{\ell+2}$, *where* $\mathbf{M} \in \mathbb{Z}^{m\ell \times m(\ell-1)}$ *is the full rank matrix defined in Eq. (6.16).*

Proof. Here, we use the matrices and vector $\mathbf{P}, \mathbf{Q}, \mathbf{U}, \mathbf{a}$ defined in Eq. (6.17). First, let $\mathbf{w} \in \mathbb{Z}^\ell$ be a non-zero vector such that $\mathbf{w}^\top\mathbf{PHQ} = \mathbf{0}$ (over $\mathbb{Z}$). Then, we can set $\mathbf{W} = (\mathbf{P}^\top\mathbf{w}) \otimes \mathbf{I}_m \in \mathbb{Z}^{m\ell\times m}$. It is easy to check that $\mathbf{W}$ is rank m and satisfies

$$\mathbf{W}^\top\mathbf{M} = \left((\mathbf{P}^\top\mathbf{w}) \otimes \mathbf{I}_m\right)^\top \cdot \mathbf{M} = \left((\mathbf{w}^\top\mathbf{P}) \otimes \mathbf{I}_m\right) \cdot (\mathbf{H} \otimes \mathbf{I}_m) = \left((\mathbf{w}^\top\mathbf{PHQ}) \cdot \mathbf{Q}^\top\right) \otimes \mathbf{I}_m = \mathbf{0},$$

where we have $\mathbf{Q}\mathbf{Q}^\top = \mathbf{I}_{\ell-1}$ due to the fact that $\mathbf{Q}$ is a permutation matrix. Furthermore, by the way we construct $\mathbf{W}$, we have

$$\mathbf{W}^\top\mathbf{W} = (\mathbf{P}^\top\mathbf{w} \otimes \mathbf{I}_m)^\top(\mathbf{P}^\top\mathbf{w} \otimes \mathbf{I}_m) = \mathbf{w}^\top\mathbf{w} \otimes \mathbf{I}_m = \mathbf{w}^\top\mathbf{w} \cdot \mathbf{I}_m.$$

Therefore $s_1(\mathbf{W}^\top\mathbf{W}) = \mathbf{w}^\top\mathbf{w}$. Hence, it suffices to prove that there exists $\mathbf{w} \in \mathbb{Z}^\ell$ such that $\mathbf{w}^\top\mathbf{PHQ} = \mathbf{0}$ and $\|\mathbf{w}\| \le 3p^\ell$. Recalling Eq. (6.17), we have

$$(\mathbf{w}^\top\mathbf{PHQ})^\top = \left[\begin{array}{c|ccccc} a_1 & 1 & u_{1,1} & \cdots & \cdots & u_{1,\ell-1} \\ a_2 & 0 & 1 & u_{2,2} & \cdots & u_{2,\ell-1} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ a_{\ell-2} & \vdots & & \ddots & 1 & u_{\ell-1,\ell-1} \\ a_{\ell-1} & 0 & \cdots & \cdots & 0 & 1 \end{array}\right] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{\ell-1} \\ w_\ell \end{bmatrix} = \mathbf{0} \in \mathbb{Z}^{\ell-1}, \tag{6.24}$$

where $a_i, u_{i,k} \in \mathbb{Z}$ for $i \in [\ell-1], k \in [i]$. Furthermore, since $\mathbf{P}, \mathbf{Q}$ are permutation matrices and all elements of $\mathbf{H}$ were chosen from $\mathbb{Z}_p$ (See the KeyGen algorithm), we have $|a_i|, |u_{i,k}| < p$. Now, from Eq. (6.24), if we set $w_1 = 1$ we can solve for the other $\{w_i\}_{i=2}^{\ell}$ terms recursively as follows:

$$\begin{cases} w_\ell &= -a_{\ell-1}w_1 \\ w_{\ell-1} &= -a_{\ell-2}w_1 - u_{\ell-1,\ell-1}w_{\ell-1} \\ &\vdots \\ w_2 &= -a_1w_1 - \sum_{i=1}^{\ell-1} u_{1,i}w_{i+1} \end{cases}$$

Since, $w_1 = 1$ and $|a_i|, |u_{i,k}| < p$ for all $i \in [\ell-1], k \in [i]$, we have $|w_i| \le \sum_{t=1}^{\ell+1-i} p|w_t| \le p(p+1)^{\ell+1-i}$. Therefore, we have

$$\mathbf{w}^\top\mathbf{w} = \sum_{i=1}^{\ell} w_i^2 \le \sum_{i=1}^{\ell} p^2(p+1)^{2(\ell+1-i)} \le (p+1)^{2(\ell+2)}.$$

Thus, we conclude that $\sqrt{s_1(\mathbf{W}^\top\mathbf{W})} = \|\mathbf{w}\| \le (p+1)^{\ell+2}$. □

Lemma 6.6. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[S_2] - \Pr[S_3]| \le \mathsf{Adv}_{\mathcal{B}}^{\mathsf{FE.LWE}_{n,m+1,q,\chi}}.$$

In particular, under the $\mathsf{FE.LWE}_{n,m+1,q,\chi}$ *assumption, we have* $|\Pr[S_2] - \Pr[S_3]| = \mathsf{negl}(n)$.

Proof. Suppose there exists an adversary $\mathcal{A}$ with non-negligible advantage in distinguishing between Game_2 and Game_3 that outputs a value $\mathsf{coin} \in \{0,1\}$, where $\mathsf{coin} = 1$ in case $\mathcal{A}$ decides its interacting with a Game_2 challenger. We use $\mathcal{A}$ to construct an $\mathsf{FE.LWE}$ algorithm $\mathcal{B}$ as follows.

Instance. $\mathcal{B}$ is given $\{\mathbf{a}_i, v_i\}_{i=0}^m \in \left(\mathbb{Z}_q^n \times \mathbb{Z}_q\right)^{m+1}$ as the problem instance of $\mathsf{FE.LWE}_{n,m+1,q,\chi}$, where recall that $\chi = D_{\mathbb{Z},\alpha q}$ and the first term is errorless, i.e., $v_0 = \mathbf{a}_0^\top \mathbf{s}$ in case of a valid $\mathsf{FE.LWE}$ sample. We can assume without loss of generality that $v_i = v_i' + z_i$ for $z_i \leftarrow D_{\mathbb{Z},\alpha q}$ $(i \in [m])$ and restate the $\mathsf{FE.LWE}$ problem so that $\mathcal{B}$'s task is now to distinguish whether $v_i' = \mathbf{a}_i^\top \mathbf{s}$ for some $\mathbf{s} \in \mathbb{Z}_q^n$ or $v_i' \leftarrow \mathbb{Z}_q$ for $i \in [0,m]$. We note this subtle change from the standard $\mathsf{FE.LWE}$ problem is only a syntactical change made for the convenience of the proof.

Setup. To construct the master public key MPK, $\mathcal{B}$ first sets the random vector $\mathbf{u}$ as $\mathbf{a}_0$, and assembles the random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ from the remaining $\mathsf{FE.LWE}$ samples $\{\mathbf{a}_i\}_{i=1}^m$ by letting the i-th column be the vector $\mathbf{a}_i$. It also samples ℓ random matrices $\mathbf{R}_i \leftarrow (D_{\mathbb{Z}^m,\sigma})^m$ and sets $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - p^{d-1} \cdot x_i^* \mathbf{G} \mod q$ for $i \in [\ell]$. Finally, it returns $\mathsf{MPK} = (\mathbf{A}, \{\mathbf{B}_i\}_{i\in\ell}, \mathbf{u})$ to $\mathcal{A}$.

Phase 1 and Phase 2. The key extraction queries made by $\mathcal{A}$ are answered as in Game_1 (which is equivalent to both Game_2 and Game_3), using the $\mathbf{R}_i$'s and $\mathbf{R}_i'$'s created during Setup.

Challenge Query. When $\mathcal{A}$ makes the challenge query for the challenge attribute vector $\vec{x}^*$ and challenge messages $\mathsf{M}_0, \mathsf{M}_1$, $\mathcal{B}$ sets the challenge ciphertext C^* as in Eq.(6.23) and returns C^* to $\mathcal{A}$.

Guess. At last, $\mathcal{A}$ outputs its guess coin. Then, $\mathcal{B}$ outputs 1 if $\mathsf{coin} = 1$ and 0 otherwise.

Analysis. It can be seen that $\mathcal{B}$ perfectly simulates the view of $\mathcal{A}$ in Game_2 if $\{\mathbf{a}_i, v_i\}_{i=0}^m$ are valid $\mathsf{FE.LWE}$ samples (i.e., $v_i' = \mathbf{a}_i^\top \mathbf{s}$) and Game_3 otherwise (i.e., $v_i' \leftarrow \mathbb{Z}_q$). We therefore conclude that $\mathsf{Adv}_{\mathcal{B}}^{\mathsf{FE.LWE}_{n,m+1,q,\chi}} = |\Pr[S_2] - \Pr[S_3]|$ as desired.

□

6.6 A Generic Construction of NIPE from LinFE

In this section, we show a generic conversion from a functional encryption scheme for inner products to a NIPE scheme. We note that the former primitive is a special case of the notion of functional encryption schemes where only linear functions are available. Henceforth we call this primitive as LinFE in the following. The idea for the conversion is drawn from the work of Agrawal et al. [ABP+17], who constructed trace and revoke schemes from LinFE.

6.6.1 Definition of Functional Encryption for Inner Product

Syntax. Let $\mathcal{Q}$ and $\mathcal{J}$ denote the predicate space and attribute spaces, where the inner product between elements (i.e., vectors) from $\mathcal{Q}$ and $\mathcal{J}$ are well-defined. Furthermore, let $\mathcal{D}$ denote the space where the inner product is taken. A *stateful* functional encryption scheme for inner products over $\mathcal{D}$ consists of the following four algorithms:

$\mathsf{Setup}(1^\lambda, 1^\ell) \to (\mathsf{MPK}, \mathsf{MSK}, \mathsf{st})$: The setup algorithm takes as input a security parameter 1^λ and the length ℓ of the vectors in the predicate and an attribute spaces, and outputs a master public key MPK, a master secret key MSK and an initial state st.

$\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \vec{y}) \to (\mathsf{sk}_{\vec{y}}, \mathsf{st})$: The key generation algorithm takes as input the master public key MPK, the master secret key MSK, the state st and a predicate vector $\vec{y} \in \mathcal{Q}$. It outputs a private key $\mathsf{sk}_{\vec{y}}$ and a updated state st. We assume that $\vec{y}$ is implicitly included in $\mathsf{sk}_{\vec{y}}$.

$\mathsf{Encrypt}(\mathsf{MPK}, \vec{x}) \to \mathsf{ct}$: The encryption algorithm takes as input a master public key MPK and attribute vector $\vec{x} \in \mathcal{J}$. It outputs a ciphertext ct.

$\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{ct}) \to \langle \vec{x}, \vec{y} \rangle$or $\perp$: The decryption algorithm takes as input the master public key MPK, a private key $\mathsf{sk}_{\vec{y}}$, and a ciphertext ct. It outputs $\langle \vec{x}, \vec{y} \rangle$ or $\perp$, which means that the ciphertext is not in a valid form.

Correctness. We require correctness of decryption: that is, for all $\lambda, \ell \in \mathbb{N}$, and all $\vec{x} \in \mathcal{J}, \vec{y} \in \mathcal{Q}$, we require

$$\Pr[\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \langle \vec{x}, \vec{y} \rangle] = 1 - \mathsf{negl}(\lambda)$$

holds, where the probability is taken over the randomness used in $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$, $(\mathsf{sk}_{\vec{y}}, \mathsf{st}) \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \vec{y})$, and $\mathsf{Encrypt}(\mathsf{MPK}, \vec{x})$.

We also define a *stateless* LinFE scheme, where we do not require any state information in the above algorithms.

Security. We define the security of a (stateful) LinFE scheme for inner product space D with predicate space $\mathcal{Q}$ and attribute space $\mathcal{J}$ by the following game between a challenger and an adversary $\mathcal{A}$.

- Setup. At the outset of the game, the challenger runs $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives the public parameter MPK to $\mathcal{A}$.

- Phase 1. $\mathcal{A}$ may adaptively make key-extraction queries. If $\mathcal{A}$ submits a predicate vector $\vec{y} \in \mathcal{Q}$ to the challenger, the challenger runs $(\mathsf{sk}_{\vec{y}}, \mathsf{st}) \leftarrow \mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}, \vec{y})$ and returns $\mathsf{sk}_{\vec{y}}$ to $\mathcal{A}$.

- Challenge Phase. At some point, $\mathcal{A}$ outputs messages $\vec{x}_0^*, \vec{x}_1^*$ on which it wishes to be challenged, with the restriction that $\langle \vec{x}_0^*, \vec{y} \rangle = \langle \vec{x}_1^*, \vec{y} \rangle$ (over $\mathcal{D}$) for all $\vec{y}$ queried during Phase 1. Then, the challenger picks a random bit $b \in \{0, 1\}$ and returns $C^* \leftarrow \mathsf{Encrypt}(\mathsf{MPK}, \vec{x}_b^*)$ to $\mathcal{A}$.

- Phase 2. After the challenge query, $\mathcal{A}$ may continue to make key-extraction queries for predicate vectors $\vec{y} \in \mathcal{Q}$, with the added restriction that $\langle \vec{x}_0^*, \vec{y} \rangle = \langle \vec{x}_1^*, \vec{y} \rangle$ (over $\mathcal{D}$).

- Guess. Finally, $\mathcal{A}$ outputs a guess b' for b. The advantage of $\mathcal{A}$ is defined as

$$\mathsf{Adv}_{\mathcal{A},\mathcal{D}}^{\mathsf{LinFE}} = \left| \Pr[b' = b] - \frac{1}{2} \right|.$$

We say that an LinFE scheme with inner product space $\mathcal{D}$ is *adaptively secure*, if the advantage of any PPT $\mathcal{A}$ is negligible. Similarly, we define *selective security* for a stateful LinFE scheme with inner product space $\mathcal{D}$, by modifying the above game so that the adversary $\mathcal{A}$ is forced to declare its challenge attribute vectors $\vec{x}_0^*, \vec{x}_1^*$ before **Setup**. Finally, we define an analogous security notion for stateless LinFE schemes, where we do not require any state information during the above game.

6.6.2 Generic Construction of NIPE from LinFE

Here, we show a generic construction of NIPE from LinFE. Specifically, we convert a LinFE scheme with predicate space $\mathcal{Q}$, attribute space $\mathcal{J}$ with inner product space D into an NIPE scheme over D with predicate space $\mathcal{P}$, attribute space $\mathcal{I}$, and message space $\mathcal{M}$. The conversion is possible when the following properties are satisfied:

- We require $\mathcal{P}, \mathcal{Q}, \mathcal{I}, \mathcal{J} \subseteq \mathcal{D}^\ell$ and $\mathcal{M} \subseteq \mathcal{D}$ for some integral domain $\mathcal{D}$.
- We also require $\{ \mathsf{M} \cdot \vec{x} \mid \mathsf{M} \in \mathcal{M},\ \vec{x} \in \mathcal{I} \} \subseteq \mathcal{J}$ and $\mathcal{P} = \mathcal{Q}$.

- Division can be efficiently performed over $\mathcal{D}$. More specifically, we require that given $\alpha, \beta \in \mathcal{D}$, it is possible to efficiently compute $\gamma \in \mathcal{D}$ satisfying $\alpha = \beta\gamma$ if such γ exists.

We now show the construction. Note that the conversion works both for the stateless and stateful cases. Let $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ be the underlying LinFE scheme and $(\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}', \mathsf{Dec}')$ be the resulting NIPE scheme.

$\mathsf{Setup}'(1^\lambda, 1^\ell)$: It is the same as $\mathsf{Setup}(1^\lambda, 1^\ell)$.

$\mathsf{KeyGen}'(\mathsf{MPK}, \mathsf{MSK}, \vec{y} \in \mathcal{P}, \mathsf{st})$: It is the same as $\mathsf{KeyGen}(\mathsf{MPK}, \mathsf{MSK}, \vec{y} \in \mathcal{P}, \mathsf{st})$.

$\mathsf{Enc}'(\mathsf{MPK}, \vec{x} \in \mathcal{I}, \mathsf{M} \in \mathcal{M})$: To encrypt a message $\mathsf{M} \in \mathcal{M}$ for an attribute $\vec{x} = (x_1, \cdots, x_\ell) \in \mathcal{I}$, it runs $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{M} \cdot \vec{x})$ and outputs ct.

$\mathsf{Dec}'(\mathsf{MPK}, (\vec{y}, \mathsf{sk}_{\vec{y}}), (\vec{x}, C))$: To decrypt a ciphertext ct with an associating attribute $\vec{x} \in \mathcal{I}$ using a secret key $\mathsf{sk}_{\vec{y}}$ with an associating predicate $\vec{y} \in \mathcal{P}$, it first computes $z = \mathsf{Dec}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{ct})$. It then computes $\langle \vec{x}, \vec{y} \rangle$ and outputs $\perp$ if $\langle \vec{x}, \vec{y} \rangle = 0$ over $\mathcal{D}$. Otherwise, it outputs $z / \langle \vec{x}, \vec{y} \rangle$. Note that the final step is possible because of the requirement on $\mathcal{D}$.

Correctness. Due to the requirements on the domains, we have $\mathsf{M} \cdot \vec{x} \subseteq \mathcal{J}$ and $\vec{y} \in \mathcal{Q} = \mathcal{P}$. Therefore, by the correctness of the underlying LinFE scheme, we have $z = \langle \mathsf{M} \cdot \vec{x}, \vec{y} \rangle = \mathsf{M} \cdot \langle \vec{x}, \vec{y} \rangle$ with overwhelming probability. Thus, the correctness of the resulting NIPE scheme follows.

Theorem 6.4. *If the underlying LinFE scheme is adaptively secure, so is the above NIPE scheme.*

Proof. Suppose there exists an adversary $\mathcal{A}$ against the NIPE scheme that has non-negligible advantage. We use $\mathcal{A}$ to construct another adversary $\mathcal{B}$ against the underlying LinFE scheme as follows.

- Setup. At the outset of the game, the challenger runs $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives the public parameter MPK to $\mathcal{B}$. $\mathcal{B}$ passes the same MPK to $\mathcal{A}$.

- Phase 1. When $\mathcal{A}$ makes a key-extraction query for a vector $\vec{y}$, $\mathcal{B}$ submits the same $\vec{y}$ to its challenger and is given $\mathsf{sk}_{\vec{y}}$. Then, it passes the same $\mathsf{sk}_{\vec{y}}$ to $\mathcal{A}$.

- Challenge Phase. When $\mathcal{A}$ outputs the messages $(\mathsf{M}_0, \mathsf{M}_1)$ and the challenge attribute $\vec{x}^*$ on which it wishes to be challenged, $\mathcal{B}$ submits $(\mathsf{M}_0 \cdot \vec{x}^*, \mathsf{M}_1 \cdot \vec{x}^*)$ to its challenger and receives the challenge ciphertext C^*. $\mathcal{B}$ passes the same C^* to $\mathcal{A}$.

- Phase 2. It is the same as **Phase 1**.

- Guess. Finally, $\mathcal{A}$ outputs a guess b'. $\mathcal{B}$ outputs the same bit as its guess.

Analysis. We first show that $\mathcal{B}$ does not violate the restriction of the security game as long as $\mathcal{A}$ does not. To see this, observe that

$$\langle \mathsf{M}_0 \cdot \vec{x}^*, \vec{y} \rangle = \mathsf{M}_0 \cdot \langle \vec{x}^*, \vec{y} \rangle = 0 = \mathsf{M}_1 \cdot \langle \vec{x}^*, \vec{y} \rangle = \langle \mathsf{M}_1 \cdot \vec{x}^*, \vec{y} \rangle$$

holds for all $\mathbf{y}$ that is queried during the game. Here, the second and the third equalities follow from the restrictions on the queries posed on $\mathcal{A}$. It is clear that $\mathcal{B}$'s simulation for $\mathcal{A}$ is perfect and $\mathcal{B}$'s advantage is exactly the same as $\mathcal{A}$. This concludes the proof of the theorem. □

One may expect that the above proof works also in the selective setting (i.e., if we start from a selectively secure LinFE, we obtain a selectively secure NIPE). However, interestingly we require to modify the proof to work in the selective setting. In particular, in the selective setting, the

LinFE adversary $\mathcal{B}$ above has to declare its target $(\mathsf{M}_0\vec{x}^*, \mathsf{M}_1\vec{x}^*)$ at the beginning of the game. However, since the NIPE adversary $\mathcal{A}$ only declares $\vec{x}^*$ at the outset and decides $(\mathsf{M}_0, \mathsf{M}_1)$ later in the game, it is difficult for $\mathcal{B}$ to correctly decide its target. One way to circumvent this problem is to restrict the message space $\mathcal{M}$ to be of polynomial size and change the proof so that $\mathcal{B}$ simply guesses $(\mathsf{M}_0, \mathsf{M}_1)$. The probability of $\mathcal{B}$ correctly guessing the values is noticeable due to the restriction on the size of the message space, which will be enough for our purpose. The drawback of the approach is that we can only encrypt short messages of logarithmic length. To encrypt a longer message, one needs to run the encryption algorithm many times to encrypt each chunk of the message. Formally, we have the following theorem.

Theorem 6.5. *Let us assume that the size of the message space $\mathcal{M}$ is polynomially bounded. Then, if the underlying LinFE scheme is selectively secure, so is the above NIPE scheme.*

Proof. Suppose there exists an adversary $\mathcal{A}$ against the NIPE scheme that has non-negligible advantage. We use $\mathcal{A}$ to construct another adversary $\mathcal{B}$ against the underlying LinFE scheme as follows.

- Initial Phase. At the outset of the game, $\mathcal{A}$ declares its target vector $\vec{x}^*$ on which it wishes to be challenged. Then, $\mathcal{B}$ randomly picks $\widehat{\mathsf{M}}_0, \widehat{\mathsf{M}}_1 \leftarrow \mathcal{M}$ and declares $(\vec{x}^*\widehat{\mathsf{M}}_0, \vec{x}^*\widehat{\mathsf{M}}_1)$ as its target.

- Setup. Then, the challenger runs $(\mathsf{MPK}, \mathsf{MSK}, \mathsf{st}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives the public parameter MPK to $\mathcal{B}$. $\mathcal{B}$ passes the same MPK to $\mathcal{A}$.

- Phase 1. When $\mathcal{A}$ makes a key-extraction query for a vector $\vec{y}$, $\mathcal{B}$ submits the same $\vec{y}$ to its challenger and is given $\mathsf{sk}_{\vec{y}}$. Then, it passes the same $\mathsf{sk}_{\vec{y}}$ to $\mathcal{A}$.

- Challenge Phase. When $\mathcal{A}$ outputs the messages $(\mathsf{M}_0, \mathsf{M}_1)$, $\mathcal{B}$ proceeds as follows. If $(\widehat{\mathsf{M}}_0, \widehat{\mathsf{M}}_1) \neq (\mathsf{M}_0, \mathsf{M}_1)$, $\mathcal{B}$ aborts and outputs a random bit. Otherwise, $\mathcal{B}$ queries the challenge ciphertext for its challenger to obtain C^*. Then it passes the same C^* to $\mathcal{A}$.

- Phase 2. It is the same as **Phase 1**.

- Guess. Finally, $\mathcal{A}$ outputs a guess b'. $\mathcal{B}$ outputs the same bit as its guess.

Analysis. We first observe that $\mathcal{B}$ does not violate the restriction of the security game as long as $\mathcal{A}$ does not. We then evaluate the advantage of $\mathcal{B}$. In the following, we denote the event of $\mathcal{B}$ correctly guessing $(\widehat{\mathsf{M}}_0, \widehat{\mathsf{M}}_0)$ by guess. Then, it is easy to see that $\mathcal{B}$'s simulation for $\mathcal{A}$ is perfect when guess occurs. Otherwise, $\mathcal{B}$ outputs a random bit. Therefore, we have

$$\begin{aligned}
& \left|\Pr[\mathcal{B} \text{ outputs } b] - \frac{1}{2}\right| \\
= \; & |\Pr[\mathsf{guess}] \cdot \Pr[\mathcal{B} \text{ outputs } b \mid \mathsf{guess}] + \Pr[\neg\mathsf{guess}] \cdot \Pr[\mathcal{B} \text{ outputs } b \mid \neg\mathsf{guess}]| \\
= \; & \left|\frac{1}{|\mathcal{M}|^2} \cdot \Pr[\mathcal{A} \text{ outputs } b] + \frac{1}{2} \cdot \left(1 - \frac{1}{|\mathcal{M}|^2}\right) - \frac{1}{2}\right| \\
= \; & \frac{1}{|\mathcal{M}|^2}\left|\Pr[\mathcal{A} \text{ outputs } b] - \frac{1}{2}\right|,
\end{aligned}$$

which is non-negligible because $\mathcal{A}$'s advantage is non-negligible and $\mathcal{M}$ is of polynomial size. This completes the proof of the theorem. □

6.6.3 Instantiations

By applying the conversion to the existing adaptively secure LinFE schemes of [ABDCP15, ALS16], we obtain several new NIPE schemes. Since the result of [ALS16] subsumes that of

[ABDCP15] in the sense that the former achieves adaptive security whereas the latter achieves selective security, we discuss new schemes obtained by applying our conversion to the former schemes. This results in new adaptively secure NIPE schemes from the LWE assumption, the DDH assumption, and the DCR assumption. In particular, our DDH and DCR instantiations are the first constructions of NIPE schemes without bilinear maps or lattices. One thing to note is that the resulting scheme obtained by our conversion can only deal with logarithmic-size message space when $\mathcal{D}$ is of polynomial size and in order to encrypt a longer message, one needs to separate the message into chunks and run the encryption algorithm multiple times to encrypt each of them.

Construction from the LWE Assumption. In [ALS16], the authors proposed two LinFE schemes from lattices. One is in the stateless setting where the inner product is taken over $\mathbb{Z}$, and the other one is in the stateful setting where the inner product is taken over $\mathbb{Z}_p$ for some prime p. To apply the conversion to the former scheme, we set $\mathcal{D} = \mathbb{Z}$, $\mathcal{P} = \mathcal{Q} = \{0, \ldots, P-1\}^\ell$, $\mathcal{I} = \{0, \ldots, I-1\}^\ell$, $\mathcal{M} = \{0, \ldots, M-1\}$ and $\mathcal{J} = \{0, \ldots, MI-1\}$ for (polynomially bounded) integers P, I, M. It is straightforward to see that these domains satisfy our conditions for the conversion. This results in a stateless NIPE scheme over $\mathbb{Z}$. To apply the conversion to the latter scheme, we set $\mathcal{D} = \mathbb{Z}_p$, $\mathcal{P} = \mathcal{Q} = \mathcal{I} = \mathcal{J} = \mathbb{Z}_p^\ell$, and $\mathcal{M} = \mathbb{Z}_p$. It is also easy to see that these domains satisfy our condition for the conversion. This results in a stateful NIPE scheme over $\mathbb{Z}_p$. Since the original scheme is adaptively secure under the LWE assumption with sub-exponential approximation factors, so is our scheme obtained by the conversion. Compared to our direct construction in Section 6.5, the main advantage of the resulting scheme is that it achieves adaptive security.

Construction from the DDH Assumption. In [ALS16], the authors proposed a stateless LinFE scheme from the DDH assumption. In the scheme, the inner product is taken over $\mathbb{Z}_q$, where q is the order of the underlying group $\mathbb{G}$. One subtlety regarding their scheme is that the decryption algorithm is efficient only when the inner product $\langle \vec{x}, \vec{y} \rangle$ is polynomially bounded. This is because the decryption algorithm first recovers $g^{\langle \vec{x}, \vec{y} \rangle}$ for the generator g of $\mathbb{G}$ and then retrieves $\langle \vec{x}, \vec{y} \rangle$ by solving the discrete logarithm problem. Due to this problem, we cannot apply the conversion in a completely black box manner and some modification is needed. To apply our conversion to their scheme, we set $\mathcal{D} = \mathbb{Z}_q$, $\mathcal{P} = \mathcal{Q} = \mathcal{I} = \mathcal{J} = \mathbb{Z}_q^\ell$, and $\mathcal{M} = \{0, 1, \ldots, M\}$ for polynomially bounded M. Then, $(\mathsf{Setup}', \mathsf{KeyGen}', \mathsf{Enc}')$ are defined as in Section 6.6.2. We slightly modify the decryption algorithm. We run the decryption algorithm of the underlying LinFE scheme to obtain $Z = g^{\mathsf{M} \cdot \langle \vec{x}, \vec{y} \rangle}$, but halt it before computing the discrete logarithm $\log_g Z$, which is impossible when $\mathsf{M} \cdot \langle \vec{x}, \vec{y} \rangle$ is exponentially large. Instead, we compute $Z^{1/\langle \vec{x}, \vec{y} \rangle} = g^{\mathsf{M}}$ and then retrieve the message M by solving the discrete logarithm problem.

The above scheme can encrypt only short messages. We can modify the scheme so that it can encrypt longer messages without degrading the efficiency much. The main idea is that we can use the above scheme as a key encapsulation mechanism (KEM). Namely, we change the above scheme so that the encryption algorithm first encrypts a randomness $s \in \mathbb{Z}_p$ and then encrypt the message M by using the "DEM key" $K = g^s$. The decryption algorithm first retrieves $K = g^s$ and then retrieves the message M using the key K.

Construction from the DCR Assumption. In [ALS16], the authors proposed two LinFE schemes from the DCR assumption. One is in the stateless setting where the inner product is taken over $\mathbb{Z}$, and the other is in the stateful setting where the inner product is taken over $\mathbb{Z}_N$. To apply the conversion to the former scheme, we set $\mathcal{D} = \mathbb{Z}$, $\mathcal{P} = \mathcal{Q} = \{0, \ldots, P-1\}^\ell$, $\mathcal{I} = \{0, \ldots, I-1\}^\ell$, $\mathcal{M} = \{0, \ldots, M-1\}$ and $\mathcal{J} = \{0, \ldots, MI-1\}$ for (possibly exponentially large) integers P, I, M. It is straightforward to see that these domains satisfy our condition for the conversion. This results in a stateless NIPE scheme over $\mathbb{Z}$. To apply the conversion to the latter scheme, we set $\mathcal{D} = \mathbb{Z}_N$, $\mathcal{P} = \mathcal{Q} = \mathcal{I} = \mathcal{J} = \mathbb{Z}_N^\ell$, and $\mathcal{M} = \mathbb{Z}_N$. Rigorously speaking, we cannot apply the conversion because $\mathbb{Z}_N$ is not an integral domain. However, we can treat $\mathbb{Z}_N$ as if it were an integral domain, since any element $x \in \mathbb{Z}_N$ with $\gcd(x, N) \neq 1$ will allow us to factorize N, which contradicts the hardness of the DCR assumption.

6.7 Formal Treatment on Multi-Dimensional Lattices

In this section, we provide some discussions on the main tool we use for this work — *multi-dimensional lattices*. We believe the new definitions and developed techniques to be of interest to applications elsewhere.

6.7.1 Background

A symmetric positive-definite matrix $\Sigma \in \mathbb{R}^{\ell \times \ell}$, expressed as $\Sigma > \mathbf{0}$ for short, is a matrix such that $\Sigma = \Sigma^\top$ and $\mathbf{x}^\top \Sigma \mathbf{x} > 0$ for all non-zero $\mathbf{x} \in \mathbb{R}^\ell$. Furthermore, for any $\Sigma > \mathbf{0}$, there exists a unique lower triangular matrix $\mathbf{U} \in \mathbb{R}^{\ell \times \ell}$ such that $\Sigma = \mathbf{U}\mathbf{U}^\top$. In the following, we denote this matrix $\mathbf{U}$ as $\sqrt{\Sigma}$. Positive definiteness defines a partial ordering on symmetric matrices: we say $\Sigma_1 > \Sigma_2$ if $(\Sigma_1 - \Sigma_2) > \mathbf{0}$. In the section, for any matrix $\mathbf{R}$, we use $s_{\max}(\mathbf{R})$ and $s_{\min}(\mathbf{R})$ to denote the largest and smallest singular value of $\mathbf{R}$, respectively. Note that in the main body, we used $s_1(\mathbf{R})$ to denote $s_{\max}(\mathbf{R})$.

6.7.2 Discrete Gaussian Measures over Multi Lattices

In this section, we define the discrete Gaussian distribution over an m-dimensional ℓ-multi lattice $\bar{\Lambda} \subseteq \mathbb{R}^{m \times \ell}$ where the Gaussian parameter is given by a symmetric positive-definite matrix $\Sigma \in \mathbb{R}^{\ell \times \ell}$. Here an m-dimensional ℓ-multi lattice is defined as a discrete additive subgroup of $\mathbb{R}^{m \times \ell}$. We emphasize that unlike in the main body of the paper, we do not require the multi lattice to be of the specific form $\Lambda^\ell = [\Lambda | \cdots | \Lambda] = \{[\mathbf{z}_1 | \cdots | \mathbf{z}_\ell] \mid \forall \mathbf{z}_i \in \Lambda, \forall i \in [\ell]\} \in \mathbb{Z}^{m \times \ell}$. From here on, we add a bar on top of multi lattice related notations, e.g., $\bar{\Lambda}, \bar{\eta}$, when we want to explicitly differentiate between a normal lattice notion and a multi lattice notion. Furthermore, the dual multi lattice $\bar{\Lambda}^*$ is defined as $\bar{\Lambda}^* = \{\mathbf{W} \in \mathbb{R}^{m \times \ell} \mid \mathbf{X}^\top \mathbf{W} \in \mathbb{Z}^{\ell \times \ell}, \forall \mathbf{X} \in \bar{\Lambda}\}$. Note that for the special case $\bar{\Lambda} = \Lambda^\ell$, we have $(\Lambda^\ell)^* = (\Lambda^*)^\ell$. Also, for any matrix $\mathbf{M} \in \mathbb{Z}^{\ell \times t}$, $\bar{\Lambda}\mathbf{M}$ denotes the m-dimensional t-multi lattice $\{\mathbf{Z}\mathbf{M} \mid \mathbf{Z} \in \bar{\Lambda}\} \in \mathbb{R}^{m \times t}$.

We first define the m-dimensional ℓ-multi Gaussian function $\bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X})$ over $\mathbb{R}^{m \times \ell}$ with a symmetric positive-definite matrix $\Sigma \in \mathbb{R}^{\ell \times \ell}$ as $\bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X}) = \exp(-\pi \cdot \mathrm{tr}(\mathbf{X}\Sigma^{-1}\mathbf{X}^\top))$. Similarly, the discrete Gaussian distribution for an m-dimensional ℓ-multi shifted lattice $\bar{\Lambda} + \mathbf{T}$ is defined as $D_{\bar{\Lambda}+\mathbf{T}, \sqrt{\Sigma}}(\mathbf{X}) = \bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X}) / \bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T})$ for all $\mathbf{X} \in \bar{\Lambda} + \mathbf{T}$ and $\mathbf{T} \in \mathbb{Z}^{m \times \ell}$, where $\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T}) = \sum_{\mathbf{X} \in \bar{\Lambda}+\mathbf{T}} \bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X})$. Note that when $\bar{\Lambda} = \Lambda^\ell$ for some lattice Λ and $\Sigma = \sigma^2 \mathbf{I}_\ell$, this corresponds to the special case we described in the main body, where each column of $\mathbf{X}$ are independent samples from Λ. This fact can be seen by observing that

$$\exp(\mathrm{tr}(\mathbf{X}\mathbf{X}^\top)) = \exp(\mathrm{tr}(\mathbf{X}^\top\mathbf{X})) = \exp(\sum_{i=1}^{\ell} \|\mathbf{x}_i\|^2) = \prod_{i=1}^{\ell} \exp(\|\mathbf{x}_i\|^2),$$

where $\mathbf{x}_i \in \mathbb{Z}^m$ is the i-th column of $\mathbf{X} \in \mathbb{Z}^{m \times \ell}$.

Vectorization of Matrices. To argue how well discrete Gaussian distributions over multi lattices behave, we need something similar to the smoothing parameter for (standard one-multi) lattices. We do this by observing that multi lattices can be viewed equivalently as a standard lattice via the isomorphism[4] $\phi : \mathbb{R}^{m\times\ell} \to \mathbb{R}^{m\ell}$ defined as follows:

$$\phi\left(\mathbf{X} = [\mathbf{x}_1|\mathbf{x}_2|\cdots|\mathbf{x}_\ell]\right) = \begin{bmatrix}\mathbf{x}_1\\ \mathbf{x}_2\\ \vdots\\ \mathbf{x}_\ell\end{bmatrix} \in \mathbb{R}^{m\ell}.$$

Since this is an isomorphism between vector spaces, it can be checked that a multi lattice $\bar{\Lambda}$ in $\mathbb{Z}^{m\times\ell}$ is isomorphic to a lattice $\phi(\bar{\Lambda})$ in $\mathbb{Z}^{m\ell}$. Above we defined ϕ for a particular pair of variables (m, ℓ), however with an abuse of notation, hereafter we define ϕ for any (m, ℓ), i.e., we view ϕ as simply an operation that stacks the columns of a given matrix on top of one another. Some standard formulas we use are as follows: for any $\mathbf{X} \in \mathbb{R}^{m\times\ell}, \mathbf{Y} \in \mathbb{R}^{\ell\times t}, \mathbf{Z} \in \mathbb{R}^{\ell\times\ell}$ we have

- $\phi(\mathbf{XY}) = (\mathbf{Y}^\top \otimes \mathbf{I}_m)\cdot\phi(\mathbf{X}) = (\mathbf{I}_t \otimes \mathbf{X})\cdot\phi(\mathbf{Y}) \in \mathbb{R}^{mt}$
- $\mathrm{tr}(\mathbf{XZX}^\top) = \phi(\mathbf{X})^\top(\mathbf{Z}\otimes\mathbf{I}_m)\phi(\mathbf{X})$

Using this, we can relate the m-dimensional ℓ-multi Gaussian function $\bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X})$ to an $m\ell$-dimensional Gaussian function $\rho_{\sqrt{\Sigma'}}(\mathbf{x})$. Concretely,

$$\begin{aligned}\bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X}) &= \exp(-\pi\cdot\mathrm{tr}(\mathbf{X}\Sigma^{-1}\mathbf{X}^\top))\\ &= \exp\big(-\pi\cdot\phi(\mathbf{X})^\top(\Sigma^{-1}\otimes\mathbf{I}_m)\phi(\mathbf{X})\big) && (6.25)\\ &= \exp\big(-\pi\cdot\phi(\mathbf{X})^\top(\Sigma\otimes\mathbf{I}_m)^{-1}\phi(\mathbf{X})\big) && (6.26)\\ &= \rho_{\sqrt{\Sigma\otimes\mathbf{I}_m}}(\phi(\mathbf{X})), && (6.27)\end{aligned}$$

where Eq.(6.25) follows from the second formula, Eq.(6.26) follows from $\mathbf{A}^{-1}\otimes\mathbf{B}^{-1} = (\mathbf{A}\otimes\mathbf{B})^{-1}$, and Eq.(6.27) follows from $\Sigma\otimes\mathbf{I}_m = (\sqrt{\Sigma}\otimes\mathbf{I}_m)(\sqrt{\Sigma}\otimes\mathbf{I}_m)^\top$. Therefore, $D_{\bar{\Lambda}+\mathbf{T},\sqrt{\Sigma}}(\mathbf{X}) = D_{\phi(\bar{\Lambda})+\phi(\mathbf{T}),\sqrt{\Sigma\otimes\mathbf{I}_m}}(\phi(\mathbf{X}))$ for any $\mathbf{X}\in\bar{\Lambda}+\mathbf{T}$.

Furthermore, using ϕ we can check that a multi lattice $\bar{\Lambda}\mathbf{M} \in \mathbb{R}^{m\times t}$ for $\mathbf{M}\in\mathbb{Z}^{\ell\times t}$ is isomorphic to the lattice $(\mathbf{M}^\top\otimes\mathbf{I}_m)\cdot\phi(\bar{\Lambda}) = \{(\mathbf{M}^\top\otimes\mathbf{I}_m)\mathbf{z} \mid \mathbf{z}\in\phi(\bar{\Lambda})\} \subseteq \mathbb{R}^{mt}$. Finally, for the special case $\bar{\Lambda} = \Lambda^\ell$, we have $(\phi(\Lambda^\ell))^* = \phi((\Lambda^*)^\ell)$.

Useful Lemmas for Multi Lattices. We will now study the behavior of a discrete Gaussian distribution over multi lattices by observing its lattice counterparts. To do so, we prepare one last tool; the smoothing parameter for lattices.

Definition 6.1. *For an m-dimensional lattice Λ and positive real $\epsilon > 0$, we define the smoothing parameter $\eta_\epsilon(\Lambda)$ as the smallest real $s > 0$ such that $\rho_{1/s}(\Lambda^*\backslash\{\mathbf{0}\}) \le \epsilon$. Furthermore, let $\Sigma > \mathbf{0}$ be any symmetric positive-definite matrix in $\mathbb{R}^{m\times m}$. We say $\sqrt{\Sigma} \ge \eta_\epsilon(\Lambda)$ if $\rho_{\sqrt{\Sigma^{-1}}}(\Lambda^*\backslash\{\mathbf{0}\}) \le \epsilon$.*

It is informative to observe that, if $\sqrt{\Sigma} \ge \eta_\epsilon(\Lambda)$, then $\sqrt{\lambda_{\mathsf{max}}} \ge \eta_\epsilon(\Lambda)$ where λ_{max} denotes the largest eigenvalue of Σ. Equivalently, since for symmetric positive-definite matrices eigenvalues equal their singular values, $\sqrt{s_{\mathsf{max}}} \ge \eta_\epsilon(\Lambda)$ where s_{max} denotes the largest singular value. On the other hand, if $\sqrt{\lambda_{\mathsf{min}}} = \sqrt{s_{\mathsf{min}}} \ge \eta_\epsilon(\Lambda)$, then $\sqrt{\Sigma} \ge \eta_\epsilon(\Lambda)$ where λ_{min} and s_{min} denotes the smallest eigenvalue and singular value of Σ, respectively.

We define the smoothing parameter for a multi lattice as below.

[4] In linear algebra, this isomorphism is sometimes called the *vectorization* of matrices.

Definition 6.2. *For an m-dimensional ℓ-multi lattice $\bar{\Lambda}$ and positive real $\epsilon > 0$, we define the (multi lattice) smoothing parameter $\bar{\eta}_\epsilon(\bar{\Lambda})$ as $\eta_\epsilon(\phi(\bar{\Lambda}))$. I.e., the smallest real $s > 0$ such that $\bar{\rho}_{1/s}(\bar{\Lambda}^* \backslash \{\mathbf{0}_{m\times\ell}\}) = \rho_{1/s}(\phi(\bar{\Lambda}^*)\backslash\{\mathbf{0}\}) \leq \epsilon$. Furthermore, let $\Sigma > \mathbf{0}$ be any symmetric positive-definite matrix in $\mathbb{R}^{\ell\times\ell}$. We say $\sqrt{\Sigma} \geq \bar{\eta}_\epsilon(\bar{\Lambda})$ if and only if $\sqrt{\Sigma} \otimes \mathbf{I}_m \geq \eta_\epsilon(\phi(\bar{\Lambda}))$.*

Observe that if $\bar{\Lambda}_0 \subseteq \bar{\Lambda}_1$, then $\bar{\eta}_\epsilon(\bar{\Lambda}_0) \geq \bar{\eta}_\epsilon(\bar{\Lambda}_1)$ for any ϵ, since $\bar{\Lambda}_0^* \supseteq \bar{\Lambda}_1^* \Leftrightarrow \phi(\bar{\Lambda}_0^*) \supseteq \phi(\bar{\Lambda}_1^*)$. Furthermore, the same argument we did above, e.g., if $\sqrt{s_{\mathsf{min}}} \geq \bar{\eta}_\epsilon(\bar{\Lambda})$, then $\sqrt{\Sigma} \geq \bar{\eta}_\epsilon(\bar{\Lambda})$, holds, since the smallest (resp. largest) singular value of the symmetric positive definite matrix $\Sigma \otimes \mathbf{I}_m$ is the same as Σ. Now that we have formally defined the smoothing parameter for multi lattices, we obtain a standard result analogous to that of one-multi lattices.

Lemma 6.7. *[Corollary of [MR04], Lemma 4.4] Let $\bar{\Lambda}$ be any m-dimensional ℓ-multi lattice. For any $\epsilon \in (0,1)$, symmetric positive-definite matrix Σ in $\mathbb{R}^{\ell\times\ell}$ such that $\sqrt{\Sigma} \geq \bar{\eta}_\epsilon(\bar{\Lambda})$, and any $\mathbf{T} \in \mathbb{R}^{m\times\ell}$, we have*

$$\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T}) \in [\frac{1-\epsilon}{1+\epsilon}, 1] \cdot \bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda}).$$

Proof. This follows from the standard results of [MR04, Lemma 4.4] and [MP12, Lemma 2.4]. Namely, it follows directly from the definition $\sqrt{\Sigma} \geq \bar{\eta}_\epsilon(\bar{\Lambda}) \Leftrightarrow \sqrt{\Sigma} \otimes \mathbf{I}_m \geq \eta_\epsilon(\phi(\bar{\Lambda}))$ and that $\sqrt{\Sigma} \otimes \mathbf{I}_m$ is non-singular. □

Finally, for the special case $\bar{\Lambda} = \Lambda$, we can bound the smoothing parameter of $\bar{\Lambda}$ using Λ.

Lemma 6.8. *For any m-dimensional lattice Λ and $\epsilon \in (0, 1/2)$, we have $\bar{\eta}_\epsilon(\Lambda^\ell) \leq \eta_{\epsilon'}(\Lambda)$, where $\epsilon' = (1+\epsilon)^{1/\ell} - 1$. In particular, for any $\epsilon = \mathsf{negl}(\lambda)$ and $\ell = \mathsf{poly}(\lambda)$, we have $\epsilon' = \mathsf{negl}(\lambda)$.*

Proof. Observe that $\phi(\Lambda^\ell) = \{[\mathbf{x}_1^\top | \cdots | \mathbf{x}_\ell^\top]^\top \in \mathbb{R}^{m\ell} \mid \mathbf{x}_i \in \Lambda, \forall i \in [\ell]\}$ and $(\Lambda^\ell)^* = (\Lambda^*)^\ell$. Then, by definition $\bar{\rho}_{1/s}((\Lambda^\ell)^*) = \rho_{1/s}(\Lambda^*)^\ell$. Furthermore, for any positive real $s \geq \eta_{\epsilon'}(\Lambda)$, we have $\rho_{1/s}(\Lambda^* \backslash \{\mathbf{0}\}) \leq \epsilon'$. Equivalently, $\rho_{1/s}(\Lambda^*) \leq 1 + \epsilon'$, since $\rho_{1/s}(\mathbf{0}) = 1$. Therefore,

$$\bar{\rho}_{1/s}((\Lambda^\ell)^* \backslash \{\mathbf{0}_{m\times\ell}\}) = \bar{\rho}_{1/s}((\Lambda^\ell)^*) - 1 \leq (1+\epsilon')^\ell - 1 = \epsilon.$$

Hence, $\eta_{\epsilon'}(\Lambda) \geq \bar{\eta}_\epsilon(\Lambda^\ell)$. □

6.7.3 Sum of Discrete Gaussians

The following theorem is a generalization of [BF11], Theorem B.1.], and can be used as an alternative tool to [BF11], Theorem B.3.]. The main advantage of our theorem is that, in the special case when the multi lattice is of the form Λ^ℓ, our theorem may allow for a much tighter (exponentially tighter) bound on the Gaussian parameter.

Theorem 6.6 (Generalization of [BF11], Theorem B.1.)**.** *Let m, ℓ, t be positive integers such that $t < \ell$. Let $\bar{\Lambda} \subseteq \mathbb{Z}^{m\times\ell}$ be a multi lattice, $\mathbf{M} \in \mathbb{Z}^{\ell\times(\ell-t)}$ be a full rank matrix, $\mathbf{T} \in \mathbb{Z}^{m\times\ell}$ be a matrix and $\Sigma \in \mathbb{R}^{\ell\times\ell}$ be a symmetric positive-definite matrix. Let $\mathbf{W} \in \mathbb{Z}^{\ell\times t}$ be a full rank matrix that satisfies $\mathbf{W}^\top\mathbf{M} = \mathbf{0} \in \mathbb{Z}^{t\times(\ell-t)}$ and let L be the m-dimensional t-multi lattice*

$$L := \{\mathbf{U} \in \mathbb{R}^{m\times t} \mid \mathbf{U}\mathbf{W}^\top \in \bar{\Lambda}\}.$$

Furthermore, suppose that $\sqrt{(\mathbf{W}^\top\Sigma^{-1}\mathbf{W})^{-1}} > \bar{\eta}_\epsilon(L)$ for some negligible ϵ.
If $\mathbf{X}$ is distributed as $D_{\bar{\Lambda}+\mathbf{T},\sqrt{\Sigma}}$, then $\mathbf{X}\mathbf{M}$ is statistically close to $D_{\bar{\Lambda}\mathbf{M}+\mathbf{T}\mathbf{M},\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}$.

Proof. The proof follows the outline of the proof of [BF11], Theorem B.1, with additional techniques to work with multi lattices.

Below, we aim at computing the probability of $\Pr[\mathbf{XM} = \mathbf{V}]$ for $\mathbf{V} \in \bar{\Lambda}\mathbf{M} + \mathbf{TM}$ when $\mathbf{X}$ is distributed as $D_{\bar{\Lambda}+\mathbf{T},\sqrt{\Sigma}}$. First, define the set $S_{\mathbf{V}} = \{\mathbf{Z} \in \bar{\Lambda} + \mathbf{T} \mid \mathbf{ZM} = \mathbf{V}\}$. Let $\mathbf{X}_0 \in \bar{\Lambda} + \mathbf{T}$ be an arbitrary solution ot $\mathbf{ZM} = \mathbf{V}$. Then, since the kernel of the linear map $\mathbf{M} \in \mathbb{Z}^{\ell\times(\ell-t)}$ is spanned by the columns of $\mathbf{W} \in \mathbb{Z}^{\ell\times t}$, we have

$$S_{\mathbf{V}} = \mathbf{X}_0 + \{\mathbf{Z} \in \bar{\Lambda} \mid \mathbf{ZM} = \mathbf{0}_{m\times(\ell-t)}\} = \mathbf{X}_0 + \{\mathbf{UW}^\top \in \bar{\Lambda} \mid \mathbf{U} \in L\},$$

where L is a multi lattice as defined in the theorem. Now,

$$\begin{aligned}\Pr[\mathbf{XM} = \mathbf{V}] = \Pr[\mathbf{X} \in S_{\mathbf{V}}] &= \sum_{\mathbf{U}\in L} \Pr[\mathbf{X} = \mathbf{X}_0 + \mathbf{UW}^\top] \\ &= \frac{1}{\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T})} \sum_{\mathbf{U}\in L} \exp\left(-\pi \cdot \operatorname{tr}\left((\mathbf{X}_0 + \mathbf{UW}^\top)\Sigma^{-1}(\mathbf{X}_0 + \mathbf{UW}^\top)^\top\right)\right) \qquad (6.28)\end{aligned}$$

To properly decouple the terms in $\operatorname{tr}(\cdot)$, we use the following fact on linear algebra.

Fact 6.1. *Let $\mathbf{M} \in \mathbb{R}^{\ell\times(\ell-t)}$, $\mathbf{W} \in \mathbb{R}^{\ell\times t}$ be full-rank matrices such that $\mathbf{W}^\top\mathbf{M} = \mathbf{0}_{t\times(\ell-t)}$, and $\Sigma \in \mathbb{R}^{\ell\times\ell}$ be a symmetric positive-definite matrix. Then, we have the following:*

$$\mathbf{M}\left(\mathbf{M}^\top\Sigma\mathbf{M}\right)^{-1}\mathbf{M}^\top = \Sigma^{-1} - \Sigma^{-1}\mathbf{W}\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)^{-1}\mathbf{W}^\top(\Sigma^{-1})^\top.$$

Proof. Denote the matrix on the right (resp. left) hand side as $\mathbf{A} \in \mathbb{R}^{\ell\times\ell}$ (resp. $\mathbf{B} \in \mathbb{R}^{\ell\times\ell}$). Then, using the fact that $\mathbf{W}^\top\mathbf{M} = \mathbf{0}_{t\times(\ell-t)}$ and $\Sigma = \Sigma^\top$, direct calculation shows that

$$\left[\Sigma^\top\mathbf{M} \mid \mathbf{W}\right]^\top\mathbf{A} = \left[\mathbf{M} \mid \mathbf{0}_{\ell\times t}\right]^\top = \left[\Sigma^\top\mathbf{M} \mid \mathbf{W}\right]^\top\mathbf{B}.$$

Since Σ and $[\mathbf{M}|\mathbf{W}] \in \mathbb{R}^{\ell\times\ell}$ are non-singular, we have that $\mathbf{A} = \mathbf{B}$. □

Then, the term $\operatorname{tr}(\cdot)$ in Eq.(6.28) can be expressed as follows:

$$\begin{aligned}&\operatorname{tr}\left((\mathbf{X}_0 + \mathbf{UW}^\top)\Sigma^{-1}(\mathbf{X}_0 + \mathbf{UW}^\top)^\top\right) \\ =&\operatorname{tr}\left(\mathbf{UW}^\top\Sigma^{-1}\mathbf{WU}^\top\right) + 2\cdot\operatorname{tr}\left(\mathbf{X}_0\Sigma^{-1}\mathbf{WU}^\top\right) + \operatorname{tr}\left(\mathbf{X}_0\Sigma^{-1}\mathbf{X}_0^\top\right) \\ =&\operatorname{tr}\left((\mathbf{U}+\mathbf{Y})\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)(\mathbf{U}+\mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{X}_0\Sigma^{-1}\mathbf{X}_0^\top\right) - \operatorname{tr}\left(\mathbf{Y}\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)\mathbf{Y}^\top\right) \qquad (6.29) \\ =&\operatorname{tr}\left((\mathbf{U}+\mathbf{Y})\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)(\mathbf{U}+\mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{X}_0\mathbf{M}\left(\mathbf{M}^\top\Sigma\mathbf{M}\right)^{-1}\mathbf{M}^\top\mathbf{X}_0^\top\right) \qquad (6.30) \\ =&\operatorname{tr}\left((\mathbf{U}+\mathbf{Y})\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)(\mathbf{U}+\mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{V}\left(\mathbf{M}^\top\Sigma\mathbf{M}\right)^{-1}\mathbf{V}^\top\right) \qquad (6.31)\end{aligned}$$

where in Eq.(6.29) we substitute $\mathbf{Y} = \mathbf{X}_0\Sigma^{-1}\mathbf{W}\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)^{-1}$, in Eq.(6.30) we use Fact 6.1, and in Eq.(6.31) we use the equality $\mathbf{V} = \mathbf{X}_0\mathbf{M}$. Then plugging Eq.(6.31) back in Eq.(6.28), we obtain

$$\begin{aligned}\Pr[\mathbf{XM} = \mathbf{V}] &= \frac{\bar{\rho}_{\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}(\mathbf{V})}{\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T})} \sum_{\mathbf{U}\in L} \exp\left(-\pi\cdot\operatorname{tr}\left((\mathbf{U}+\mathbf{Y})\left(\mathbf{W}^\top\Sigma^{-1}\mathbf{W}\right)(\mathbf{U}+\mathbf{Y})^\top\right)\right) \\ &= \frac{\bar{\rho}_{\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}(\mathbf{V})}{\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T})} \cdot \bar{\rho}_{\sqrt{(\mathbf{W}^\top\Sigma^{-1}\mathbf{W})^{-1}}}(L + \mathbf{Y}).\end{aligned}$$

Since $\sqrt{(\mathbf{W}^\top\Sigma^{-1}\mathbf{W})^{-1}} > \bar{\eta}_\epsilon(L)$ and from Lemma 6.7, for all $\mathbf{Y} \in \mathbb{R}^{m\times\ell}$ we have

$$\Pr[\mathbf{XM} = \mathbf{V}] \in \left[\frac{1-\epsilon}{1+\epsilon},\ 1\right] \cdot \frac{\bar{\rho}_{\sqrt{(\mathbf{W}^\top\Sigma^{-1}\mathbf{W})^{-1}}}(L)}{\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda}+\mathbf{T})} \cdot \bar{\rho}_{\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}(\mathbf{V})$$

Since ϵ is negligible and $\bar{\rho}_{\sqrt{(\mathbf{W}^\top\Sigma^{-1}\mathbf{W})^{-1}}}(L)/\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda}+\mathbf{T})$ is a constant independent of $\mathbf{V}$, it follows that $\Pr[\mathbf{XM} = \mathbf{V}] \in [\frac{1-\epsilon}{1+\epsilon},\ 1] \cdot \bar{\rho}_{\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}(\mathbf{V})/\bar{\rho}_{\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}(\bar{\Lambda}\mathbf{M}+\mathbf{TM})$. Hence, by definition, $\mathbf{XM}$ is statistically close to $D_{\bar{\Lambda}\mathbf{M}+\mathbf{MT},\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}$. □

Corollary 6.1. *Let q be a prime or some power of a prime p. Let n, m, ℓ, t be positive integers such that $m \geq 2n\log q$ and $\ell > t$, let $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ be a random matrix and $\mathbf{T} \in \mathbb{Z}^{m\times\ell}$ be an arbitrary matrix. Let $\mathbf{M} \in \mathbb{Z}^{\ell\times(\ell-t)}$ be a full rank matrix and let $\mathbf{W} \in \mathbb{Z}^{\ell\times t}$ satisfy $\mathbf{W}^\top\mathbf{M} = \mathbf{0} \in \mathbb{Z}^{t\times(\ell-t)}$. Finally, let σ be a positive real such that $\sigma > \sqrt{s_{\max}(\mathbf{W}^\top\mathbf{W})} \cdot \omega(\sqrt{\log m})$.*

If, $\mathbf{X} \in \mathbb{Z}^{m\times\ell}$ is distributed as $D_{\Lambda^\perp(\mathbf{A})^\ell+\mathbf{T},\sigma\mathbf{I}_\ell}$, then $\mathbf{XM} \in \mathbb{Z}^{m\times(\ell-t)}$ is statistically close to $D_{\Lambda^\perp(\mathbf{A})^\ell\mathbf{M}+\mathbf{TM},\sigma\sqrt{\mathbf{M}^\top\mathbf{M}}}$.

Proof. Plugging in $\Sigma = \sigma^2\mathbf{I}_\ell$, to use Theorem 6.6 it suffices to show that $\sigma \cdot \sqrt{(\mathbf{W}^\top\mathbf{W})^{-1}} > \bar{\eta}_\epsilon(L)$ for some negligible ϵ, where L is the m-dimensional t-multi lattice defined as

$$L := \{\mathbf{U} \in \mathbb{R}^{m\times t} \mid \mathbf{U}\mathbf{W}^\top \in \Lambda^\perp(\mathbf{A})^\ell\}.$$

First, notice that, since $\Lambda^\perp(\mathbf{A})$ is closed under addition, we have $\Lambda^\perp(\mathbf{A})^t \subseteq L$. This implies that $\bar{\eta}_\epsilon(L) \leq \bar{\eta}_\epsilon(\Lambda^\perp(\mathbf{A})^t)$. Next, by Lemma 6.8 we have $\bar{\eta}_\epsilon(\Lambda^\perp(\mathbf{A})^t) \leq \eta_{\epsilon'}(\Lambda^\perp(\mathbf{A}))$, where $\epsilon' = (1+\epsilon)^{1/t} - 1$ is negligible, since $t = \mathsf{poly}(\lambda)$. Furthermore, for a random choice of $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, we have $\eta_{\epsilon'}(\Lambda^\perp(\mathbf{A})) < \omega(\sqrt{\log m})$ with all but negligible probability [GPV08]. Therefore, since $(s_{\max}(\mathbf{W}^\top\mathbf{W}))^{-1} = s_{\min}((\mathbf{W}^\top\mathbf{W})^{-1})$, if $\sigma > \sqrt{s_{\max}(\mathbf{W}^\top\mathbf{W})} \cdot \omega(\sqrt{\log m})$, then $\sigma \cdot \sqrt{s_{\min}((\mathbf{W}^\top\mathbf{W})^{-1})} > \bar{\eta}_\epsilon(L)$. Here, $s_{\max}(\mathbf{Z})$ (resp. $s_{\min}(\mathbf{Z})$) denotes the largest (resp. smallest) singular value of $\mathbf{Z}$. Finally, by definition this implies $\sigma \cdot \sqrt{(\mathbf{W}^\top\mathbf{W})^{-1}} > \bar{\eta}_\epsilon(L)$ for some negligible ϵ, which completes the proof. □

Chapter 7

Attribute-based Signatures for Unbounded Circuits in the ROM

7.1 Introduction

Attribute-based signature (ABS) was introduced by [MPR11] as a versatile tool allowing a signer to *anonymously* authenticate a message M w.r.t. a public signing policy C only if the signer has a signing key associated to an attribute $x \in \{0,1\}^*$ that satisfies C, i.e., $C(x) = 1$. An attribute-based signature scheme reveals no information on the signer's identity or the attribute other than the fact that the signature is valid, hence the anonymity property of ABS schemes. One of the central research themes on ABS schemes is to expand the expressiveness of the class of policies that can be supported by the schemes. In the bilinear map setting, there has been a long line of interesting works, including ABS schemes for threshold policy (e.g., [HLLR12]), boolean formula (e.g., [MPR11, OT11, OT13, EGK14]) and the current state-of-the-art; *unbounded circuits* [SAH16].[1]

On the other hand, the constructions of ABS schemes without bilinear maps, in particular ABS schemes from lattices, are much less investigated. To the best of our knowledge, there are only two major works concerning lattice-based ABS schemes [EE16, Tsa17]. Rachid et al. [EE16] construct a lattice-based ABS scheme for boolean formulas using a non-interactive zero-knowledge (NIZK) proof system as the main building block, following one of the most promising ways of constructing ABS schemes [MPR11, EGK14, SAH16]. Informally, a signature for a signer with attribute $\mathbf{x}$ is simply a zero-knowledge proof attesting to the fact that he has a certificate corresponding to the attribute $\mathbf{x}$ issued by the authority and that the policy C associated to the message M satisfies $C(\mathbf{x}) = 1$. Although this approach has been very effective in the bilinear map setting where [SAH16] were able to obtain ABS schemes for *unbounded circuits*, this has not been the case for lattices. One of the main reasons behind this is the lack of efficient lattice-based NIZK proof systems for a wide enough language. In particular, we only have efficient NIZK proof systems tailored for specific languages, such as proving possession of a solution to the short integer solution (SIS) problem or the learning with errors (LWE) problem [LNSW13], proving possession of a valid signature of the Boyen digital signature scheme [Boy10, LLNW14, LNW15] and so on,

[0]The contents of this chapter is based on the work presented at PKC 2018 under the title "Attribute-based Signatures for Unbounded Circuits in the ROM and Efficient Instantiations from Lattices" [EK18].

[1] In this work, we only consider *message-policy* ABS schemes. Recall that using universal circuits, we can convert message-policy ABS schemes into *key-policy* ABS schemes [BF14], where the functionality of the secret keys and messages are reversed.

which in general does not seem strong enough for constructing ABS schemes. Recently, [YAL$^+$17] showed (informally) how to construct lattice-based NIZK proof systems for languages accepted by monotone span programs, however, this still does not seem strong enough to use as a building block for ABS schemes supporting *unbounded* circuits as policies.

Tsabary [Tsa17] constructs lattice-based ABS schemes following a different approach; they show equivalence between a homomorphic signature (HS) scheme and a (message-policy) ABS scheme. Therefore, based on the HS construction of Gorbunov et al. [GVW15b], they achieve a lattice-based ABS scheme for *bounded circuits* that does not make use of NIZK proof systems.[2] Here, by bounded, we mean that the required hardness assumptions on the LWE and/or SIS problems grow exponentially in the depth of the circuit, e.g., to base the security of the ABS scheme under a polynomial LWE assumption, we need to restrict the depth of the circuit to be $O(\log \lambda)$, where λ is the security parameter. However, it seems challenging to improve their techniques to ABS schemes for unbounded circuits, due to the inherent noise-growth incurred by the homomorphic operations of matrices while computing the circuit gate-by-gate. The only known method of overcoming these $O(\log \lambda)$ depth barrier concerning homomorphic operations is the bootstrapping technique of fully homomorphic encryptions [Gen09], however, it is still an open problem whether there is a signature analogue of this technique.

7.1.1 Our Contribution

In this chapter, we affirmatively close the gap between the state-of-the-art ABS schemes based on bilinear maps and lattices by constructing the first lattice-based ABS scheme for *unbounded circuits* in the random oracle model. We start by providing a general construction of ABS schemes supporting unbounded-circuits as policies. We then give an instantiation in the lattice setting showing that all the building blocks required by our generic construction is obtainable from lattices. We stress that, despite the expressiveness of the signing policy, we manage to prove the security of our scheme under surprisingly mild SIS and LWE assumptions with polynomial modulus size $q = \tilde{O}(\ell\lambda^{1.5})$, where ℓ denotes the length of the inputs to the circuits. Specifically, the required hardness assumptions are independent of the depth of the circuits that express the policies. Furthermore, the sizes of the public parameter, signing keys and signatures are $\tilde{O}(\ell\lambda^2)$, $\tilde{O}(\lambda)$ and $\tilde{O}((\ell\lambda + |C|)\lambda^2)$, respectively, where $|C|$ is the size of the circuit (i.e., policy) associated to the message.

To this end we prepare two new tools equipped for the lattice setting: we provide a generalization of the forking lemma of [PS00] which we call the *general multi-forking lemma with oracle access* and further construct a new lattice-based NIZK proof system for proving possession of a valid Boyen signature [Boy10] that departs from the previously known techniques (e.g., [LLNW14, LNW15]). Below, we give a more detailed overview of the techniques we used in our work.

[2] We note that the ABS scheme presented in [Tsa17] does not fulfill the standard security requirements of (message-policy) ABS schemes as originally defined in [MPR11]; achieving either unforgeability or anonymity in its full capacity comes at the cost of getting a much weaker version of the other, i.e., one has to choose between single-key-selective-unforgeability or leaking information about the signing key.

7.2 Technical Overview

Generic Construction of ABS for Unbounded Circuits. We propose a generic construction of ABS schemes supporting unbounded depth circuits as policies in the random oracle model[3], which employs the following primitives as its building blocks; a commitment scheme, a digital signature scheme and a Σ-protocol for a sufficiently wide relation. As a separate theoretical contribution, since all of the above primitives are implied from one-way functions, our result implies that one-way functions are sufficient to construct an ABS scheme for unbounded circuits in the random oracle model. Here, the random oracle is used only to convert the underlying Σ-protocol into a NIZK proof system via the Fiat-Shamir transformation [FS86].

At a high level, the generic construction of our ABS scheme follows closely the bilinear map based construction of [SAH16] (which is non-generic and proven in the standard model). We briefly review the construction in slightly more detail; first, the attribute authority issues a signature σ on an attribute $\mathbf{x} \in \{0,1\}^\ell$ to certify that a signer is indeed authorized to sign a message on behalf of that attribute. Then, to sign anonymously, the signer produces a zero-knowledge proof attesting to the following two facts:

(I) the signature σ issued by the authority is valid, and

(II) the corresponding secret attribute $\mathbf{x}$ satisfies the circuit C associated to the message M.

However, in spite of the similarities shared with the construction of [SAH16], the security proof of our construction requires a rather sensitive and technical analysis, which calls for new tools. This difficulty mainly stems from the fact that security proofs relying on the Fiat-Shamir-based NIZK proof systems are often times not as simple as the construction appears to be and in some cases the intuition may fail, e.g., [BPW12, BFW16].

Our proof of security of the generic ABS scheme relies on our generalization of the forking lemma of [PS00], which we call the *general multi-forking lemma with oracle access*. Our forking lemma can be seen as a generalization and a simplification of the general forking lemma of [BN06] and the improved forking lemma of [BPVY00]. In particular, we analyze the output behavior of an algorithm when run multiple times on related inputs, instead of when only run twice as in [BN06], while also providing it with oracle access to a deterministic algorithm. Recall that the original forking lemma of [PS00] applies to Fiat-Shamir type signature schemes and roughly states that, if there exists a valid forger $\mathcal{A}$, then one can rewind $\mathcal{A}$ initialized with the same randomness tape to find two accepting transcripts with the same commitment but different challenges, leading, via the special soundness property of Σ-protocols, to extract the secret signing key from the transcripts and hence a proof of security of the signature scheme in the random oracle model.

First, we require the forking lemma to analyze the output behavior of an algorithm on multiple runs to capture the situation arising in the recent lattice-based NIZK proof systems (e.g., [LNSW13, LLNW14, LNW15]) where the extractor of the underlying Σ-protocol requires more than two valid transcripts to extract a witness. Although the improved forking lemma of [BPVY00] captures this multiplicity of the forking lemma of a particular El Gamal-type signature scheme, it seems hard to apply in situations like ours where we are not dealing with regular signature schemes. Our forking lemma, similar to the one of [BN06], divorces the probabilistic essence of the forking lemma from any particular application context. Furthermore, our forking lemma provides worst-case rather than expected-time guarantees; the improved forking lemma of [BPVY00] roughly states that an expected $O(1/\epsilon)$ repeated executions of a forger $\mathcal{A}$ with advantage ϵ is required to extract a valid witness. We believe this feature to be more suitable for standard assumptions that are defined for PPT algorithms, as also stated in [BN06].

[3] In this work, we only consider circuits that do not have random oracle gates.

Second, and more importantly, our forking lemma allows the algorithm $\mathcal{A}$ that can be rewinded, to have oracle access to some algorithm $\mathcal{O}$ that *cannot* be rewinded. This is a useful feature for the forking algorithm to have in situations where the simulator cannot rewind all the algorithms which he is interacting with. This may be easiest to explain with a concrete example; in particular, when we reduce the eu-cma security of the underlying digital signature scheme to the security of our ABS scheme, the simulator (which is the eu-cma adversary) simulates the view of an ABS security game to the ABS adversary $\mathcal{A}$, and answers the queries made by $\mathcal{A}$ using its eu-cma challenger $\mathcal{O}$. At some point when $\mathcal{A}$ outputs a forgery for the ABS security game, the simulator hopes to extract the witness from the forgery and use it to win his own eu-cma security game. However, for this particular situation, the problem with all the previous forking lemmas is that the simulator will not be able to run the forking algorithm in the specified way; the simulator *can* rewind $\mathcal{A}$ to a particular point where the fork happens, however, the simulator *cannot* rewind the eu-cma challenger $\mathcal{O}$ in the same way, since it is outside the simulator's (i.e., eu-cma adversary's) control. Then, since the behavior of $\mathcal{A}$ is implicitly dependent on the behavior of the eu-cma challenger, the standard forking lemma does not provide meaningful analysis of the output of $\mathcal{A}$ on multiple runs. We therefore present a general multi-forking lemma *with oracle access* to capture these situations where the simulator is restricted to rewinding only some of the algorithms he is interacting with. We note that in case one is willing to use some algebraic problem such as the SIS or LWE problem as the underlying hardness assumption, these situations do not show up, since once given a fixed problem instance, the simulator can reuse it in every run to simulate the view to $\mathcal{A}$.

Finally, one of the benefits of using the Fiat-Shamir-based NIZK proof system is that we do not have to rely on the dummy attribute technique of those ABS schemes based on Goth-Sahai NIZK proof systems [MPR11, SAH16] to prove adaptive unforgeability, thus obtaining a more efficient signing algorithm. At a high level, this is because Fiat-Shamir based NIZK proof systems can be simulation-sound and extractable at the same time, whereas Goth-Sahai NIZK proof systems can only be instantiated to have one of the two properties. Therefore, during the proof of adaptive unforgeability, since the simulator needs to set up the common reference string in the extractable mode to extract a witness from the forgery, the simulator has to rely on these extra dummy attributes, which are never used in the actual scheme, to simulate signatures (i.e., proofs).

Instantiation from Lattices. To instantiate our generic ABS construction from lattices, we require three primitives: a signature scheme, a commitment scheme, and a Σ-protocol for a relation capturing the aforementioned items (I) and (II). As for the signature scheme, we can use the simple and efficient lattice-based signature scheme of Boyen [Boy10], which has been extensively studied in the lattice-based NIZK literatures. In particular, Ling et al. [LNSW13] provides an efficient Σ-protocol for proving possession of a valid Boyen signature (i.e., item (I)). However, unfortunately, it is not known whether the Σ-protocol of Ling et al. can be extended to prove circuit satisfiability, which is what we require in item (II), and in fact, recent subsequent results of [LLM$^+$16, YAL$^+$17] suggest that they are not powerful enough to capture circuit satisfiability. On the other hand, Xie et al. [XXW13] provides a lattice-based Σ-protocol for proving NP relations via arithmetic circuit satisfiability, which is what we exactly require in item (II), however, it does not seem possible to simply combine the two different types of Σ-protocols of [LNSW13] and [XXW13].

To this end, in this chapter we present a new Σ-protocol for proving possession of a valid Boyen signature by expressing the verification algorithm of the Boyen signature as a simple arithmetic circuit that is compatible with the Σ-protocol of Xie et al. Specifically, since both items (I) and (II) is now represented as simple arithmetic circuits, we can effectively use the Σ-protocol of Xie et al. to obtain our desired Σ-protocol. The main observation is that, most operations that show up in lattice-based cryptography are composed of simple arithmetic operations such as matrix multiplications, and therefore naturally leads to simple arithmetic circuit representations. For our particular case, the verification algorithm of the Boyen signature scheme essentially boils down to checking two simple conditions; whether a vector $\mathbf{z}$ satisfies $\|\mathbf{z}\|_\infty \leq \beta$ and $\mathbf{A}\mathbf{z} = \mathbf{u} \mod q$ for public matrix $\mathbf{A}$ and vector $\mathbf{u}$. Here, we intentionally dismiss the message for simplicity. As it can be seen, the latter equation is readily expressed by a very simple arithmetic circuit. On the other hand, the first inequality requires some extra work, however, this too can be expressed as an simple arithmetic circuit without much overhead by efficiently encoding predicates such as $x \overset{?}{\in} \{-1, 0, 1\}$ into arithmetic circuits.

7.3 Preparation

7.3.1 Commitment Schemes with Gap Openings

We define a standard commitment scheme that supports an additional notion we call *gap openings*. This additional notion will make it conceptually easier when we combine it with gap-Σ-protocols, which we later define. Informally, a commitment scheme with a gap opening is a standard commitment scheme where there may exist additional valid openings that are never created during the commitment algorithm. For those readers who are only interested in the generic attribute-based signature constructions from standard (i.e., non-lattice-based) Σ-protocols, they can safely skip the "gap" arguments.

Definition 7.1 (Commitments)**.** *A commitment scheme with message space $\mathcal{M}$ and commitment space $\mathcal{C}$ is a triple of PPT algorithms* $(\mathsf{C.Gen}, \mathsf{C.Com}, \mathsf{C.Open})$ *of the following form:*

$\mathsf{C.Gen}(1^\lambda) \to \mathsf{pk}$: *The key generation algorithm takes as input the security parameter 1^λ and outputs a public commitment key* pk*.*

$\mathsf{C.Com}(\mathsf{pk}, \mathsf{M}) \to (c, d)$: *The commitment algorithm takes as inputs the commitment key* pk *and message $\mathsf{M} \in \mathcal{M}$, and outputs a commitment/opening pair (c, d). We denote $\mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})$ as the set of all possible outputs of this algorithm under fixed* pk *and* M*.*

$\mathsf{C.Open}(\mathsf{pk}, \mathsf{M}, c, d) \to 1\backslash 0$: *The* deterministic *opening algorithm takes as inputs the commitment key* pk*, message* M *and commitment/opening pair (c, d) as inputs and outputs 1 or 0. We denote $\mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}, \mathsf{M})$ as the set of all possible pairs (c, d) this algorithm outputs 1 under fixed* pk *and* M*.*

Here, we require that checking membership of an element in $\mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})$ is efficient. We also require the commitment scheme to satisfy the following correctness notion: for all $\mathsf{M} \in \mathcal{M}, \mathsf{pk} \leftarrow \mathsf{C.Gen}(1^\lambda), (c, d) \leftarrow \mathsf{C.Com}(\mathsf{pk}, \mathsf{M})$ we have $\mathsf{C.Open}(\mathsf{pk}, \mathsf{M}, c, d) = 1$.

It is clear that we have $\mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M}) \subseteq \mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}, \mathsf{M})$ for all pk and $\mathsf{M} \in \mathcal{M}$. We say the commitment scheme has a *gap-opening* when $\mathcal{D}_{\mathsf{Com}} \subset \mathcal{D}_{\mathsf{G\text{-}Com}}$, i.e., there are valid openings that are never created by the commitment algorithm $\mathsf{C.Com}$. We require the following security notions for a commitment scheme:

Binding. We call the scheme unconditionally (resp. computationally) binding if for all (resp. PPT) algorithm $\mathcal{A}$, we have the following:

$$\Pr[\mathsf{pk} \leftarrow \mathsf{C.Gen}(1^\lambda); (c, \mathsf{M}, \mathsf{M}', d, d') \leftarrow \mathcal{A}(\mathsf{pk}) :$$

$$\mathsf{C.Open}(\mathsf{pk}, \mathsf{M}, c, d) = \mathsf{C.Open}(\mathsf{pk}, \mathsf{M}', c, d') = 1 \wedge \mathsf{M} \neq \mathsf{M}'] \leq \mathsf{negl}(\lambda)$$

Note that even though such a pair (c, d) may never be outputted by the commitment algorithm $\mathsf{C.Com}$, the binding property must hold even for adversaries that output $(c, d) \in \mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}, \mathsf{M}) \backslash \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})$.

Hiding. We call the scheme unconditionally (resp. computationally) hiding if for all (resp. PPT) algorithm $\mathcal{A}$ and any message $\mathsf{M} \in \mathcal{M}$, we have the following:[4]

$$\Pr[\mathsf{pk} \leftarrow \mathsf{C.Gen}(1^\lambda);\ b \leftarrow \{0,1\};\ c_0 \leftarrow \mathcal{C};\ (c_1, d) \leftarrow \mathsf{C.Com}(\mathsf{pk}, \mathsf{M});$$
$$b' \leftarrow \mathcal{A}(\mathsf{pk}, \mathsf{M}, c_b) : b = b'] \leq 1/2 + \mathsf{negl}(\lambda)$$

7.3.2 Digital Signature Schemes.

In this work we consider *deterministic* digital signature schemes; a scheme where the randomness of the signing algorithm is derived from the secret key and message. A deterministic digital signature scheme can be easily obtained from any digital signature scheme by using a pseudorandom function (PRF) for generating the randomness used in the signing algorithm (see for example [Kat10]).

Definition 7.2 (Digital Signatures)**.** *A digital signature scheme* S *with message space* $\{0,1\}^\ell$ *is a triple of polynomial time algorithms* ($\mathsf{S.KeyGen}$, $\mathsf{S.Sign}$, $\mathsf{S.Verify}$) *of the following form:*

$\mathsf{S.KeyGen}(1^\lambda.1^\ell) \to (\mathsf{vk}, \mathsf{sk})$: *The randomized key generation algorithm takes as input the security parameter* 1^λ *and the message length* 1^ℓ*, and outputs a verification key* vk *and signing key* sk.

$\mathsf{S.Sign}(\mathsf{sk}, \mathbf{x}) \to \sigma$: *The deterministic signing algorithm takes as inputs the signing key* sk *and message* $\mathbf{x} \in \{0,1\}^\ell$*, and outputs a signature* σ*.*

$\mathsf{S.Verify}(\mathsf{vk}, \mathbf{x}, \sigma) \to 1\backslash 0$: *The deterministic verification algorithm takes as inputs the verification key* vk*, message* $\mathbf{x} \in \{0,1\}^\ell$ *and signature* σ*, and outputs 1 or 0.*

A digital signature scheme is called correct *if the following holds for all* $\lambda, \ell \in \mathbb{N}$ *and* $\mathbf{x} \in \{0,1\}^\ell$*:*

$$\Pr[(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S.KeyGen}(1^\lambda, 1^\ell); \sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, \mathbf{x}) : \mathsf{S.Verify}(\mathsf{vk}, \mathbf{x}, \sigma) = 1] = 1 - \mathsf{negl}(\lambda)$$

We model the security of existential unforgeability under an adaptive chosen message attack ($\mathsf{eu\text{-}cma}$) using the following game between an adversary $\mathcal{A}$ and a challenger.

Setup: The challenger runs $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{S.KeyGen}(1^\lambda, 1^\ell)$ and provides $\mathcal{A}$ the verification key vk.

Signature Queries: When $\mathcal{A}$ submits a message $\mathbf{x} \in \{0,1\}^\ell$, the challenger responds by returning $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}, \mathbf{x})$.

Output: Finally, $\mathcal{A}$ outputs a pair $(\mathbf{x}, \sigma)$. The adversary $\mathcal{A}$ wins if $\mathsf{S.Verify}(\mathsf{vk}, \mathbf{x}, \sigma) = 1$ and $\mathbf{x}$ is not one of the messages $\mathcal{A}$ has made signature queries.

We define the advantage of an adversary $\mathcal{A}$ as the probability that $\mathcal{A}$ wins the above game, where the probability is taken over the randomness used by the challenger and the adversary.

Definition 7.3. *A digital signature scheme is called* $\mathsf{eu\text{-}cma}$ *secure if the advantage of the above game is negligible for all PPT adversaries.*

[4] We assume that the commitment space $\mathcal{C}$ is efficiently sampleable. Furthermore, as long as the hiding property holds, $\mathcal{C}$ may be larger than the set of all possible commitments. These types of commitment schemes show up in many of the lattice-based commitment schemes.

7.3.3 Arithmetic Circuit Representation

Let C be an arithmetic circuit over a ring R having ℓ input wires, one output wire and N gates. Here the gates are labelled by either $+$ (addition) or $\times$ (product) gates. The input wires are indexed by $1, \cdots, \ell$, the internal wires are indexed by $\ell+1, \cdots, \ell+N-1$ and the output wire has index $\ell+N$. We assume each gate takes only two incoming wires with multiple fan-out wires, where all the fan-out wires are indexed with the same index. We specify the topology of an arithmetic circuit by a function $\mathsf{topo} : \{\ell+1, \cdots, \ell+N\} \to \{+, \times\} \times \{1, \cdots, \ell+N-1\} \times \{1, \cdots, \ell+N-1\}$. They map a non-input wire to its first and second incoming wires in which these three wires are connected by either a gate labelled by $+$ or $\times$. For $(\star, i_1, i_2) \leftarrow \mathsf{topo}(i)$, we require that $i_1, i_2 < i$ where $\star \in \{+, \times\}$.

In the following, we consider $\mathcal{C}_\lambda$ as a collection of arithmetic circuits defined over a ring R_λ each having λ input wires. We also define the collection $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$. Further, unless stated otherwise, we simply call arithmetic circuits as circuits.

7.3.4 Attribute-Based Signature Scheme

An attribute-based signature scheme supporting the class of arithmetic circuits $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ and message space $\{0,1\}^*$ is defined by the following four probabilistic polynomial time algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify})$:

$\mathsf{Setup}(1^\lambda, 1^\ell) \to (\mathsf{mpk}, \mathsf{msk})$: The setup algorithm takes as input the security parameter 1^λ and the input length 1^ℓ of the circuits in $\mathcal{C}_\ell$, and outputs the master public key mpk and the master secret key msk.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x}) \to \mathsf{sk}_\mathbf{x}$: The signing key generation algorithm takes as input the master public key mpk, the master secret key msk and an attribute $\mathbf{x} \in \{0,1\}^\ell$, and outputs a signing key $\mathsf{sk}_\mathbf{x}$.

$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_\mathbf{x}, C, \mathsf{M}) \to \Sigma$: The signing algorithm takes as input the master public key mpk, a secret key $\mathsf{sk}_\mathbf{x}$ associated with an attribute $\mathbf{x}$, a circuit $C \in \mathcal{C}_\ell$ and a message $\mathsf{M} \in \{0,1\}^*$, and outputs a signature σ.

$\mathsf{Verify}(\mathsf{mpk}, \mathsf{M}, C, \Sigma) \to \mathsf{Valid}/\mathsf{Invalid}$: The verification algorithm takes as input the master public key mpk, a message M, a circuit C and a signature Σ, and outputs Valid or $\mathsf{Invalid}$.

Correctness. We require the following correctness condition to hold: for all $\lambda, \ell \in \mathbb{N}$, $\mathbf{x} \in \{0,1\}^\ell$, $C \in \mathcal{C}_\ell$ such that $C(\mathbf{x}) = 1$, it holds with all but negligible probability that $\mathsf{Verify}(\mathsf{mpk}, \mathsf{M}, C, \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_\mathbf{x}, C, \mathsf{M})) = \mathsf{Valid}$, where the probability is taken over the randomness used in $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and $\mathsf{sk}_\mathbf{x} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x})$.

We define two security notions for attribute-based signature schemes. The first notion is privacy, which requires the signature to not leak any information on the signer's attribute beyond the fact that the attribute satisfies the predicate. The other notion is unforgeability, which requires any collusion of signers are unable to forge a new signature with a predicate which is not satisfied by any attribute in the collusion even if they see signatures on messages of their choice.

Definition 7.4 (Privacy). *The security notion of privacy for an attribute-based signature scheme is defined by the following game between a challenger and an adversary $\mathcal{A}$:*

Setup. *The challenger runs* $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ *and gives* $(\mathsf{mpk}, \mathsf{msk})$ *to $\mathcal{A}$.*

Challenge. *$\mathcal{A}$ outputs a message $\mathsf{M} \in \{0,1\}^*$, two attributes $\mathbf{x}_0, \mathbf{x}_1 \in \{0,1\}^\ell$ and a circuit $C \in \mathcal{C}_\ell$ such that $C(\mathbf{x}_0) = C(\mathbf{x}_1) = 1$ to the challenger. The challenger first runs $\mathsf{sk}_{\mathbf{x}_0} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x}_0)$ and $\mathsf{sk}_{\mathbf{x}_1} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x}_1)$. Then, it picks a random bit $b \leftarrow \{0,1\}$ and returns to $\mathcal{A}$ the signature $\Sigma^* \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathbf{x}_b}, C, \mathsf{M})$ along with the two secret keys $(\mathsf{sk}_{\mathbf{x}_0}, \mathsf{sk}_{\mathbf{x}_1})$.*

Forgery. *Finally, $\mathcal{A}$ outputs a guess $b' \in \{0,1\}$ for b.*

The advantage of $\mathcal{A}$ is defined as $|\Pr[b' = b] - 1/2|$. We say that the attribute-based signature scheme is computationally private *if the advantage of any PPT algorithm $\mathcal{A}$ is negligible. We say it is* unconditionally private *if the advantage of any (possibly inefficient) algorithm $\mathcal{A}$ is negligible.*

Definition 7.5 (Unforgeability)**.** *The security notion of adaptively unforgeable for an attribute-based signature scheme is defined by the following game between a challenger and an adversary $\mathcal{A}$:*

Setup. *The challenger runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^\ell)$ and gives mpk to $\mathcal{A}$.*

Queries. *$\mathcal{A}$ may adaptively make the following queries to the challenger:*

- **Signing.** *$\mathcal{A}$ submits a signing query on any attribute, message and circuit tuple $(\mathbf{x}, \mathsf{M}, C)$ such that $C(\mathbf{x}) = 1$ to the challenger. The challenger runs $\mathsf{sk}_\mathbf{x} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x})$. Then, it returns the signature $\Sigma \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_\mathbf{x}, C, \mathsf{M})$ to $\mathcal{A}$.*
- **Key reveal.** *$\mathcal{A}$ submits a key reveal query on any attribute $\mathbf{x}$ to the challenger. The challenger returns the signing key $\mathsf{sk}_\mathbf{x} \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x})$ to $\mathcal{A}$.*

Forgery. *Finally, $\mathcal{A}$ outputs a signature $(\mathsf{M}^*, C^*, \Sigma^*)$.*

The adversary $\mathcal{A}$ wins the game if the following three conditions hold:

(i) $\mathsf{Verify}(\mathsf{mpk}, \mathsf{M}^*, C^*, \Sigma^*) = \mathsf{Valid}$,

(ii) *Adversary $\mathcal{A}$ did not submit a key reveal query for $\mathbf{x}$ such that $C^*(\mathbf{x}) = 1$,*

(iii) *Adversary $\mathcal{A}$ did not submit a signing query on $(\mathbf{x}, \mathsf{M}^*, C^*)$ for any $\mathbf{x}$ such that $C^*(\mathbf{x}) = 1$*

The advantage of $\mathcal{A}$ is defined as the probability of $\mathcal{A}$ winning the above game. We say that the attribute-based signature scheme is adaptively unforgeable *if the advantage of any PPT algorithm $\mathcal{A}$ is negligible.*

7.3.5 General Multi-Forking Lemma with Oracle Access

Here we state and prove an extended version of the forking lemma of [PS00], which will play a central role in our proof of security of our ABS scheme. Our forking lemma analyzes the output behavior of an algorithm $\mathcal{A}$ when run multiple times on related inputs, instead of when only run twice, while also providing it with oracle access to a *deterministic* algorithm $\mathcal{O}$.

Lemma 7.1 (General Multi-Forking Lemma with Oracle Access)**.** *Fix an integer $q \geq 1$ and a set $\mathcal{H}$ of size $h \geq 2$. Let $\mathcal{A}$ be a randomized algorithm that has oracle access to some* deterministic *algorithm $\mathcal{O}$, where on input $\mathsf{param}, h_1, \cdots, h_q$, algorithm $\mathcal{A}$ returns a pair; the first element is an integer in the range $0, \cdots, q$ and the second element is what we refer to as a* side output*. Let IG be a randomized algorithm called the* input generator*. The accepting probability of $\mathcal{A}$, denoted acc, is defined as the probability that $J \geq 1$ in the experiment below:*

$$(\mathsf{param}, \overline{\mathsf{param}}) \leftarrow \mathsf{IG};\ h_1, \cdots, h_q \leftarrow \mathcal{H};\ (J, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}}, \cdot)}(\mathsf{param}, h_1, \cdots, h_q).$$

```
Algorithm $\mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param})$

Pick coin $\rho$ for $\mathcal{A}$ at random.
$h_1^{(1)}, \cdots, h_q^{(1)} \leftarrow \mathcal{H}$
$(I^{(1)}, \sigma^{(1)}) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param}, h_1^{(1)}, \cdots, h_q^{(1)}; \rho)$
if $I^{(1)} = 0$ then return $(0, \{\epsilon_k\}_{k\in[\ell]})$
for $k = 2$ to $\ell$ do
    $h_{I^{(1)}}^{(k)}, \cdots, h_q^{(k)} \leftarrow \mathcal{H}$
    $(I^{(k)}, \sigma^{(k)}) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param}, h_1^{(1)}, \cdots, h_{I^{(1)}-1}^{(1)}, h_{I^{(1)}}^{(k)}, \cdots h_q^{(k)}; \rho)$
if $I^{(1)} = I^{(k)}$ and $h_{I^{(1)}}^{(k)} \neq h_{I^{(1)}}^{(k')}$ for all $k, k' \in [\ell]$ then
    return $(1, \{\sigma^{(k)}\}_{k\in[\ell]})$
else
    return $(0, \{\epsilon_k\}_{k\in[\ell]})$.
```

Figure 7.1: Description of the forking algorithm $\mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}$.

For a positive integer $\ell \geq 2$, the forking algorithm $\mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}$ *associated to $\mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}$ is a randomized oracle algorithm that takes input* param *and proceeds as in Figure 7.1, where $\{\epsilon_k\}_{k\in[\ell]}$ denotes an arbitrary set of strings. Let*

$$\mathsf{frk} = \Pr[(\mathsf{param}, \overline{\mathsf{param}}) \leftarrow \mathsf{IG};\ (b, \{\sigma_k\}_{k\in[\ell]}) \leftarrow \mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param})\ :\ b = 1].$$

Then,

$$\mathsf{frk} \geq \mathsf{acc} \cdot \left(\left(\frac{\mathsf{acc}}{q} \right)^{\ell-1} - \frac{f(\ell)}{h} \right), \tag{7.1}$$

where $f(\ell)$ is some universal positive valued function that only depends on the value ℓ.

Proof. For any input $x = (\mathsf{param}, \overline{\mathsf{param}})$, denote $\mathsf{acc}(x)$ as the probability that $J \geq 1$ in the following experiment:

$$h_1, \cdots, h_q \leftarrow H;\ (J, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param}, h_1, \cdots, h_q).$$

Also, let $\mathsf{frk}(x) = \Pr[(b, \{\sigma_k\}_{k\in[\ell]}) \leftarrow \mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param})\ :\ b = 1]$. We claim that there exists some universal positive valued function $f(\ell)$ such that for all x,

$$\mathsf{frk}(x) \geq \mathsf{acc}(x) \cdot \left(\left(\frac{\mathsf{acc}(x)}{q} \right)^{\ell-1} - \frac{f(\ell)}{h} \right). \tag{7.2}$$

By taking the expectation of $\mathsf{frk}(x)$ over $x = (\mathsf{param}, \overline{\mathsf{param}}) \leftarrow \mathsf{IG}$ and using the fact $\mathbb{E}[\mathsf{acc}(x)^\ell] \geq \mathbb{E}[\mathsf{acc}(x)]^\ell$ (which follows from Jensen's inequality), we obtain Eq. (7.1). Therefore, to prove the claim, we must prove Eq. (7.2). Now, for any input x, with the probabilities taken over the coin tosses of $\mathsf{F}_{\mathcal{A},\ell}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}(\mathsf{param})$, $\mathsf{frk}(x)$ is equivalent to the following.

$$\mathsf{frk}(x) = \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell])\ \wedge\ (I^{(1)} \geq 1)\ \wedge\ (h_{I^{(1)}}^{(k)} \neq h_{I^{(1)}}^{(k')} \text{ for all } k, k' \in [\ell])\right]$$

$$
\begin{aligned}
&= \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1)\right] \\
&\qquad - \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1) \;\wedge\; (h^{(k)}_{I^{(1)}} = h^{(k')}_{I^{(1)}} \text{ for some } k, k' \in [\ell])\right] \\
&\geq \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1)\right] \\
&\qquad - \Pr\left[(I^{(1)} \geq 1) \;\wedge\; (h^{(k)}_{I^{(1)}} = h^{(k')}_{I^{(1)}} \text{ for some } k, k' \in [\ell])\right] \\
&= \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1)\right] - \Pr\left[(I^{(1)} \geq 1)\right] \cdot (1 - \prod_{k=1}^{\ell-1} \frac{h-k}{h})
\end{aligned}
$$

Here, we can rewrite $1 - \prod_{k=1}^{\ell-1} \frac{h-k}{h} = \frac{1}{h} \cdot \left(\sum_{k=0}^{\ell-2} \alpha_k(\ell) \cdot \frac{1}{h^k} \right)$, where $(\alpha_k(\ell))_{k=0}^{\ell-2}$ are functions that only depend on ℓ. Since $h \geq 1$, we can always upper bound the right hand side by $f(\ell)/h$ using some positive valued function $f(\ell)$ that only depends on ℓ, where for example, we can use $f(\ell) = (\ell - 1) \cdot \max\{|\alpha_k(\ell)|\}_{k=0}^{\ell-2}$. Here, note that $f(\ell)$ is some universal function that depends neither on $\mathcal{A}$ nor $\mathcal{O}$. Therefore, we can further rewrite the inequality as follows:

$$
\mathsf{frk}(x) \geq \Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1)\right] - \frac{\mathsf{acc}(x) \cdot f(\ell)}{h}.
$$

Hence, it remains to show that $\Pr\left[\left(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]\right) \;\wedge\; \left(I^{(1)} \geq 1\right)\right] \geq \mathsf{acc}(x)^\ell / q^{\ell-1}$. Let $\mathcal{R}$ denote the set from which $\mathcal{A}$ draws its random coins. For each $i \in [q]$, let $X_i : \mathcal{R} \times \mathcal{H}^{i-1} \to [0,1]$ be defined by setting $X_i(\rho, h_1, \cdots, h_{i-1})$ to

$$
\Pr[h_i, \cdots, h_q \leftarrow \mathcal{H} \;;\; (J, \sigma) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}}, \cdot)}(\mathsf{param}, h_1, \cdots, h_q; \rho) \;:\; J = i]
$$

for all $\rho \in \mathcal{R}$ and $h_1, \cdots, h_{i-1} \in \mathcal{H}$. Here, regard X_i as a random variable over the uniform distribution on its domain. Then,

$$
\begin{aligned}
&\Pr\left[(I^{(1)} = I^{(k)} \text{ for all } k \in [\ell]) \;\wedge\; (I^{(1)} \geq 1)\right] \\
&= \sum_{i=1}^{q} \Pr\left[I^{(k)} = i \text{ for all } k \in [\ell]\right] \\
&= \sum_{i=1}^{q} \Pr[I^{(1)} = i] \cdot \Pr[I^{(2)} = i \mid I^{(1)} = i] \cdots \Pr[I^{(\ell)} = i \mid I^{(k)} = i \text{ for } k \in [\ell - 1]] \\
&= \sum_{i=1}^{q} \left(\Pr[I^{(1)} = i] \cdot \prod_{k=2}^{\ell} \Pr[I^{(k)} = i \mid I^{(1)} = i] \right) && (7.3) \\
&= \sum_{i=1}^{q} \sum_{\rho, h_1, \cdots, h_{i-1}} X_i(\rho, h_1, \cdots, h_{i-1})^\ell \cdot \frac{1}{|\mathcal{R}| \cdot |\mathcal{H}|^{i-1}} && (7.4) \\
&= \sum_{i=1}^{q} \mathbb{E}[X_i^\ell] \quad \geq \quad \sum_{i=1}^{q} \mathbb{E}[X_i]^\ell. && (7.5)
\end{aligned}
$$

Here Eq. (7.3) follows from independence of $I^{(k)}$ and $I^{(k')}$ for $k, k' \in [2, \ell]$, Eq. (7.4) follows from the fact that once $\rho, h_1, \cdots, h_{i-1}$ are fixed $\Pr[I^{(1)} = i] = \Pr[I^{(k)} = i \mid I^{(1)} = i] = X_i(\rho, h_1, \cdots, h_{i-1})$ and Eq. (7.5) follows from Jensen's inequality where we use the fact that $f(x) = x^\ell$ is a convex function. Finally, using Holder's inequality, we obtain

$$
\sum_{i=1}^{q} \mathbb{E}[X_i]^\ell \geq \frac{1}{q^{\ell-1}} \cdot \left(\sum_{i=1}^{q} \mathbb{E}[X_i] \right)^\ell = \frac{1}{q^{\ell-1}} \cdot \mathsf{acc}(x)^\ell.
$$

This completes the proof of Eq. (7.1), hence concluding our claim. □

Remarks. As can be checked from the proof, we can set the function $f(\ell)$ so that in case $\ell = 2$, we have $f(2) = 1$. Therefore, by setting the deterministic oracle $\mathcal{O}$ to be an oracle that outputs nothing, the above lemma implies the general forking lemma of [BN06].

7.4 Gap-Σ-Protocols and Non-Interactive Zero-Knowledge Proofs

Before presenting the main tools we use in this chapter, we recall some standard notions. A *language* $\mathcal{L} \subseteq \{0,1\}^*$ is said to have polynomial time recognizable *relation* $\mathcal{R} \subseteq \{0,1\}^* \times \{0,1\}^*$ if $\mathcal{L} = \{x \mid \exists w \text{ s.t. } (x,w) \in \mathcal{R}\}$ where $|w| \leq \mathsf{poly}(|x|)$. We call the string w a *witness* to the *statement* $x \in \mathcal{L}$. Occasionally, we write $\mathcal{L}_{\mathcal{R}}$ to emphasize that the language $\mathcal{L}$ is induced by the relation $\mathcal{R}$. In the following, we implicitly assume that all languages and relations are parameterized by the security parameter. Furthermore, an non-interactive proof system is defined as follows:

Definition 7.6 (Non-Interactive Proof System)**.** *Let $\mathcal{R}$ be a relation with an associated language $\mathcal{L}_{\mathcal{R}}$. Then, a non-interactive proof system for language $\mathcal{L}_{\mathcal{R}}$ consists of two algorithms $(\mathcal{P}, \mathcal{V})$, where $\mathcal{P}$ may be randomized and $\mathcal{V}$ is deterministic such that for any pair $(x,w) \in \mathcal{R}$, if $\pi \leftarrow \mathcal{P}(x,w)$ then $\mathcal{V}(x,\pi)$ outputs 1, which signifies that the proof π was correct, and otherwise outputs 0.*

7.4.1 Gap-Σ-Protocols

Σ-protocols are a special type of 3-round interactive proof systems that is also a proof of knowledge. Below, we define (a special type of) the *gap*-Σ-protocol, which is a generalization of the standard Σ-protocol where we allow the extracted witness to lie in a slightly larger space than the actual witness being proven during the protocol. Furthermore, the special soundness is defined for cases where more than 2 valid transcripts are required to extract a witness. These non-standard formalizations are required, since most of the lattice-based Σ-protocols are not captured by the standard formalizations.

Later on, we see that many of the results known to the standard Σ-protocols can be translated to the gap-Σ-protocol settings. Readers familiar with standard Σ-protocols that are not interested in the lattice-based instantiations may safely skip this section, since our generic construction of attribute-based signature scheme can be constructed from standard Σ-protocols as well.

Definition 7.7 (Gap-Σ-protocols)**.** *Let m be an integer constant and t an integer-valued function of the security parameter. Let $(\mathcal{P}, \mathcal{V})$ be a two-party protocol, where $\mathcal{V}$ is PPT, and let $\mathcal{L}, \mathcal{L}' \subseteq \{0,1\}^*$ be languages with witness relations $\mathcal{R}, \mathcal{R}'$ such that $\mathcal{R} \subseteq \mathcal{R}'$. Then $(\mathcal{P}, \mathcal{V})$ is called a gap-$\Sigma_{m,t}$-protocol for relations $(\mathcal{R}, \mathcal{R}')$ with challenge space $\mathcal{C} = \{0, 1, \cdots, m-1\}^t$, if it satisfies the following conditions:*

- **3-move form:** *The protocol is of the following form:*
  - · *The prover $\mathcal{P}$, on input $(x,w) \in \mathcal{R}$, sends a* commitment *α to $\mathcal{V}$.*
  - · *The verifier $\mathcal{V}$ samples a* challenge *$\beta \leftarrow \mathcal{C}$ and sends it to $\mathcal{P}$.*
  - · *The prover $\mathcal{P}$ sends a* response *γ to $\mathcal{V}$, and $\mathcal{V}$ outputs 1 or 0 based on the protocol transcript (α, β, γ).*

  The protocol transcript (α, β, γ) is called a valid transcript *if the verifier $\mathcal{V}$ outputs 1, i.e., accepts the protocol run.*

- **Completeness:** *Whenever* $(x, w) \in \mathcal{R}$, $\mathcal{V}$ *accepts with probability* 1.

- **Soundness:** *If* $(x, w) \notin \mathcal{R}$, *then any cheating (possibly inefficient) prover* $\mathcal{P}^*$ *succeeds with probability at most* $(\frac{m-1}{m})^t$. *We call this value the* soundness error.

- **Special gap-soundness:** *There exists a PPT algorithm* $\mathcal{E}$ *(the* knowledge extractor*) which takes* m *valid transcripts* $\{(\alpha, \beta_i, \gamma_i)\}_{i\in[m]}$ *for some statement* $x \in \mathcal{L}$, *where there exists at least one index* $j \in [t]$ *such that* $\{\beta_{i,j}\}_{i\in[m]} = \{0, 1, \cdots, m-1\}$ *as inputs, and outputs* w *such that* $(x, w) \in \mathcal{R}'$. *Here* $\beta_{i,j}$ *denotes the* j*-th value of the string* β_i. *Note that the knowledge extractor outputs a witness in the* gap *relation.*

- **Special honest-verifier zero-knowledge (HVZK):** *There exists a PPT algorithm* $\mathcal{S}$ *(the* HVZK simulator*) taking* $x \in \mathcal{L}$ *as input, that outputs* (α, β, γ) *whose distribution is indistinguishable from an accepting protocol transcript generated by a real protocol run. Although no guarantees on the outputs are made, the simulator* $\mathcal{S}$ *is also defined over the inputs* $x \notin \mathcal{L}$.

We call the gap-$\Sigma_{m,t}$*-protocol computationally (resp. statistically) special HVZK if the simulated transcript is computationally (resp. statistically) indistinguishable from a real transcript.*

Lastly, we say the gap-Σ*-protocol has* high-commitment entropy *if for all* $(x, w) \in \mathcal{R}$ *and* α, *the probability that an honestly generated commitment by* $\mathcal{P}$ *takes on the value* α *is negligible.*

We omit the subscript (m, t) of the gap-$\Sigma_{m,t}$-protocol whenever it is irrelevant to the context. Occasionally, we omit t and simply write gap-Σ_m-protocol to emphasize that the soundness error is negligible in the security parameter. We note that the standard Σ-protocol is a special case of the gap-Σ-protocol where $m = 2, \mathcal{R} = \mathcal{R}'$. In this case the soundness error will simply be 2^{-t} and special gap-soundness implies special soundness, since if there exists an index $j \in [t]$ for which the binary strings (i.e., the challenges) differ, then it implies that the two challenges are different. Finally, we assume without loss of generality that all of the gap-Σ-protocols we consider in this work have high-commitment entropy, since the condition can be easily met by appending a super-logarithmic number of public random bits to the commitments.

Often times, the gap in the relations allows for much more efficient schemes, and do not affect their usefulness in practice as long as $\mathcal{R}'$ is still a sufficiently hard relation, e.g., [FO97, DF02, AJLA+12, BCK+14]. Here, note that for the case $m \geq 3$, we define a stronger notion of special gap-soundness compared to the most general definitions one can consider for gap-Σ protocols, i.e., we require a stronger condition than $\beta_i \neq \beta_j$ for $i \neq j \in [m]$, since in many cases in lattice-based Σ-protocols, we need to impose this stronger restriction to extract a witness. This is mainly due to the fact that in lattice-based schemes, gap-$\Sigma_{m,t}$-protocols are constructed from running t gap-$\Sigma_{m,1}$-protocols in parallel, and we require the stronger restriction to extract a witness from one of the inner gap-$\Sigma_{m,1}$-protocols, see, e.g., [KTX08, LNSW13]. We note that for simplicity, in this work we only consider gap-Σ-protocols that are complete with probability 1. Namely, our formalization does not capture those gap-Σ-protocols that are based on the rejection sampling technique such as [Lyu09, Lyu12, BCK+14, BDOP16]. This is purely because these gap-Σ-protocols do not offer zero-knowledge proofs for relations that are strong enough for our application in mind.[5]

[5] Note that we intentionally disregard [BKLP15] from our work. Although they offer an attractive rejection sampling-based gap-Σ-protocol for proving arbitrary arithmetic operations that are more efficient than those of [XXW13] which we use in Section 7.6, we were not able to verify the correctness of their proof sketch. In particular, the knowledge extractor for the protocol for proving multiplicative relations could not be constructed as stated in their paper.

Before continuing, we provide the following simple composition lemma for gap-Σ-protocols for completeness.

Lemma 7.2. *Given a gap-$\Sigma_{m,1}$-protocol for relations $(\mathcal{R}, \mathcal{R}')$, we can construct a gap-$\Sigma_{m,t}$-protocol for the same relations by running t instances of the gap-$\Sigma_{m,1}$-protocol in parallel. In particular, when t is super-logarithmic in the security parameter*[6] *, the soundness error of the gap-$\Sigma_{m,t}$-protocol is negligible.*

Proof. It is straightforward to check that completeness and special HVZK hold. The condition on soundness error holds, since in each internal gap-$\Sigma_{m,1}$-protocol, a cheating prover has at most probability $1 - 1/m$ of succeeding. Furthermore, special soundness holds too, since if there exists such an index $j \in [t]$ such that $\{\beta_{i,j}\}_{i \in [m]} = \{0, 1, \cdots, m-1\}$, we can use the knowledge extractor of the internal gap-$\Sigma_{m,1}$-protocol to extract the witness w such that $(x, w) \in \mathcal{R}'$ from the j-th run. $\square$

Finally, we formally describe the Fiat-Shamir transformation [FS86] (who [BR93] attributes to Blum), which is a technique to make any (gap-)Σ-protocol into a non-interactive proof system by using a cryptographic hash function.

Definition 7.8. *Let $(\mathcal{P}, \mathcal{V})$ be a gap-Σ-protocol with relation $(\mathcal{R}, \mathcal{R}')$, and $H(\cdot)$ a hash function with range equal to the verifier's challenge space $\mathcal{C}$. The* Fiat-Shamir *transformation of gap-Σ is the non-interactive proof system $(\mathcal{P}^H, \mathcal{V}^H)$ defined as follows:*

$\mathcal{P}^H(x, w)$: *Run $\mathcal{P}(x, w)$ to obtain a commitment α, and compute $\beta \leftarrow H(x, \alpha)$. Then complete the run of $\mathcal{P}$ with β as the challenge to get the response γ. Finally output the pair $\pi = (\alpha, \beta, \gamma)$ as the proof.*

$\mathcal{V}^H(x, \pi = (\alpha, \beta, \gamma))$: *Return the output of $\mathcal{V}(\alpha, \beta, \gamma)$ if $\beta = H(x, \alpha)$ and 0 otherwise.*

7.4.2 Non-Interactive Zero-Knowledge Proof Systems

We formalize the notion of non-interactive zero-knowledge (NIZK) proof systems in the *explicitly programmable* random oracle model [Wee09], where the zero-knowledge (ZK) simulator is allowed to explicitly program the random oracle. We follow the notations provided in [FKMV12] for presentation. Namely, we model the ZK simulator of a NIZK proof system as a stateful PPT algorithm $\mathcal{S}$ that can operate in two modes: $(h, \mathsf{st}) \leftarrow \mathcal{S}(1, \mathsf{st}, q)$ takes care of answering random oracle queries, and $(\pi, \mathsf{st}) \leftarrow \mathcal{S}(2, \mathsf{st}, x)$ simulates the proof. Here, the calls to $\mathcal{S}(1, \cdots)$ and $\mathcal{S}(2, \cdots)$ share the common state st that is updated after each invocation of the simulator. Furthermore, we define three algorithms $\mathcal{S}_1, \mathcal{S}_2, \hat{\mathcal{S}}_2$ that run simulator $\mathcal{S}$ internally: $\mathcal{S}_1(q)$ returns the first output of $(h, \mathsf{st}) \leftarrow \mathcal{S}(1, \mathsf{st}, q)$, $\mathcal{S}_2(x, w)$ ignores the second input w and returns the first output of $(\pi, \mathsf{st}) \leftarrow \mathcal{S}(2, \mathsf{st}, x)$ if and only if $(x, w) \in \mathcal{R}$ (or equivalently $x \in \mathcal{L}$), and $\hat{\mathcal{S}}_2(x)$ is essentially the same as $\mathcal{S}_2(x, w)$ except that it does not take a second input w and is also defined for inputs such that $x \notin \mathcal{L}$. Observe that $\mathcal{S}_2$ and $\hat{\mathcal{S}}_2$ are identical for inputs $x \in \mathcal{L}$, and unlike $\mathcal{S}_2$, $\hat{\mathcal{S}}_2$ may be invoked to simulate proofs for invalid statements.

Definition 7.9 (Non-Interactive Zero-Knowledge Proof System)**.** *Let $\mathcal{R}$ be a relation with an associated language $\mathcal{L}_{\mathcal{R}}$. We say a non-interactive proof system $(\mathcal{P}, \mathcal{V})$ is a statistical NIZK proof*

[6] Up until this point, we have made the security parameter implicit for the simplicity of presentation. Later on, it will be clear from context that everything, including the relations, are parameterized by the security parameter.

system for language $\mathcal{L}_\mathcal{R}$ with a (PPT) ZK simulator $\mathcal{S}$ in the random oracle model, if for any algorithm $\mathcal{D}$ we have

$$\left|\Pr[\mathcal{D}^{H(\cdot),\mathcal{P}^H(\cdot,\cdot)}(1^\lambda) = 1] - \Pr[\mathcal{D}^{\mathcal{S}_1(\cdot),\mathcal{S}_2(\cdot,\cdot)}(1^\lambda) = 1]\right| = \mathsf{negl}(\lambda),$$

where $H(\cdot)$ is modeled as a random oracle, and both $\mathcal{P}$ and $\mathcal{S}_2$ output $\perp$ if $(x, w) \notin \mathcal{R}$. It is called a computational NIZK proof system in case the above holds only for all PPT algorithms $\mathcal{D}$.

It is a well known fact that in the random oracle model, the Fiat-Shamir transformation of any Σ-protocol is a NIZK proof system. It is straightforward to prove that it is also the case for gap-Σ-protocols, as we state in the following lemma.

Lemma 7.3 (Fiat-Shamir NIZK Proof Systems)**.** *Let $(\mathcal{P}, \mathcal{V})$ be a gap-Σ-protocol with relation $(\mathcal{R}, \mathcal{R}')$ that is computationally (resp. statistically) special HVZK, and $H(\cdot)$ a hash function with range equal to the verifier's challenge space $\mathcal{C}$. Then, in the random oracle model, the non-interactive proof system $(\mathcal{P}^H, \mathcal{V}^H)$ obtained by the Fiat-Shamir transformation of gap-Σ is a computational (resp. statistical) non-interactive zero-knowledge proof system for the language $\mathcal{L}_\mathcal{R}$.*

Proof sketch. To prove that the proof system $(\mathcal{P}^H, \mathcal{V}^H)$ is a NIZK proof system for the language $\mathcal{L}_\mathcal{R}$, it suffices to show that there exists a ZK simulator $\mathcal{S}$ as in the above Definition 7.9. Below, we construct $\mathcal{S}$ by invoking the HVZK simulator $\mathcal{S}_\Sigma$ of the underlying gap-Σ-protocol $(\mathcal{P}, \mathcal{V})$:

- $\mathcal{S}(1, \mathsf{st}, q = (x, \alpha)) \to (h = \beta, \mathsf{st})$: To answer random oracle queries, it searches the table $\mathcal{T}_H$ kept in the state st whether an output for $q = (x, \alpha)$ is already defined. If so it returns the previously defined assigned value. If not, it samples a uniformly random value $\beta \leftarrow \mathcal{C}$ and stores $(q = (x, \alpha), h = \beta)$ in the table. Note that this corresponds to algorithm $\mathcal{S}_1$.

- $\mathcal{S}(2, \mathsf{st}, x) \to (\pi = (\alpha, \beta, \gamma), \mathsf{st})$: To simulate a proof for the statement $x \in \mathcal{L}_\mathcal{R}$, it runs the HVZK simulator $\mathcal{S}_\Sigma$ on input x to obtain a proof (α, β, γ). Then, it updates the table $\mathcal{T}_H$ by adding $(q = (x, \alpha), h = \beta)$. If $\mathcal{T}_H$ happens to be already defined on input $q = (x, \alpha)$, $\mathcal{S}$ aborts. This completely specifies algorithm $\mathcal{S}_2$ as required. Observe that the simulator $\mathcal{S}$ can also be run on statements $x \notin \mathcal{L}_\mathcal{R}$ using the above method, since $\mathcal{S}_\Sigma$ is well-defined for $x \in \mathcal{L}$ as well. In particular, the above description for $\mathcal{S}$ also specifies algorithm $\hat{\mathcal{S}}_2$ as well.

Since, we only consider gap-Σ-protocols with high-commitment entropy, the probability of simulator $\mathcal{S}$ aborting is negligible, which ends the proof sketch. $\square$

In the following, we use the above algorithm $\mathcal{S}$ as the ZK simulator for a NIZK proof system $(\mathcal{P}^H, \mathcal{V}^H)$ based on the Fiat-Shamir transformation of a gap-Σ-protocol $(\mathcal{P}, \mathcal{V})$. Note that we do not explicitly define the soundness property of the NIZK proof system, since this property will be implicitly implied when we construct a knowledge extractor during the security proof.

7.5 Generic Construction of Attribute-based Signatures

Overview and Preparation. Before presenting our construction, we provide a brief overview. The main idea is that the attribute authority issues a signature σ (i.e., certificate) on an attribute $\mathbf{x} \in \{0, 1\}^\ell$ to certify that the intended signer is allowed to sign a message on behalf of that attribute. To sign anonymously, the signer proves the following facts in zero-knowledge: the signature issued by the attribute authority is valid and the corresponding secret attribute satisfies the public circuit $C \in \mathcal{C}_\ell$ (i.e., policy) attached to the message. To do so, the signer first commits to the signature, the attributes and all of the values assigned to the internal wires of the circuit C on input the attribute $\mathbf{x}$. Then, he proves in zero-knowledge that the values inside the commitments satisfy the equations Eq. (7.6 - 7.8) in our construction.

Therefore, the tools we need to prepare are a digital signature scheme, a commitment scheme and a NIZK proof system to prove the above relations between committed values. For the rest of the overview, we describe the relations and languages we require for our NIZK proof system. Our construction relies on a gap-Σ-protocol for the relations $(\mathcal{R}_{\mathsf{ABS}}, \mathcal{R}'_{\mathsf{ABS}})$ defined below and employs the Fiat-Shamir transformation provided in Definition 7.8 to turn it in into a NIZK proof system. In the following, x_i for $i \in [\ell+1, \ell+N-1]$ denotes the values assigned to the i-th (internal) wire of C on input $\mathbf{x} = (x_1, \cdots, x_\ell)$ and $\mathsf{vk}_{\mathsf{Sign}}$, $\mathsf{pk}_{\mathsf{Com}}$ denotes the verification key and public commitment key of the underlying digital signature scheme and commitment scheme, respectively. Then the relation $\mathcal{R}_{\mathsf{ABS}}$ is defined as follows:

$$\mathcal{R}_{\mathsf{ABS}} = \Big\{ \Big(\mathsf{statement} = \big(\mathsf{vk}_{\mathsf{Sign}}, \mathsf{pk}_{\mathsf{Com}}, C \in \mathcal{C}_\ell, c_\sigma, (c_i)_{i=1}^{\ell+|C|-1}\big),$$
$$\mathsf{witness} = \big(\mathbf{x} = (x_1, \cdots, x_\ell), \sigma, d_\sigma, (d_i)_{i=1}^{\ell+|C|-1}\big)\Big) \Big|$$

the committed values in $c_\sigma, (c_i)_{i=1}^{\ell+|C|-1}$ satisfy the following conditions

- $\mathsf{S.Verify}(\mathsf{vk}_{\mathsf{Sign}}, \mathbf{x}, \sigma) = 1$
- $x_i = x_{i_1} \star_i x_{i_2}$ for $i \in [\ell+1, \ell+|C|-1]$ where $(\star_i, i_1, i_2) \leftarrow \mathsf{topo}_C(i)$
- $1 = x_{(\ell+|C|)_1} \star_{\ell+|C|} x_{(\ell+|C|)_2}$ where $(\star_{\ell+|C|}, i_{(\ell+|C|)_1}, i_{(\ell+|C|)_2}) \leftarrow \mathsf{topo}_C(\ell+|C|)$
- $(c_\sigma, d_\sigma) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}_{\mathsf{Com}}, \sigma)$ and $(c_i, d_i) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}_{\mathsf{Com}}, x_i)$ for $i \in [\ell+|C|-1]\Big\}$

Here, recall that $\mathcal{D}_{\mathsf{Com}}(\mathsf{pk}_{\mathsf{Com}}, \mathsf{M})$ is the set of all possible outputs of the commitment algorithm $\mathsf{C.Com}(\mathsf{pk}_{\mathsf{Com}}, \mathsf{M})$ that we require to have an efficient method for checking membership of an element. We simply define the corresponding language $\mathcal{L}_{\mathsf{ABS}}$ as the language $\mathcal{L}_{\mathcal{R}_{\mathsf{ABS}}}$ induced by the relation $\mathcal{R}_{\mathsf{ABS}}$. Furthermore, the gap-relation $\mathcal{R}'_{\mathsf{ABS}}$ is defined analogously to $\mathcal{R}_{\mathsf{ABS}}$ except that we replace the last condition as follows:

- $(c_\sigma, d_\sigma) \in \mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}_{\mathsf{Com}}, \sigma)$ and $(c_i, d_i) \in \mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}_{\mathsf{Com}}, x_i)$ for $i \in [\ell+|C|-1]$

The only difference between the two relations are the condition on the commitment and opening pairs. Namely, it is only required that the pairs are in the set $\mathcal{D}_{\mathsf{G\text{-}Com}}(\cdot)$ and not in the more restricted set $\mathcal{D}_{\mathsf{Com}}(\cdot)$. Recall that $\mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}_{\mathsf{Com}}, \mathsf{M})$ is the set of all commitment and opening pairs that the opening algorithm outputs 1 on message M. This set is efficiently recognizable, since we can use the opening algorithm to check if the pair is included in $\mathcal{D}_{\mathsf{G\text{-}Com}}(\mathsf{pk}_{\mathsf{Com}}, \mathsf{M})$. As we noted in Section 7.4.1, we require this gap-relation $\mathcal{R}'_{\mathsf{ABS}}$ purely for technical reasons, since in many of the lattice-based Σ-protocols we can only extract witnesses that lie in a slightly larger space than the actual witnesses being proven in the actual protocol. Similarly to above, we define the language $\mathcal{L}'_{\mathsf{ABS}}$ as the language $\mathcal{L}_{\mathcal{R}'_{\mathsf{ABS}}}$ induced by the relation $\mathcal{R}'_{\mathsf{ABS}}$.

For simplicity, in the following we omit $\mathsf{vk}_{\mathsf{Sign}}$ and $\mathsf{pk}_{\mathsf{Com}}$ from the statement, since they are fixed by the Setup algorithm and all signers use the same $\mathsf{vk}_{\mathsf{Sign}}$ and $\mathsf{pk}_{\mathsf{Com}}$.

Construction. Here, we provide our attribute-based signature scheme for unbounded (arithmetic) circuits. In the following, assume a digital signature scheme ($\mathsf{S.KeyGen}$, $\mathsf{S.Sign}$, $\mathsf{S.Verify}$), a commitment scheme ($\mathsf{C.Gen}, \mathsf{C.Com}, \mathsf{C.Open}$) and a NIZK proof system for the relation $\mathcal{R}_{\mathsf{ABS}}$.

$\mathsf{Setup}(1^\lambda, 1^\ell)$: On input the security parameter 1^λ and the input length 1^ℓ for the family of circuits $\mathcal{C}_\ell$, generate a verification key and a signing key $(\mathsf{vk}_{\mathsf{Sign}}, \mathsf{sk}_{\mathsf{Sign}}) \leftarrow \mathsf{S.KeyGen}(1^\lambda, 1^\ell)$

and a public commitment key $\mathsf{pk}_{\mathsf{Com}} \leftarrow \mathsf{C.Gen}(1^\lambda)$. Then output

$$\mathsf{mpk} = (\mathsf{vk}_{\mathsf{Sign}}, \mathsf{pk}_{\mathsf{Com}}, H(\cdot), G(\cdot)) \quad \text{and} \quad \mathsf{msk} = (\mathsf{sk}_{\mathsf{Sign}}).$$

Here, $H(\cdot)$ and $G(\cdot)$ are hash functions used by the NIZK proof system and by algorithm Sign, respectively, which are programmed as random oracles in the security reduction. Further, we assume the output space of $G(\cdot)$ to be $\{0,1\}^\ell$.[7]

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x})$: On input $\mathbf{x} = (x_1 \cdots, x_\ell) \in \{0,1\}^\ell$, create a signature on the attribute $\mathbf{x} \in \{0,1\}^\ell$ by running $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \mathbf{x})$. Then, output the secret key as $\mathsf{sk}_{\mathbf{x}} = (\mathbf{x}, \sigma)$.

$\mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathbf{x}}, C, \mathsf{M})$: On input message $\mathsf{M} \in \{0,1\}^\star$ and circuit $C \in \mathcal{C}_\ell$ with an associating topology topo_C proceed as follows:

1. Compute $\mathbf{h} = (h_1, \cdots, h_\ell) \leftarrow G(\mathsf{M}, C)$[8] and create a new circuit $\hat{C} \in \mathcal{C}_\ell$ with two dummy gates connected to each of the input wires of C. Namely, to the input wires $i \in [\ell]$ of C, we add a series composition of two addition gates where one gate adds h_i and the other gate adds $-h_i$; on input x_i to the i-th input wire of $\hat{C}$, it first evaluates to $x_i + h_i$ and then evaluates back to x_i, on which point it gets fed to the i-th (input) wire of C. Here, the value $\mathbf{h}$ is hard-wired into $\hat{C}$, and is considered as one of the internal wires. Further, let N be the number of gates $|\hat{C}|$.
2. Compute the assignment to each non-input wires in $\hat{C}(x_1, \cdots, x_\ell)$: for all $i \in [\ell + 1, \ell + (N-1)]$, compute $(\star_i, i_1, i_2) \leftarrow \mathsf{topo}(i)$ where $\star_i \in \{+, \times\}$, and denote the newly created values $(x_i)_{i=\ell+1}^{\ell+N-1}$ in ascending order as
$$\begin{cases} x_i = x_{i_1} + x_{i_2} & \text{if } \star_i = + \\ x_i = x_{i_1} \cdot x_{i_2} & \text{if } \star_i = \times \end{cases}.$$
3. Create a commitment $(c_\sigma, d_\sigma) \leftarrow \mathsf{C.Com}(\mathsf{pk}_{\mathsf{Com}}, \sigma)$ of the signature σ. Furthermore, for all $i \in [\ell + N - 1]$, create a commitment $(c_i, d_i) \leftarrow \mathsf{C.Com}(\mathsf{pk}_{\mathsf{Com}}, x_i)$ that commits to the value of each wire in $\hat{C}$ (except for the output wire).
4. Generate a NIZK proof π proving that the committed values satisfy relation $\mathcal{R}_{\mathsf{ABS}}$. Concretely, it generates a proof for the following conditions.[9]
   - The attribute $\mathbf{x} = (x_1, \cdots, x_\ell)$ committed to $(c_i)_{i=1}^\ell$ and the signature σ committed to c_σ satisfy the following verification equation:
   $$\mathsf{S.Verify}(\mathsf{vk}_{\mathsf{Sign}}, \mathbf{x}, \sigma) = 1. \tag{7.6}$$
   - For all $i \in [\ell + 1, \ell + N - 1]$, the value x_i committed to c_i satisfy the following equation:
   $$\begin{cases} x_i = x_{i_1} + x_{i_2} & \text{if } \star_i = + \\ x_i = x_{i_1} \cdot x_{i_2} & \text{if } \star_i = \times \end{cases}. \tag{7.7}$$

[7] Here, we do not explicitly define the input and output space of the hash functions, since it may differ according to the underlying NIZK proof system being used.

[8] Here, we assume that we can encode C uniquely into a binary string.

[9] Note that we intentionally dismiss the conditions $(c, d) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}_{\mathsf{Com}}, \star)$ as in the overview, i.e., proving knowledge of a valid opening, since they will be implicitly proven by the fact that the committed messages satisfy Eq. (7.6 - 7.8).

– The values $x_{(\ell+N)_1}$ and $x_{(\ell+N)_2}$ committed to $c_{(\ell+N)_1}$ and $c_{(\ell+N)_2}$, respectively, satisfy the following equation:

$$\begin{cases} 1 = x_{(\ell+N)_1} + x_{(\ell+N)_2} & \text{if } \star_{\ell+N} = + \\ 1 = x_{(\ell+N)_1} \cdot x_{(\ell+N)_2} & \text{if } \star_{\ell+N} = \times \end{cases}. \tag{7.8}$$

5. Finally, output $\Sigma = \big(c_\sigma, (c_i)_{i=1}^{\ell+N-1}, \pi\big)$.

$\mathsf{Verify}(\mathsf{mpk}, \mathsf{M}, C, \Sigma)$: Compute $\mathbf{h} \leftarrow G(\mathsf{M}, C)$ and construct the circuit $\hat{C}$ as in Step 1 of the Sign algorithm. Then, verify the proof with respect to the circuit $\hat{C}$. Output Valid if the proof is verified valid, and output $\mathsf{Invalid}$ otherwise.

Correctness. Observe that $\hat{C}(\mathbf{x}) = C(\mathbf{x})$ for all $\mathsf{M}, \mathbf{x}$. Therefore, the correctness of the scheme follows simply from the correctness of the underlying NIZK proof system. In particular, a signer that has a certified attribute $\mathbf{x}$ such that $C(\mathbf{x}) = 1$ can properly generate a proof proving Eq. (7.6 - 7.8).

7.5.1 Security Analysis

Theorem 7.1 (Privacy)**.** *Assume a statistically hiding commitment scheme with gap-openings and a statistically special HVZK gap-Σ-protocol for relations $(\mathcal{R}_{\mathsf{ABS}}, \mathcal{R}'_{\mathsf{ABS}})$. Then, converting the gap-Σ-protocol into a Fiat-Shamir NIZK proof system, the above attribute-based signature scheme is statistically private in the random oracle model. In case either the hiding property or the special HVZK property only holds computationally, then we obtain computational privacy.*

Proof. For our Fiat-Shamir NIZK proof system, we use the ZK simulator $\mathcal{S}$ that we have defined in Lemma 7.3. Then, privacy of the attribute-based signature scheme follows naturally from the hiding property of the commitment scheme and by the ZK simulator $\mathcal{S}$. In the following, we only consider the case for statistical privacy, i.e., the commitment scheme is statistically hiding and the Fiat-Shamir NIZK proof system is statistically zero-knowledge. It is straightforward to obtain an analogous result for computational privacy. Here, assume $\mathcal{A}$ submits $(\mathsf{M}, \mathbf{x}_0, \mathbf{x}_1, C)$ as the challenge, and let $\hat{C}$ be the circuit created at Step 1 of the Sign algorithm that has N gates. In the actual game, which we denote by $\mathsf{Game}_{\mathsf{real}}$, the challenger picks a random bit $b \leftarrow \{0,1\}$ and returns the signature $\Sigma^* \leftarrow \mathsf{Sign}(\mathsf{mpk}, \mathsf{sk}_{\mathbf{x}_b}, C, \mathsf{M})$, where $\Sigma^* = \big(c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}, \pi^*\big)$ along with the two secret keys $\mathsf{sk}_{\mathbf{x}_0}$ and $\mathsf{sk}_{\mathbf{x}_1}$. Here the proof π^* is the actual zero-knowledge proof created with the witness satisfying $\big(\big(\hat{C}, c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}\big), (\mathbf{x}_b, \sigma, d_\sigma^*, (d_i^*)_{i=1}^{\ell+N-1})\big) \in \mathcal{R}_{\mathsf{ABS}}$ as input, where $\sigma \leftarrow \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \mathbf{x}_b)$ and every commitment/opening pairs are created by running $\mathsf{C.Com}(\mathsf{pk}_{\mathsf{Com}}, \cdot)$. Recall that we omit $\mathsf{vk}_{\mathsf{Sign}}, \mathsf{pk}_{\mathsf{Com}}$ from the statement for simplicity. We consider a game Game_0, where the proof π^* is instead created by running the ZK simulator $\mathcal{S}$ (in particular $\mathcal{S}_2$). Since the statement being proven is in the language $\mathcal{L}_{\mathsf{ABS}}$, by the definition of statistical NIZK proof systems (See Definition 7.9), the proof π^* created in $\mathsf{Game}_{\mathsf{real}}$ and Game_0 are statistically indistinguishable.

Below, we consider changing all the commitments to uniformly random values and simulating a proof for some random (false) statement, at which point the adversary $\mathcal{A}$ will have zero-advantage in winning the privacy game. In order to carry out the proof, we consider $\ell + N - 1$ hybrid games, where in the i-th game Game_i, the challenger swaps the commitment c_i^* with a uniformly random element from the commitment space. Furthermore, to create a proof π^*, the challenger invokes oracle $\hat{\mathcal{S}}_2$ on input the statement $\big(\hat{C}, c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}\big)$. Here recall that $\hat{\mathcal{S}}_2$ is the oracle run by the ZK simulator $\mathcal{S}$, which does not necessarily require the statement to belong in the language $\mathcal{L}_{\mathsf{ABS}}$ to create a simulated proof. Finally, the Game_i challenger outputs a challenge signature $\Sigma^* = \left(c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}, \pi^*\right)$ along with the secret keys $\mathsf{sk}_{\mathbf{x}_0}$ and $\mathsf{sk}_{\mathbf{x}_1}$. Due to the statistically hiding property of the underlying commitment scheme, the view of the adversary in Game_{i-1} and Game_i is negligible.

Finally, in game $\mathsf{Game}_{\ell+N}$, the challenger swaps the commitment c_σ^* with a uniformly random element from the commitment space. Otherwise, he acts exactly the same as the $\mathsf{Game}_{\ell+N-1}$ challenger. Following the same argument above, the differences in the view of the adversary in $\mathsf{Game}_{\ell+N-1}$ and $\mathsf{Game}_{\ell+N}$ is negligible due to the statistically hiding property of the commitment scheme. Furthermore, since all the commitments are now uniformly random over the commitment space in $\mathsf{Game}_{\ell+N}$, the signature $\Sigma^* = \left(c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}, \pi^*\right)$ is completely independent of the attributes $\mathbf{x}_0, \mathbf{x}_1$. Therefore we have that in $\mathsf{Game}_{\ell+N}$, the advantage of adversary $\mathcal{A}$ is 0.

Combining the hybrid games together, we have that the advantage of any adversary $\mathcal{A}$ winning $\mathsf{Game}_{\mathsf{real}}$ is negligible, if the underlying commitment scheme is statistically hiding and the NIZK proof system is statistically zero-knowledge.

□

Theorem 7.2 (Adaptive Unforgeability). *Assume a computationally hiding and a statistically binding commitment scheme with gap openings, a computationally special HVZK gap-Σ_m-protocol*[10] *for relations* $(\mathcal{R}_{\mathsf{ABS}}, \mathcal{R}'_{\mathsf{ABS}})$ *and an* $\mathsf{eu\text{-}cma}$ *secure (deterministic) digital signature scheme. Then, by converting the gap-Σ_m-protocol into a Fiat-Shamir NIZK proof system, the above attribute-based signature scheme is adaptively unforgeable in the random oracle model.*

Proof. Assume there exists a PPT adversary $\mathcal{B}_{\mathsf{ABS}}$ that wins the adaptive unforgeability game with advantage $\epsilon = \epsilon(\lambda)$. Furthermore, let $Q_H = Q_H(\lambda)$ be the number of unique random oracle queries $\mathcal{B}_{\mathsf{ABS}}$ makes to $H(\cdot)$ that is bounded by some polynomial in the security parameter λ. Our proof proceeds in a sequence of games, where X_i denotes the event the adversary wins in Game_i. Our final goal is to construct an adversary $\mathcal{B}_{\mathsf{Sign}}$ that breaks the $\mathsf{eu\text{-}cma}$ security of the underlying digital signature scheme by using $\mathcal{B}_{\mathsf{ABS}}$. As in the proof of Theorem 7.1 for privacy, we use the ZK simulator $\mathcal{S}$ defined in Lemma 7.3 for our Fiat-Shamir NIZK proof system.

$\mathsf{Game}_{\mathsf{real}}$: This game is identical to the real adaptive unforgeability game where all the random oracle queries to $H(\cdot)$ and $G(\cdot)$ are answered randomly by the challenger. At the end of the game, $\mathcal{B}_{\mathsf{ABS}}$ outputs a valid forged signature $(\mathsf{M}^*, C^*, \Sigma^*)$ with probability $\Pr[X_{\mathsf{real}}] = \epsilon$.

Game_1 : In this game, we change the way the challenger answers the random oracle queries to $H(\cdot)$ and the signing queries. Namely, we use the ZK simulator $\mathcal{S}$ associated to the NIZK proof system to answer these. Recall that simulator $\mathcal{S}$ has two modes for running the two oracles $\mathcal{S}_1$ and $\hat{\mathcal{S}}_2$. When $\mathcal{B}_{\mathsf{ABS}}$ submits a random oracle query to $H(\cdot)$, the challenger relays this to oracle $\mathcal{S}_1$ and returns the value outputted by $\mathcal{S}_1$ to $\mathcal{B}_{\mathsf{ABS}}$. Here, the random oracle queries to $G(\cdot)$ are answered by the Game_1 challenger as in the previous game. Furthermore, when $\mathcal{B}_{\mathsf{ABS}}$ submits a signing query on an attribute, message and circuit tuple $(\mathbf{x}, \mathsf{M}, C)$ such that $C(\mathbf{x}) = 1$, it first runs $\mathsf{sk}_{\mathbf{x}} = (\mathbf{x}, \sigma) \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{x})$ and constructs the circuit $\hat{C}$ with N gates using $\mathbf{h} \leftarrow G(\mathsf{M}, C)$ as in Step 1 of the Sign algorithm. Then it proceeds with Step 2 and 3 to create commitments $\left(c_\sigma, (c_i)_{i=1}^{\ell+N-1}\right)$ along with valid openings

[10] Here, recall that we write gap-Σ_m-protocol, when we make explicit of the fact that m valid transcripts are requried for special gap-soundness to hold. Furthermore, this notation also implies that the soundness error is negligible (See Section 7.4.1).

$\big(d_\sigma, (d_i)_{i=1}^{\ell+N-1}\big)$. Finally, it invokes $\hat{\mathcal{S}}_2$ on input the statement $\big(\hat{C}, c_\sigma, (c_i)_{i=1}^{\ell+N-1}\big) \in \mathcal{L}_{\mathsf{ABS}}$[11] and obtains a proof π, and returns the signature $\Sigma = \big(c_\sigma, (c_i)_{i=1}^{\ell+N-1}, \pi\big)$ to $\mathcal{B}_{\mathsf{ABS}}$. Here, the simulated proofs of $\hat{\mathcal{S}}_2$ are distributed negligibly close to the actual proofs in $\mathsf{Game}_{\mathsf{real}}$ by the definition of the NIZK proof system (See Definition 7.9), and the fact that the oracles $\mathcal{S}_2$ and $\hat{\mathcal{S}}_2$ are equivalent in case the statement to be proven is in the language. Hence,

$$|\Pr[X_{\mathsf{real}}] - \Pr[X_1]| = \mathsf{negl}(\lambda).$$

Game_2 : In this game, we change the way the challenger creates the commitment for the signature σ produced during the signing query. In the previous game, when $\mathcal{B}_{\mathsf{ABS}}$ submitted a signing query on an attribute, message and circuit tuple $(\mathbf{x}, \mathsf{M}, C)$ such that $C(\mathbf{x}) = 1$, the challenger created a proper commitment c_σ for the signature σ following Step 3 of the Sign algorithm, i.e., $(c_\sigma, d_\sigma) \leftarrow \mathsf{Com}(\mathsf{pk}_{\mathsf{Com}}, \sigma)$. In this game, however, the Game_2 challenger will instead sample a random value c in the commitment space $\mathcal{C}_{\mathsf{Com}}$ and sets $c_\sigma = c$. Then, as in Game_2, it invokes $\hat{\mathcal{S}}_2$ on input $\big(\hat{C}, c_\sigma, (c_i)_{i=1}^{\ell+N-1}\big)$ and obtains a proof π, and returns the signature $\Sigma = \big(c_\sigma, (c_i)_{i=1}^{\ell+N-1}, \pi\big)$ to $\mathcal{B}_{\mathsf{ABS}}$. Here, recall that oracle $\hat{\mathcal{S}}_2$ is defined to simulate proofs for false statements that are not in the language $\mathcal{L}_{\mathsf{ABS}}$ as well. Now, following the same argument in the previous proof of Theorem 7.1 for privacy, the differences in the view of the adversary in Game_1 and Game_2 are computationally indistinguishable due to the computationally hiding property of the commitment scheme.[12] In other words, we have

$$|\Pr[X_1] - \Pr[X_2]| = \mathsf{negl}(\lambda).$$

Game_3 : In this game, we add an additional winning condition for adversary $\mathcal{B}_{\mathsf{ABS}}$ to satisfy. Namely, when $\mathcal{B}_{\mathsf{ABS}}$ outputs a forgery $(\mathsf{M}^*, C^*, \Sigma^*)$, the Game_3 challenger checks if the random oracle $G(\cdot)$ was ever queried on a message-circuit pair $(\mathsf{M}, C) \neq (\mathsf{M}^*, C^*)$ such that $\hat{C} = \hat{C}^*$. Note that this implies $G(\mathsf{M}, C) = G(\mathsf{M}^*, C^*)$. Hereafter, we say $\mathcal{B}_{\mathsf{ABS}}$ wins if and only if in addition to the winning condition of the previous game, there are no such message-circuit pairs. Since, the output values of the random oracle $G(\cdot)$ are uniformly random over $\{0,1\}^\ell$ for $\ell = \mathsf{poly}(n)$, the probability that a collision occurs for different message-circuit pairs is negligible. Hence,

$$|\Pr[X_2] - \Pr[X_3]| = \mathsf{negl}(\lambda).$$

Below, we denote $\epsilon_3 = \Pr[X_3]$.

In the following, we define the algorithms $\mathcal{A}$ and $\mathcal{O}$ to be used in the forking algorithm $\mathsf{F}_{\mathcal{A},m}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}$ of the generful multi-forking lemma with oracle access (See Lemma 7.1). Looking ahead, the forking algorithm will be used by adversary $\mathcal{B}_{\mathsf{Sign}}$ to win the $\mathsf{eu\text{-}cma}$ security of the underlying digital signature scheme. At a high level, $\mathcal{A}$ will be an algorithm constructed from composing the

[11] Recall we ignore the public parameters $\mathsf{vk}_{\mathsf{Sign}}$ and $\mathsf{pk}_{\mathsf{Com}}$ from the statement for simplicity.

[12] More formally, as in the proof of Theorem 7.1 for privacy, we create q_{sign} hybrid games and swap the commitments of the signature to a random value in the commitment space one hybrid game at a time until we have swapped every signature commitments into the desired random form, where q_{sign} is the number of signature queries $\mathcal{B}_{\mathsf{ABS}}$ makes. Note that q_{sign} is polynomial in the security parameter λ.

Game_3 challenger, $\mathcal{B}_{\mathsf{ABS}}$ and the ZK simulator $\mathcal{S}$ that simulates Game_3, and $\mathcal{O}(\overline{\mathsf{param}}, \cdot)$ will be the signing algorithm $\mathsf{S.Sign}(\mathsf{sk}, \cdot)$ used in the underlying $\mathsf{eu\text{-}cma}$ security game.

To provide the full description of algorithms $\mathcal{A}$ and $\mathcal{O}$, we first define the input generator IG, the set $\mathcal{H}$ and the integer q, which are required to define the inputs for $\mathcal{A}$ and $\mathcal{O}$. First, the input generator IG outputs $(\mathsf{param}, \overline{\mathsf{param}})$ where param constitutes of the verification key $\mathsf{vk}_{\mathsf{Sign}}$, public commitment key $\mathsf{pk}_{\mathsf{Com}}$ and any extra auxiliary parameters required to specify the ABS scheme (e.g., the family of circuits), and $\overline{\mathsf{param}}$ is simply the signing key $\mathsf{sk}_{\mathsf{Sign}}$. Here, $\mathsf{vk}_{\mathsf{Sign}}, \mathsf{sk}_{\mathsf{Sign}}$ and $\mathsf{pk}_{\mathsf{Com}}$ are generated by running $(\mathsf{vk}_{\mathsf{Sign}}, \mathsf{sk}_{\mathsf{Sign}}) \leftarrow \mathsf{S.KeyGen}(1^\lambda, 1^\ell)$ and $\mathsf{pk}_{\mathsf{Com}} \leftarrow \mathsf{C.Gen}(1^\lambda)$. Furthermore, we define the set $\mathcal{H}$ to be the verifier's challenge space $\mathcal{C}_\Sigma$ of the underlying gap-Σ_m-protocol, and set q as Q_H; the number of unique random oracle queries made to $H(\cdot)$ by $\mathcal{B}_{\mathsf{ABS}}$. To summarize, $\mathcal{A}$ will be given param and $h_1, \cdots, h_{Q_H} \in \mathcal{H}$ as input.

We next specify how algorithms $\mathcal{A}$ and $\mathcal{O}$ run. First, the deterministic algorithm $\mathcal{O}$ is simply defined as the signing algorithm of the underlying deterministic digital signature scheme; $\mathcal{O}(\overline{\mathsf{param}}, \cdot) = \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \cdot)$. Here, $\mathcal{O}$ is deterministic since the signing algorithm is deterministic once fixed a signing key $\mathsf{sk}_{\mathsf{Sign}}$. Next, we define $\mathcal{A}$ as the randomized algorithm that simulates Game_3 and outputs a small modification of the forgery returned by $\mathcal{B}_{\mathsf{ABS}}$. We first explain how $\mathcal{A}$ simulates Game_3: $\mathcal{A}$ essentially runs the Game_3 challenger, $\mathcal{B}_{\mathsf{ABS}}$ and the ZK simulator $\mathcal{S}$ internally, with two conceptual changes concerning the Game_3 challenger and the ZK simulator $\mathcal{S}$. In particular the Game_3 challenger is modified to an algorithm which we call the Game'_3 challenger, so that it does not run $(\mathsf{vk}_{\mathsf{Sign}}, \mathsf{sk}_{\mathsf{Sign}}) \leftarrow \mathsf{S.KeyGen}(1^\lambda, 1^\ell)$ anymore. Instead of generating $(\mathsf{vk}_{\mathsf{Sign}}, \mathsf{sk}_{\mathsf{Sign}})$ on its own, the Game'_3 challenger is provided with $\mathsf{vk}_{\mathsf{Sign}}$ by $\mathcal{A}$, and no longer possesses $\mathsf{sk}_{\mathsf{Sign}}$. Whenever the Game'_3 challenger requires to run the signing algorithm $\mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \cdot)$, $\mathcal{A}$ simply invokes $\mathcal{O}(\overline{\mathsf{param}}, \cdot) = \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \cdot)$, which it has oracle access to, and returns whatever outputtd by $\mathcal{O}$ to the Game'_3 challenger. Furthermore, the ZK simulator $\mathcal{S}$ (See Lemma 7.3) is modified in a way so that it does not sample a random value $h_i \leftarrow \mathcal{C}_\Sigma$ when invoked on a random oracle query to $H(\cdot)$. Concretely, on the i-th unique random oracle query to $H(\cdot)$, it simply outputs the value h_i provided by $\mathcal{A}$.[13] This is only a conceptual change, since $\mathcal{C}_\Sigma = \mathcal{H}$ and h_i are sampled uniformly over $\mathcal{H}$. Therefore, the above changes do not alter the view of $\mathcal{B}_{\mathsf{ABS}}$. Hence the advantage of $\mathcal{B}_{\mathsf{ABS}}$ winning the game simulated by $\mathcal{A}$ is exactly the same as of Game_3. Finally, we describe the output of $\mathcal{A}$. In particular, at the end of the simulation of Game_3, $\mathcal{B}_{\mathsf{ABS}}$ outputs a valid forgery $(\mathsf{M}^*, C^*, \Sigma^*)$ where $\Sigma^* = \left(c^*_\sigma, (c^*_i)_{i=1}^{\ell+N-1}, \pi^*\right)$ with probability ϵ_3. In the following let χ^* denote the statement $(\hat{C}^*, c^*_\sigma, (c^*_i)_{i=1}^{\ell+N-1})$, where $\hat{C}^*$ is the circuit with N gates constructed from C^* in Step 1 of the Sign algorithm. Since this is a valid forgery, we must have $\chi^* \in \mathcal{L}_{\mathsf{ABS}}$. Given the forgery of $\mathcal{B}_{\mathsf{ABS}}$, $\mathcal{A}$ first parses the proof π^* as $(\alpha^*, \beta^*, \gamma^*)$, where α^*, β^*, γ^* are the commitment, challenge and response of the underlying gap-Σ_m-protocol (See Definition 7.8), respectively. $\mathcal{A}$ then checks whether $H(\cdot)$ was queried on (χ^*, α^*). If not it outputs $(0, \epsilon_1)$. Otherwise, there exists an index $i^* \in [Q_H]$ for which the challenge $\beta^* = H(\chi^*, \alpha^*)$ is set to h_{i^*}, i.e., $\beta^* = h^{i^*}$. In this case, it outputs $(i^*, (\alpha^*, h_{i^*}, \gamma^*, \chi^*, \mathsf{M}^*, C^*))$. Now, since $\mathcal{A}$ simulates Game_3 perfectly and the probability of $\mathcal{B}_{\mathsf{ABS}}$ outputting a valid forgery without knowledge of the output of $H(\chi^*, \alpha^*)$ (i.e., the challenge) is negligible, we have

$$\begin{aligned} \mathsf{acc} &= \Pr\left[(i^*, (\alpha^*, h_{i^*}, \gamma^*, \chi^*, \mathsf{M}^*, C^*)) \leftarrow \mathcal{A}^{\mathcal{O}(\overline{\mathsf{param}}, \cdot)}(\mathsf{param}, h_1, \cdots, h_{Q_H}) \;:\; i^* \geq 1\right] \\ &\geq \epsilon_3 - \mathsf{negl}(\lambda), \end{aligned} \tag{7.9}$$

where the probability is taken over the choice of $(\mathsf{param}, \overline{\mathsf{param}})$, $(h_i)_{i=1}^{Q_H}$ and the randomness used

[13] More formally, we can think the state st provided to the ZK simulator $\mathcal{S}$ includes $(h_i)_{i=1}^{Q_H}$, assuming without loss of generality that $\mathcal{S}$ knows the bound on the number of query made by $\mathcal{B}_{\mathsf{ABS}}$.

by $\mathcal{A}$.

Finally we construct an adversary $\mathcal{B}_{\mathsf{Sign}}$ against the $\mathsf{eu\text{-}cma}$ security of the underlying digital signature scheme using the forking algorithm $\mathsf{F}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}_{\mathcal{A},m}$. In particular the advantage of $\mathcal{B}_{\mathsf{Sign}}$ will be $\epsilon_3^m/Q_H^{m-1} - \mathsf{negl}(\lambda)$ for a constant m. Hence, assuming the $\mathsf{eu\text{-}cma}$ security of the digital signature scheme, ϵ_3 is negligible. Therefore, since $\epsilon = \epsilon_3 \pm \mathsf{negl}(\lambda)$, we conclude that ϵ is negligible, thus completing the proof. Below, let $\mathcal{C}_{\mathsf{Sign}}$ be the challenger for the $\mathsf{eu\text{-}cma}$ game of the underlying digital signature scheme. Also, let $\mathsf{vk}_{\mathsf{Sign}}$ be the verification key given to $\mathcal{B}_{\mathsf{Sign}}$ and $\mathsf{sk}_{\mathsf{Sign}}$ be the signing key used by $\mathcal{C}_{\mathsf{Sign}}$ to answer the signature queries. In particular, $\mathcal{C}_{\mathsf{Sign}}$ uses the signing algorithm $\mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}},\cdot)$ to answer signature queries made be $\mathcal{B}_{\mathsf{ABS}}$. Now, given $\mathsf{vk}_{\mathsf{Sign}}$, $\mathcal{B}_{\mathsf{Sign}}$ runs $\mathsf{pk}_{\mathsf{Com}} \leftarrow \mathsf{C.Gen}(1^\lambda)$ and prepares param, i.e., the input to $\mathcal{A}$ provided by the input generator IG. This can be done efficiently since param constitutes only of public values: $\mathsf{vk}_{\mathsf{Sign}}, \mathsf{pk}_{\mathsf{Com}}$ and some other public auxiliary parameters specifying the ABS scheme. Since the forking algorithm only requires oracle access to the deterministic algorithm $\mathcal{O}(\overline{\mathsf{param}},\cdot) = \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}},\cdot)$, which is provided by $\mathcal{C}_{\mathsf{Sign}}$, $\mathcal{B}_{\mathsf{Sign}}$ can properly run the forking algorithm $\mathsf{F}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}_{\mathcal{A},m}(\mathsf{param})$ as specified. Note that $\mathsf{param}, \overline{\mathsf{param}}$ are distributed exactly as the output of the input generator IG defined above. Now, due to the general multi-forking lemma with oracle access (Lemma 7.1), we obtain the following pairs with probability frk:

$$\left(1,\ \ \{\alpha^{(k)}, h^{(k)}, \gamma^{(k)}, \chi^{(k)}, \mathsf{M}^{(k)}, C^{(k)}\}_{k\in[m]}\right), \text{ where } \ \chi^{(k)} = \left(\hat{C}^{(k)}, c_\sigma^{(k)}, (c_i^{(k)})_{i=1}^{\ell+N-1}\right)_{k\in[m]}. \tag{7.10}$$

Here, by Eq. (7.1) of Lemma 7.1, we have

$$\mathsf{frk} \geq \mathsf{acc} \cdot \left(\left(\frac{\mathsf{acc}}{Q_H}\right)^{m-1} - \frac{f(m)}{|\mathcal{C}_\Sigma|} \right) = \frac{\mathsf{acc}^m}{Q_H^{m-1}} - \mathsf{negl}(\lambda), \tag{7.11}$$

where $\mathcal{C}_\Sigma$ is the output range of $H(\cdot)$ that is super-polynomially large, m is a constant representing the number of valid transcripts we require to extract a witness and $f(m)$ is a universal positive valued function that only depends on m, i.e., a constant value when viewed as a funtion on the security parameter λ. Now, we argue that for all $k \in [m]$, the values of the commitments $\alpha^{(k)}$ and statements $\chi^{(k)}$ are equivalent, respectively. Let $i^* \in [Q_H]$ be the index outputted by $\mathcal{A}$ in the first run inside the forking algorithm $\mathsf{F}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}_{\mathcal{A},m}(\mathsf{param})$. Then, up until the i^*-th unique random oracle query to $H(\cdot)$, the behavior of $\mathcal{B}_{\mathsf{ABS}}$ is the same for every run, since we fix the randomness being used by the challenger Game_3', $\mathcal{B}_{\mathsf{ABS}}$ and the ZK simulator $\mathcal{S}$. This implies that whatever submitted by $\mathcal{B}_{\mathsf{ABS}}$ on the i^*-th unique random oracle query to $H(\cdot)$, which is the pair $(\alpha^{(k)}, \chi^{(k)})$, must be the same in every run. Let us denote this as $(\alpha^*, \chi^* = (\hat{C}^*, c_\sigma^*, (c_i^*)_{i=1}^{\ell+N-1}))$. Therefore, by running $\mathsf{F}^{\mathcal{O}(\overline{\mathsf{param}},\cdot)}_{\mathcal{A},m}(\mathsf{param})$, $\mathcal{B}_{\mathsf{Sign}}$ obtains m valid transcript of the form $\left(\alpha^*, h^{(k)}, \gamma^{(k)}, \chi^*, \mathsf{M}^*, C^*\right)_{k\in[m]}$ where M^*, C^* are the same in every run as well, due to the winning condition we added in Game_3 and the fact that $\hat{C}^*$ is the same in every run.

Next, we show that $\mathcal{B}_{\mathsf{Sign}}$ can properly extract a witness from the valid transcripts using the knowledge extractor of the underlying gap-Σ_m-protocol (See special gap-soundness of Definition 7.7). Recall that the range of the random oracle $H(\cdot)$ is $\mathcal{C}_\Sigma = \{0, 1, \cdots, m-1\}^t$ for some constant m and an integer-valued function t that is poly-logarithmic in the security parameter λ. Now, by Definition 7.7, in order to extract a witness there needs to exist at least one index $j \in [t]$ such that $\{h_j^{(k)}\}_{k\in[m]} = \{0, 1, \cdots, m-1\}$. Since each $h^{(k)}$ are sampled uniformly random over $\mathcal{C}_H = \{0, 1, \cdots, m-1\}^t$, the probability of no such $j \in [t]$ existing is $(1 - \frac{m!}{m^m})^t$, which is negligible in the security parameter for our choices of m, t. Therefore, with all but negligible probability,

$\mathcal{B}_{\mathsf{Sign}}$ is able to extract a witness $(\mathbf{x}^*, \sigma^*, d^*_\sigma, (d^*_i)_{i=1}^{\ell+N-1})$ in the gap-language $\mathcal{L}'_{\mathsf{ABS}}$ from the m valid transcripts. Furthermore, since we use a statistically binding commitment scheme, the $(\mathbf{x}^*, \sigma^*)$ pair extracted from the transcripts are the actual pairs used by $\mathcal{B}_{\mathsf{ABS}}$ to create a forgery, with all but negligible probability.

Finally, we show that $(\mathbf{x}^*, \sigma^*)$ is a valid signature forgery that allows $\mathcal{B}_{\mathsf{Sign}}$ to win the $\mathsf{eu\text{-}cma}$ game between the challenger $\mathcal{C}_{\mathsf{Sign}}$. Namely, we show that $\mathbf{x}^*$ was never queried as the key reveal query by $\mathcal{B}_{\mathsf{ABS}}$ in all of the m runs of $\mathcal{A}$. Note that the only situation $\mathcal{A}$ invokes the signing oracle $\mathcal{O}(\overline{\mathsf{param}}, \cdot) = \mathsf{S.Sign}(\mathsf{sk}_{\mathsf{Sign}}, \cdot)$ is when $\mathcal{B}_{\mathsf{ABS}}$ submits a key reveal query to the Game'_3 challenger. This is because we altered the game in Game_2 so that the ZK simulator is used to answer the signing queries made by $\mathcal{B}_{\mathsf{ABS}}$. Now, since $\mathcal{B}_{\mathsf{ABS}}$ outputs a valid forgery we have $\hat{C}^*(\mathbf{x}^*) = 1$. Then, by the way we construct $\hat{C}^*(\mathbf{x}^*)$ in Step 1 of the Sign algorithm, we have $C(\mathbf{x}^*) = 1$ as well. On the other hand, due to the winning condition of $\mathcal{B}_{\mathsf{ABS}}$, $\mathcal{B}_{\mathsf{ABS}}$ must have never made a key reveal query on $\mathbf{x}^*$ such that $C^*(\mathbf{x}^*) = 1$ (in any of the runs). Therefore, we conclude that $\mathbf{x}^*$ was never queried to the Game'_3 challenger by $\mathcal{B}_{\mathsf{ABS}}$ in any of the runs of $\mathcal{A}$; $(\mathbf{x}^*, \sigma^*)$ is a valid forgery.

Hence, combining Eq. (7.9), (7.11) and the previous games together, assuming a PPT adversary $\mathcal{B}_{\mathsf{ABS}}$ that makes at most Q_H queries to the random oracle $H(\cdot)$ and wins the adaptive unforgeability game with advantage ϵ, there exists a PPT adversary $\mathcal{B}_{\mathsf{Sign}}$ that wins the $\mathsf{eu\text{-}cma}$ security with advantage $\epsilon^m / Q_H^{m-1} - \mathsf{negl}(\lambda)$ for a constant m.

□

7.5.2 Implications

Since a computationally hiding and statistically binding commitment scheme, a deterministic digital signature scheme and a computationally special HVZK Σ-protocols for any NP-language are all implied from one-way functions (See for example [Nao91, Rom90, PSV06]), we obtain the following lemma as an implication of our above result:

Lemma 7.4. *If one-way functions exist, then there exist computationally private and adaptive unforgeable attribute-based signature schemes for unbounded circuits in the random oracle model.*

7.6 ABS for Unbounded Circuits from Lattices

In this section, we provide an efficient instantiation of our generic ABS construction for unbounded circuits from lattices. In particular, we prepare a lattice-based signature scheme and a commitment scheme with gap-openings, and construct an associating lattice-based gap-Σ-protocol for the relation $\mathcal{R}_{\mathsf{ABS}}$. We believe our gap-Σ-protocol for proving possession of a valid signature, which departs from the previously known stern-type protocol of [LNSW13], to have applications in other contexts such as group signatures. Finally, we will be using the $\omega(\cdot)$-notation throughout the rest of this section. Note that $\omega(f(X))$ denotes any function that grows asymptotically faster than $f(X)$. For instance, when we state that the communication is $\omega(f(X))$, it can be set as small as $f(X) \cdot \log X$.

7.6.1 Preparing Tools

We present the underlying lattice-based digital signature scheme and commitment scheme with gap-openings that we use as building blocks for our lattice-based ABS scheme.

Digital Signature Scheme. Here, we review the lattice-based digital signature scheme of Boyen [Boy10] with an improved security reduction by [MP12]. This scheme is a lattice-based analogue of the Waters' pairing-based signature [Wat05]. Below, we provide a deterministic version of Boyen's signature scheme, where the signing algorithm uses a PRF for generating the required randomness. In the following, by lattice convention, we use the dimension of the lattice n to denote the security parameter.

Theorem 7.3. *Let n, m, q be positive integers such that $m \geq 2n \log q$. Let α, β be positive reals such that $\alpha = \Omega(\sqrt{\ell n \log q} \log n)$ and $\beta = \alpha\omega(\sqrt{\log m})$. Then, the following algorithms* $(\mathsf{S.KeyGen}, \mathsf{S.Sign}, \mathsf{S.Verify})$ *form a deterministic digital signature scheme with message space $\mathcal{M} = \{0,1\}^\ell$ that is* eu-cma *secure under hardness of the* $\mathsf{SIS}^\infty_{n,m,q,\ell\tilde{O}(n)}$ *problem.*

$\mathsf{S.KeyGen}(1^n, 1^\ell)$: *It samples a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ with a trapdoor $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ using algorithm* $\mathsf{TrapGen}(1^n, 1^m, q)$. *It also samples matrices $\mathbf{A}_i \leftarrow \mathbb{Z}_q^{n\times m}$ for $i \in [0, \ell]$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$ and generates a seed for a PRF by running $r \leftarrow \mathsf{PRF.Gen}(1^n)$. Finally it outputs the verification key vk and signing key sk as*

$$\mathsf{vk} = (\mathbf{A}, \mathbf{A}_0, \cdots, \mathbf{A}_\ell, \mathbf{u}), \quad \mathsf{sk} = (\mathbf{T_A}, r).$$

$\mathsf{S.Sign}(\mathsf{sk}, \mathbf{x})$: *On input the message $\mathbf{x} \in \{0,1\}^\ell$, it first constructs the matrix $\mathbf{A_x} = \mathbf{A}_0 + \sum_{i=1}^{\ell} x_i \mathbf{A}_i \in \mathbb{Z}_q^{n\times m}$, where x_i is the i-th bit of $\mathbf{x}$. Then using $\mathbf{T_A}$, it samples a short vector $\mathbf{z} \in \mathbb{Z}^{2m}$ such that $[\mathbf{A}|\mathbf{A_x}]\mathbf{z} = \mathbf{u} \mod q$ using algorithm* $\mathsf{SampleLeft}(\mathbf{A}, \mathbf{A_x}, \mathbf{u}, \mathbf{T_A}, \alpha)$, *where the output of* $\mathsf{PRF.Eval}(r, \mathbf{x})$ *is used as the randomness. Finally, it outputs $\sigma = \mathbf{z}$ as the signature.*

$\mathsf{S.Verify}(\mathsf{vk}, \mathbf{x}, \sigma)$: *It first checks that $\mathbf{x} \in \{0,1\}^\ell$. Next, it checks whether $[\mathbf{A}|\mathbf{A_x}]\mathbf{z} = \mathbf{u} \mod q$ and $\|\mathbf{z}\|_\infty \leq \beta$. It outputs 1 if all the above check passes, otherwise it outputs 0.*

Commitment Scheme. Here, we present the commitment scheme of [XXW13] with minor modification. Specifically, we slightly deviate from their construction to be consistent with our notion of commitment schemes with gap openings from Section 7.3.1. Furthermore, for simplicity, we present the commitment scheme based on standard lattices, whereas [XXW13] uses ring lattices. Finally, we use Lemma 4 of [LLNW14] instead of Lemma 1 of [XXW13] to optimize the required parameters of the commitment scheme. In the following let $[\cdot||\cdot]$ denote the vertical concatenation of vectors.

Theorem 7.4. *Let $n, \bar{m}, q$ be positive integers such that $\bar{m} \geq 3n$, q a prime. Further, let γ, γ' be positive reals such that $q \geq (4\gamma + 1)^2$ and $\gamma \geq \gamma'\omega(\log n)$. Then, the following algorithms* $(\mathsf{C.Gen}, \mathsf{C.Com}, \mathsf{C.Open})$ *form a computationally hiding and statistically binding commitment scheme with gap openings under the hardness of the* $\mathsf{LWE}_{n,\bar{m},q,D_{\mathbb{Z},\gamma}}$ *problem. Here the message space $\mathcal{M}$ is $\mathbb{Z}_q$ and the commitment space $\mathcal{C}$ is $\mathbb{Z}_q^{\bar{m}}$.*

$\mathsf{C.Gen}(1^n)$: *It samples $\mathbf{B} \leftarrow \mathbb{Z}_q^{(n+1)\times\bar{m}}$ and outputs* $\mathsf{pk} = \mathbf{B}$.

$\mathsf{C.Com}(\mathsf{pk}, \mathsf{M})$: *For a message $\mathsf{M} \in \mathbb{Z}_q$, it samples a random vector $\mathbf{s} \leftarrow \mathbb{Z}_q^n$. Then, it samples $\mathbf{e} \leftarrow D_{\mathbb{Z}^{\bar{m}},\gamma'}$ until $\|\mathbf{e}\|_\infty \leq \gamma$ holds.*[14] *Finally, it outputs $(c, d) = (\mathbf{B}^\top[\mathbf{s}||\mathsf{M}] + \mathbf{e} \mod q, (\mathbf{s}, \mathbf{e}))$.*

[14] For our parameter selection, this procedure will end in a constant number of trials with all but negligible probability.

$\mathsf{C.Open}(\mathsf{pk}, \mathsf{M}, c, d)$: *It first checks if* $\mathsf{M} \in \mathbb{Z}_q$. *It then parses* $d = (\mathbf{s}, \mathbf{e})$ *and checks if* $c = \mathbf{B}^\top[\mathsf{s}||\mathsf{M}] + \mathbf{e} \mod q$ *and* $\|\mathbf{e}\|_\infty \leq 2\gamma$ *hold. If all the check passes it outputs 1, otherwise it outputs 0.*

Observe that the above commitment scheme has gap-openings; although the commitment algorithm $\mathsf{C.Com}$ only samples vectors $\mathbf{e}$ such that $\|\mathbf{e}\|_\infty \leq \gamma$, the opening algorithm $\mathsf{C.Open}$ accepts $\mathbf{e}$ such that $\gamma < \|\mathbf{e}\|_\infty \leq 2\gamma$ as well. In addition, we can easily check membership of an element $(c, d = (\mathbf{s}, \mathbf{e}))$ in $\mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})$ by checking whether the opening algorithm outputs 1 and $\|\mathbf{e}\|_\infty \leq \gamma$ holds.

[XXW13] provides three gap-Σ-protocols for proving useful relations over committed values: Σ_{Open} for proving knowledge of a valid opening and Σ_{Add}, Σ_{Mult}[15] for proving arithmetic relations (over $\mathbb{Z}_q$) of committed values. We additionally construct one useful gap-Σ-protocol $\Sigma_{\mathsf{EqTo}\star}$ for proving that a commitment opens to a specific value. So as not to interrupt the main objective of this section, we refer the details of the construction to Section 7.7. Then, the above commitment scheme is equipped with the following four *basic* gap-Σ-protocols.

Theorem 7.5. *The commitment scheme with gap openings in Theorem 7.4 has associating computationally special HVZK gap-Σ-protocols* $(\Sigma_{\mathsf{Open}}, \Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}, \Sigma_{\mathsf{Mult}})$ *for the following four relations:*

$$\begin{aligned}
\mathcal{R}_{\mathsf{Open}} &= \{(\mathsf{pk}, c), (\mathsf{M}, d) \mid (c, d) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})\},\\
\mathcal{R}_{\mathsf{EqTo}\star} &= \{(\mathsf{pk}, c, \mathsf{M}), d \mid (c, d) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M})\},\\
\mathcal{R}_{\mathsf{Add}} &= \{(\mathsf{pk}, (c_i)_{i=1}^3), ((\mathsf{M}_i, d_i)_{i=1}^3) \mid \mathsf{M}_3 = \mathsf{M}_1 + \mathsf{M}_2 \ \wedge \ (c_i, d_i) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M}_i) \ \textit{for } i \in [3]\},\\
\mathcal{R}_{\mathsf{Mult}} &= \{(\mathsf{pk}, (c_i)_{i=1}^3), ((\mathsf{M}_i, d_i)_{i=1}^3) \mid \mathsf{M}_3 = \mathsf{M}_1 \cdot \mathsf{M}_2 \ \wedge \ (c_i, d_i) \in \mathcal{D}_{\mathsf{Com}}(\mathsf{pk}, \mathsf{M}_i) \ \textit{for } i \in [3]\}.
\end{aligned}$$

The gap-relations $(\Sigma'_{\mathsf{Open}}, \Sigma'_{\mathsf{EqTo}\star}, \Sigma'_{\mathsf{Add}}, \Sigma'_{\mathsf{Mult}})$ *are defined similarly except that the set* $\mathcal{D}_{\mathsf{G\text{-}Com}}$ *is used instead of* $\mathcal{D}_{\mathsf{Com}}$.

The above gap-Σ-protocols of [XXW13] additionally require internally a standard commitment scheme, which is used by the prover in the first round to send a commitment to the verifier. Although, we can use the commitment scheme of [XXW13] provided above, we use the more efficient lattice-based commitment scheme of Kawachi et al. [KTX08] to instantiate the gap-Σ-protocols. In this case, the communication costs of $\Sigma_{\mathsf{Open}}, \Sigma_{\mathsf{EqTo}\star}$ are $\omega(\bar{m}\log q \log\gamma \log n)$ and $\Sigma_{\mathsf{Add}}, \Sigma_{\mathsf{Mult}}$ are $\omega(\bar{m}\log^3 q \log\gamma \log n)$. Plugging in some example parameters required by the commitment scheme, e.g., $\bar{m} = 3n$, the communication costs can be set to be $\tilde{O}(n)$, where recall n is the security parameter.

Remark 7.1. *The above four basic gap-Σ-protocols can be composed in parallel to obtain a gap-Σ-protocol for larger relations, e.g., provided with commitments* $(c_i)_{i=1}^4$ *of the values* $(\mathsf{M}_i)_{i=1}^4$ *satisfying* $\mathsf{M}_4 = \sum_{i=1}^3 \mathsf{M}_i$*, we can prove this relation by creating one extra auxiliary commitment* c_{aux} *for* $\mathsf{M}_{\mathsf{aux}} = \mathsf{M}_1 + \mathsf{M}_2$ *and running two* Σ_{Add} *in parallel for the statement pairs* $(\mathsf{pk}, c_1, c_2, c_{\mathsf{aux}})$ *and* $(\mathsf{pk}, c_{\mathsf{aux}}, c_3, c_4)$.

7.6.2 ABS for Unbounded Circuits Based on Lattices

To instantiate the generic ABS construction in Section 7.5 from lattices, it is sufficient to prove that the above digital signature scheme and commitment scheme are equipped with a gap-Σ-protocol for the relation $\mathcal{R}_{\mathsf{ABS}}$. Therefore, below we aim at constructing a gap-Σ protocol for

[15] In their paper, they present two protocols for proving arithmetic relations, however, in our work we only consider the more efficient protocol in [XXW13], Section 4.3.

proving Eq. (7.6), (7.7) and (7.8) in our ABS construction, where the attribute $\mathbf{x}$ and Boyen signatures σ are committed using the commitment scheme of [XXW13]. Below, we provide the equations that appear in our ABS construction for reference.

- The attribute $\mathbf{x} = (x_1, \cdots, x_\ell)$ committed to $(c_i)_{i=1}^{\ell}$ and the signature σ committed to c_σ satisfy the following verification equation:

$$\mathsf{S.Verify}(\mathsf{vk}_{\mathsf{Sign}}, \mathbf{x}, \sigma) = 1. \tag{7.6}$$

- For all $i \in [\ell + 1, \ell + N - 1]$, the value x_i committed to c_i satisfy the following equation:

$$\begin{cases} x_i = x_{i_1} + x_{i_2} & \text{if } \star_i = + \\ x_i = x_{i_1} \cdot x_{i_2} & \text{if } \star_i = \times \end{cases}. \tag{7.7}$$

- The values $x_{(\ell+N)_1}$ and $x_{(\ell+N)_2}$ committed to $c_{(\ell+N)_1}$ and $c_{(\ell+N)_2}$, respectively, satisfy the following equation:

$$\begin{cases} 1 = x_{(\ell+N)_1} + x_{(\ell+N)_2} & \text{if } \star_{\ell+N} = + \\ 1 = x_{(\ell+N)_1} \cdot x_{(\ell+N)_2} & \text{if } \star_{\ell+N} = \times \end{cases}. \tag{7.8}$$

Taking the above Remark 7.1 into consideration, a gap-Σ-protocol for proving Eq. (7.7) and (7.8), which are essentially proving that the circuit is computed correctly, can be constructed by simply composing the basic gap-Σ-protocols $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}, \Sigma_{\mathsf{Mult}}$ in parallel. In more detail, we use Σ_{Add} and Σ_{Mult} to prove that we computed each gates correctly, and use $\Sigma_{\mathsf{EqTo}\star}$ to prove that the value associated to the output wire is equal to 1. Therefore, in the following, we only focus on how to construct a gap-Σ-protocol for proving Eq. (7.6); we construct a gap-Σ-protocol for proving possession of a valid Boyen-signature using $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}, \Sigma_{\mathsf{Mult}}$. Here, we stress that we cannot simply use the gap-Σ-protocol for proving possession of a valid Boyen-signature of [LNSW13] for our purpose, since their protocol does not allow us to efficiently prove possession of messages satisfying complex arithmetic relations.[16] In other words, since Eq. (7.6) and (7.7) share the same witness $\mathbf{x} = (x_1, \cdots, x_\ell)$, we will not be able to combine the different types of gap-Σ-protocols of [LNSW13] and [XXW13] to construct a gap-Σ protocol for the relation $\mathcal{R}_{\mathsf{ABS}}$.

To summarize, our goal is to construct a gap-Σ-protocol for proving possession of a valid Boyen signature $\sigma = \mathbf{z} = [z_1, \cdots, z_{2m}]^\top \in \mathbb{Z}^{2m}$, where $\mathbf{x} = (x_1, \cdots, x_\ell) \in \{0,1\}^\ell$ is viewed as the message, provided the verification key $\mathsf{vk}_{\mathsf{Sign}}$ and the commitments to the signature σ and message $\mathbf{x}$. Then, since the basic gap-Σ-protocols of Theorem 7.4 allows for parallel composition, our desired gap-Σ-protocol for the relation $\mathcal{R}_{\mathsf{ABS}}$ is obtained by composing the gap-Σ protocol for the Boyen signature with the gap-Σ-protocols for Eq. (7.7) and (7.8) together. Below, we assume the commitment c_σ of the signature is provided in the form $(\bar{c}_k)_{k \in [2m]}$ where each $\bar{c}_i$ is a commitment of the k-th element $z_k \in \mathbb{Z}$ of $\mathbf{z}$ (viewed as an element in $\mathbb{Z}_q$), and the commitment of the message $c_{\mathbf{x}}$ is provided in the form $(c_i)_{i \in [\ell]}$ where each c_i is a commitment of the value $x_i \in \{0,1\}$. Now, due to the verification algorithm of the Boyen signature scheme, proving a signature is valid is equivalent to proving the following three statements:

$$\mathbf{x} \in \{0,1\}^\ell \iff x_i \in \{0,1\} \text{ for } i \in [\ell], \tag{7.12}$$

[16] The subsequent works of [LLM+16, YAL+17] allow proving possession of a valid Boyen-signature while also proving possession of messages satisfying some simple arithmetic relations. However, their protocols are not strong enough to prove arbitrary circuits in zero-knowledge.

$$\|\mathbf{z}\|_\infty \leq \beta \quad \Longleftrightarrow \quad |z_k| \leq \beta \text{ for } k \in [2m], \tag{7.13}$$

$$\Big[\mathbf{A} | \mathbf{A}_0 + \sum_{i=1}^{\ell} x_i \mathbf{A}_i\Big] \mathbf{z} = \mathbf{u} \mod q, . \tag{7.14}$$

Below we construct gap-Σ-protocols respectively for the above equations by converting each of them into an arithmetic circuit, and using the basic gap-Σ-protocols provided in Theorem 7.4 as building blocks to prove the satisfiability of each circuit.

Gap-Σ-Protocol for Proving Eq. (7.12). It is sufficient to prove that for every $i \in [\ell]$, the commitment $c_i \leftarrow \mathsf{C.Com}(\mathsf{pk}, x_i)$ opens to either 0 or 1. To do so, we first create auxiliary commitments $c_{\mathsf{zero}} \leftarrow \mathsf{C.Com}(\mathsf{pk}, 0)$ and $g_i \leftarrow \mathsf{C.Com}(\mathsf{pk}, x_i^2)$ for $i \in [\ell]$. Then using the commitments $(c_i)_{i\in[\ell]}$ and the auxiliary commitments, and combining the basic gap-Σ-protocols $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}$ and Σ_{Mult} together, we construct a gap-Σ-protocol for proving the following statement for all $i \in [\ell]$:

$$c_{\mathsf{zero}} \text{ opens to } 0 \quad \wedge \quad x_i^2 = x_i \cdot x_i \quad \wedge \quad 0 = x_i^2 - x_i$$

Since all arithmetic operations are over the finite field $\mathbb{Z}_q$, the only x_i that satisfy the above relations are $x_i = 0$ or 1. Therefore, the above gap-Σ-protocol indeed proves Eq. (7.12). The total communication cost is $\omega(\ell \bar{m} \log^3 q \log \gamma \log n)$. For example, the parameters can be set to be $\tilde{O}(\ell n)$.

Gap-Σ-Protocol for Proving Eq. (7.13). Here, for simplicity of the protocol, we assume that β can be written as $2^\zeta - 1$ for some positive integer ζ. Equivalently, $\zeta = \log(\beta + 1)$. This does not harm the efficiency nor the security of the signature scheme by much, since given any β, there always exists a value of the form $2^\zeta - 1$ in between β and 2β.

First, we prepare some notations. For $k \in [2m]$, let $z_{k,j}$ be the j-th bit of the binary representation of $z_k \in \mathbb{Z}$ for $j \in [\zeta]$. Note that, we extend the standard binary decomposition to negative integers as well in the obvious way. In particular, we can bit decompose any $z_k \in [-\beta, \beta]$ as $z_k = \sum_{j=1}^{\zeta} 2^{j-1} z_{k,j}$, where $z_{k,j} \in \{-1, 0, 1\}$.[17] Further, set $w_{k,j} = 2^{j-1} z_{k,j}$ for $j \in [\zeta]$ and $w_{k,[j']} = \sum_{j=1}^{j'} w_{k,j}$ for $j' \in [2, \zeta]$. Finally, define $w_{k,[1]} = w_{k,1}$. Next, create the following auxiliary commitments for $k \in [2m]$: $c_{\mathsf{zero}} \leftarrow \mathsf{C.Com}(\mathsf{pk}, 0)$, $c_{\mathsf{coeff},j} \leftarrow \mathsf{C.Com}(\mathsf{pk}, 2^{j-1})$, $\bar{c}_{k,j,\mu} \leftarrow \mathsf{C.Com}(\mathsf{pk}, z_{k,j}^{\mu})$, $h_{k,j} \leftarrow \mathsf{C.Com}(\mathsf{pk}, w_{k,j})$ for $\mu \in [3]$, $j \in [\zeta]$, and $h_{k,[j']} \leftarrow \mathsf{C.Com}(\mathsf{pk}, w_{k,[j']})$ for $j' \in [2, \zeta]$. Then, using the commitments $(\bar{c}_k)_{k\in[2m]}$, the auxiliary commitments and composing the gap-Σ-protocols $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}$ and Σ_{Mult} together, we construct a gap-Σ-protocol for the following statement for all $k \in [2m], j \in [\zeta]$ and $j' \in [2, \zeta]$:[18]

$$\begin{aligned} &c_{\mathsf{zero}} \text{ opens to } 0 \quad \wedge \quad c_{\mathsf{coeff},j} \text{ opens to } 2^j \quad \wedge \quad z_{k,j}^2 = z_{k,j} \cdot z_{k,j} \quad \wedge \quad z_{k,j}^3 = z_{k,j}^2 \cdot z_{k,j} \quad \wedge \\ &0 = z_{k,j}^3 - z_{k,j} \quad \wedge \quad w_{k,j} = 2^{j-1} \cdot z_{k,j} \quad \wedge \quad w_{k,[j']} = w_{k,j'} + w_{k,[j'-1]} \quad \wedge \quad 0 = z_k - w_{k,[\zeta]}. \end{aligned}$$

We check that the above statement is equivalent to Eq. (7.13), i.e., each z_k satisfy $|z_k| \leq \beta$ for all $k \in [2m]$. First, since q is a prime, the only $z_{k,j}$ satisfying $z_{k,j}^3 - z_{k,j} = 0$ over $\mathbb{Z}_q$ are $-1, 0, 1$. Hence, the above statement proves that $z_{k,j} \in \{-1, 0, 1\}$. Furthermore, when $z_{k,j} \in \{-1, 0, 1\}$, we

[17] A subtly is that unlike standard bit decomposition, the bit representation is not unique anymore, e.g., 11 can be decomposed as $(1, 1, 0, 1)$ or $(-1, 0, 1, 1)$. However, this will not affect our following argument.

[18] Since we prove c_{zero} opens to 0 in the above gap-Σ-protocol for proving Eq. (7.12), we will not require this when we compose the gap-Σ-protocols together. The same holds for the aforementioned gap-Σ-protocol for proving Eq. (7.14).

have $|z_k| \leq \sum_{j=1}^{\zeta} 2^{j-1} |z_{k,j}| \leq 2^{\zeta-1} = \beta$. Therefore, if the above statement holds, then we must have $|z_k| \leq \beta$ for all $k \in [2m]$. The total communication cost is $\omega(m\bar{m}\log\beta\log^3 q\log\gamma\log n)$. For example, the parameters can be set to be $\tilde{O}(n^2)$.

Gap-Σ-Protocol for Proving Eq. (7.14). We first prepare some notations. Let $a_{s,k}$ (resp., $a_{i,s,k}$) denote the (s, k_1)-th (resp., $(s, k_2 - m)$-th) entry of $\mathbf{A}$ (resp., $\mathbf{A}_i$) $\in \mathbb{Z}_q^{n\times m}$, for $s \in [n]$, $k_1 \in [m]$ (resp., $k_2 \in [m+1, 2m]$) and $i \in [0, \ell]$. Then, observe that we can rewrite Eq. (7.14) using the following equations for $s \in [n]$:

$$\sum_{k_1=1}^{m} a_{s,k_1} \cdot z_{k_1} + \sum_{k_2=m+1}^{2m} \left(a_{0,s,k_2} + \sum_{i=1}^{\ell} x_i \cdot a_{i,s,k_2} \right) \cdot z_{k_2} = u_s \tag{7.15}$$

Next, we prepare some auxiliary values for $s \in [n]$ in order to prove the above equations using the gap-Σ-protocols $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}$ and Σ_{Mult}: $w_{i,s,k_2} = x_i \cdot a_{i,s,k_2}$, $w_{[i'],s,k_2} = \sum_{i=1}^{i'} w_{i,s,k_2}$, $a_{s,k_2} = a_{0,s,k_2} + w_{[\ell],s,k_2}$ for $i \in [\ell]$, $i' \in [2,\ell]$, $k_2 \in [m+1, 2m]$, $b_{s,k} = a_{s,k} \cdot z_k$ for $k \in [2m]$, $b_{s,[k']} = \sum_{k_1=1}^{k'} b_{s,k_1}$ for $k' \in [2, m]$, $b_{s,[k']} = \sum_{k_2=m+1}^{k'} b_{s,k_2}$ for $k' \in [m+2, 2m]$ and $t_s = b_{s,[m]} + b_{s,[2m]}$. Further define $w_{[1],s,k_2} = w_{1,s,k_2}$, $b_{s,[1]} = b_{s,1}$ and $b_{s,[m+1]} = b_{s,m+1}$. Next, we create auxiliary commitments for the related values for $s \in [n]$: $c_{\mathsf{mat},s,k_1} \leftarrow \mathsf{C.Com}(\mathsf{pk}, a_{s,k_1})$, $c_{\mathsf{mat},i,s,k_2} \leftarrow \mathsf{C.Com}(\mathsf{pk}, a_{i,s,k_2})$ for $i \in [0,\ell]$, $k_1 \in [m]$, $k_2 \in [m+1, 2m]$, $\omega_{i,s,k_2} \leftarrow \mathsf{C.Com}(\mathsf{pk}, w_{i,s,k_2})$, $\omega_{[i'],s,k_2} \leftarrow \mathsf{C.Com}(\mathsf{pk}, w_{[i'],s,k_2})$, $\alpha_{s,k_2} \leftarrow \mathsf{C.Com}(\mathsf{pk}, a_{s,k_2})$ for $i \in [\ell]$, $i' \in [2,\ell]$, $k_2 \in [m+1, 2m]$, $\beta_{s,k} \leftarrow \mathsf{C.Com}(\mathsf{pk}, b_{s,k})$ for $k \in [2m]$, $\beta_{s,[k']} \leftarrow \mathsf{C.Com}(\mathsf{pk}, b_{s,[k']})$ for $k' \in [2, m] \cup [m+2, 2m]$. Then, using the commitment $(c_i)_{i=1}^{\ell}$, $(\bar{c}_k)_{k\in[2m]}$, the auxiliary commitments and composing the gap-Σ-protocols $\Sigma_{\mathsf{EqTo}\star}, \Sigma_{\mathsf{Add}}$ and Σ_{Mult} together, we construct a gap-Σ-protocol for the following statement for all $s \in [n]$, $i \in [\ell]$, $i' \in [2,\ell]$, $k_1 \in [m]$, $k_2 \in [m+1, 2m]$, $k \in [2m]$, $k' \in [2, m] \cup [m+2, 2m]$:

$$\begin{aligned}
&c_{\mathsf{zero}} \text{ opens to } 0 \quad \wedge \quad c_{\mathsf{mat},s,k_1}, c_{\mathsf{mat},0,s,k_2}, c_{\mathsf{mat},i,s,k_2} \text{ opens to } a_{s,k}, a_{0,s,k}, a_{i,s,k}, \text{ respectively} \quad \wedge \\
&w_{i,s,k_2} = x_i \cdot a_{i,s,k_2} \quad \wedge \quad w_{[i'],s,k_2} = w_{i',s,k_2} + w_{[i'-1],s,k_2} \quad \wedge \quad a_{s,k_2} = a_{0,s,k_2} + w_{[\ell],s,k_2} \quad \wedge \\
&b_{s,k} = a_{s,k} \cdot z_k \quad \wedge \quad b_{s,[k']} = b_{s,k'} + b_{s,[k'-1]} \quad \wedge \quad t_s = b_{s,[m]} + b_{s,[2m]} \quad \wedge \quad 0 = u_s - t_s
\end{aligned}$$

The above statement can be checked that it is equivalent to proving Eq. (7.15) for $s \in [n]$. The total communication cost is $\omega(\ell n m\bar{m}\log^3 q\log\gamma\log n)$. For example, the parameters can be set to be $\tilde{O}(\ell n^3)$.

Gap-Σ-Protocol for $\mathcal{R}_{\mathsf{ABS}}$. To summarize, we obtain a gap-Σ-protocol for proving possession of a valid Boyen signature by composing the gap-Σ-protocols for proving Eq. (7.12-7.14) together. Then, by composing this protocol with the aforementioned gap-Σ-protocols for proving Eq. (7.7) and (7.8), we obtain our desired gap-Σ-protocol for the relation $\mathcal{R}_{\mathsf{ABS}}$ where the total communication cost is $\omega((m(\ell n + \log\beta) + |C|)\bar{m}\log^3 q\log\gamma\log n)$. Here, $|C|$ is size of the circuit (i.e., policy) associated to the message. For example, the parameters can be set to be $\tilde{O}((\ell n + |C|)n^2)$. Thus, we obtain our lattice-based ABS scheme for *unbounded circuits* in the random oracle model by instantiating the generic ABS construction in Section 7.5 with our gap-Σ protocol for $\mathcal{R}_{\mathsf{ABS}}$.

7.7 Gap-Σ-Protocol for the Relation $\mathcal{R}_{\mathsf{EqTo}\star}$

In this section, we show how to construct a gap-Σ protocol for relation $\mathcal{R}_{\mathsf{EqTo}\star}$ building on top of the commitment scheme of [XXW13]. Observe that due to Lemma 7.2, we only require to construct a gap-$\Sigma_{m,1}$-protocol for relations $(\mathcal{R}_{\mathsf{EqTo}\star}, \mathcal{R}'_{\mathsf{EqTo}\star})$ for some m. Let the commitment

c be $\mathbf{B}^\top[\mathbf{s}\|\mathsf{M}] + \mathbf{e} \in \mathbb{Z}_q^{\bar{m}}$, where $\|\mathbf{e}\|_\infty \leq \gamma$. The goal of the protocol is for the prover $\mathcal{P}$ to convince the verifier $\mathcal{V}$ that c is a valid commitment of M without leaking any other information. Before stating the gap-Σ-protocol, $\mathcal{P}$ first bit decomposes $\mathbf{e} \in \mathbb{Z}^{\bar{m}}$ to $k = \lfloor \log \gamma \rfloor + 1$ vectors $\tilde{\mathbf{e}}_i \in \{-1, 0, 1\}^{\bar{m}}$ such that $\mathbf{e} = \sum_{i=0}^{k-1} 2^i \tilde{\mathbf{e}}_i$. Then, $\mathcal{P}$ appends to each vector $\tilde{\mathbf{e}}_i$ an arbitrary vector $\bar{\mathbf{e}}_i$ in $\{-1, 0, 1\}^{2\bar{m}}$ such that the number of -1, 0, 1 in the vector $[\tilde{\mathbf{e}}_i\|\bar{\mathbf{e}}_i]$ are respectively $\bar{m}$. Denote $\mathcal{B}_{3\bar{m}}$ as the set of vectors where the number of $-1, 0, 1$ are exactly $\bar{m}$, and $\mathbf{e}_i$ as the vector $[\tilde{\mathbf{e}}_i\|\bar{\mathbf{e}}_i] \in \mathcal{B}_{3\bar{m}}$. Then, we have the following:

$$c = \mathbf{B}^\top[\mathbf{s}\|\mathsf{M}] + \hat{\mathbf{I}} \sum_{i=0}^{k-1} 2^i \mathbf{e}_i = \hat{\mathbf{B}}^\top \mathbf{s} + \mathsf{M} \cdot \mathbf{b} + \hat{\mathbf{I}} \sum_{i=0}^{k-1} 2^i \mathbf{e}_i \in \mathbb{Z}_q^{\bar{m}},$$

where $\hat{\mathbf{B}} \in \mathbb{Z}_q^{n \times \bar{m}}$ is the matrix excluding the last row of $\mathbf{B}$, $\mathbf{b}^\top \in \mathbb{Z}_q^{\bar{m}}$ is the last row of $\mathbf{B}$ and $\hat{\mathbf{I}} = [\mathbf{I}_{\bar{m}}|\mathbf{0}_{\bar{m} \times 2\bar{m}}]$.

Let $\mathcal{S}_{3\bar{m}}$ be the set of all permutations over $3\bar{m}$ elements. Then the following Figure 7.2 depicts the gap-$\Sigma_{3,1}$-protocol for the relation $\mathcal{R}_{\mathsf{EqTo}\star}$. Here, Com can be an arbitrary commitment scheme. The concrete parameter selection we provide in the main body is obtained by using the efficient lattice-based commitment scheme of Kawachi et al. [KTX08]. Finally, we run this protocol $t = \omega(\log \lambda)$ times in parallel to obtain our desired gap-Σ_3-protocol (See Lemma 7.2). Recall the subscript 3 signifies that we require 3 valid transcripts for the extractor to work.

The gap-Σ-protocol can be checked that it is correct. We omit the proof of special gap-soundness and special HVZK, since it follows naturally from the proofs provided in [XXW13] for the gap-Σ-protocol for the relation $\mathcal{R}_{\mathsf{Open}}$.

1. **Commitment:** The Prover samples $\mathbf{v} \leftarrow \mathbb{Z}_q^n$, $\mathbf{r}_i \leftarrow \mathbb{Z}_q^{3\bar{m}}$, $\pi_i \leftarrow \mathcal{S}_{3\bar{m}}$ for $i \in [0, k-1]$ and randomness ρ_1, ρ_2, ρ_3 to be used in Com. Then, he sends the commitment $\mathsf{CMT} = (C_1, C_2, C_3)$ to the verifier, where

$$C_1 = \mathsf{Com}\Big((\pi_i)_{i=1}^{k-1}, \mathbf{t}_1 = \hat{\mathbf{B}}^\top \mathbf{v} + \hat{\mathbf{I}} \sum_{i=0}^{k-1} 2^i \mathbf{r}_i; \rho_1\Big), \quad C_2 = \mathsf{Com}\big((\mathbf{t}_{2,i} = \pi_i(\mathbf{r}_i))_{i=1}^{k-1}; \rho_2\big),$$
$$C_3 = \mathsf{Com}\Big((\mathbf{t}_{3,i} = \pi_i(\mathbf{r}_i + \mathbf{e}_i))_{i=1}^{k-1}; \rho_3\Big).$$

2. **Challenge:** The Verifier sends a challenge $\mathsf{Ch} \leftarrow \{1, 2, 3\}$ to the Prover.

3. **Response:** Depending on the value of Ch, the Prover sends the respone RSP computed as follows:

   - $\mathsf{Ch} = 1$: Set $\mathsf{RSP} = ((\mathbf{t}_{2,i}, \mathbf{t}_{3,i})_{i=0}^{k-1}, \rho_2, \rho_3)$.
   - $\mathsf{Ch} = 2$: Set $\mathsf{RSP} = (\mathbf{t}_1, (\pi_i, \mathbf{t}_{3,i})_{i=0}^{k-1}, \rho_1, \rho_3)$.
   - $\mathsf{Ch} = 3$: Set $\mathsf{RSP} = (\mathbf{t}_1, (\pi_i, \mathbf{t}_{2,i})_{i=0}^{k-1}, \rho_1, \rho_2)$.

4. **Verification:** Receiving RSP, the Verifier proceeds as follows:

   - $\mathsf{Ch} = 1$: Check that $\mathbf{t}_{3,i} - \mathbf{t}_{2,i} \in \mathcal{B}_{3\bar{m}}$ for all $i \in [0, k-1]$ and $C_2 = \mathsf{Com}((\mathbf{t}_{2,i})_{i=0}^{k-1}; \rho_2), C_3 = \mathsf{Com}((\mathbf{t}_{3,i})_{i=0}^{k-1}; \rho_3)$.
   - $\mathsf{Ch} = 2$: Check that $c + \mathbf{t}_1 - \hat{\mathbf{I}} \sum_{i=0}^{k-1} 2^i \pi_i^{-1}(\mathbf{t}_{3,i}) - \mathsf{M} \cdot \mathbf{b} \in \Lambda(\hat{\mathbf{B}})$ and $C_1 = \mathsf{Com}((\pi_i)_{i=0}^{k-1}, \mathbf{t}_1; \rho_1), C_3 = \mathsf{Com}((\mathbf{t}_{3,i})_{i=0}^{k-1}; \rho_3)$.
   - $\mathsf{Ch} = 3$: Check that $\mathbf{t}_1 - \hat{\mathbf{I}} \sum_{i=0}^{k-1} 2^i \pi_i^{-1}(\mathbf{t}_{2,i}) \in \Lambda(\hat{\mathbf{B}})$ and $C_1 = \mathsf{Com}((\pi_i)_{i=0}^{k-1}, \mathbf{t}_1; \rho_1), C_2 = \mathsf{Com}((\mathbf{t}_{2,i})_{i=0}^{k-1}; \rho_2)$.

   In each case, the Verifier outputs 1 if and only if all the check passes.

Figure 7.2: gap-$\Sigma_{3,1}$-protocol for the relation $\mathcal{R}_{\mathsf{EqTo}\star}$

Bibliography

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h) ibe in the standard model. In *EUROCRYPT*, pages 553–572. Springer, 2010.

[ABB+17] Erdem Alkim, Nina Bindel, Johannes A. Buchmann, Özgür Dagdelen, Edward Eaton, Gus Gutoski, Juliane Krämer, and Filip Pawlega. Revisiting TESLA in the quantum random oracle model. In *PQCrypto*, pages 143–162. Springer, 2017.

[ABDCP15] Michel Abdalla, Florian Bourse, Angelo De Caro, and David Pointcheval. Simple functional encryption schemes for inner products. In *PKC*, pages 733–751. Springer, 2015.

[ABP+17] Shweta Agrawal, Sanjay Bhattacherjee, Duong Hieu Phan, Damien Stehlé, and Shota Yamada. Efficient trace-and-revoke with public traceability. In *CCS*, pages 2277–2293. ACM, 2017.

[ABS17] Miguel Ambrona, Gilles Barthe, and Benedikt Schmidt. Generic transformations of predicate encodings: Constructions and applications. In *CRYPTO*, pages 36–66. Springer, 2017.

[ACF09] Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions from identity-based key encapsulation. In *EUROCRYPT*, pages 554–571. Springer, 2009.

[ACF14] Michel Abdalla, Dario Catalano, and Dario Fiore. Verifiable random functions: Relations to identity-based key encapsulation and new constructions. *Journal of Cryptology*, pages 544–593, 2014.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618. Springer, 2009.

[ADN+10] Joël Alwen, Yevgeniy Dodis, Moni Naor, Gil Segev, Shabsi Walfish, and Daniel Wichs. Public-key encryption in the bounded-retrieval model. In *EUROCRYPT*, pages 113–134. Springer, 2010.

[AFL16] Daniel Apon, Xiong Fan, and Feng-Hao Liu. Fully-secure lattice-based ibe as compact as pke. Cryptology ePrint Archive, Report 2016/125, 2016. https://eprint.iacr.org/2016/125.pdf.

[AFV11] Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *ASIACRYPT*, pages 21–40. Springer, 2011.

[AHY15] Nuttapong Attrapadung, Goichiro Hanaoka, and Shota Yamada. A framework for identity-based encryption with almost tight security. In *ASIACRYPT*, pages 521–549. Springer, 2015.

[AIK04] Benny Applebaum, Yuval Ishai, and Eyal Kushilevitz. Cryptograpcrhy in NC^0. In *FOCS*, pages 166–175. IEEE, 2004.

[AJLA$^+$12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold fhe. In *EUROCRYPT*, pages 483–501. Springer, 2012.

[Ajt96] Miklós Ajtai. Generating hard instances of lattice problems. In *STOC*, pages 99–108. ACM, 1996.

[AKPW13] Joël Alwen, Stephan Krenn, Krzysztof Pietrzak, and Daniel Wichs. Learning with rounding, revisited. In *CRYPTO*, pages 57–74. Springer, 2013.

[AKS04] Manindra Agrawal, Neeraj Kayal, and Nitin Saxena. Primes is in p. *Annals of mathematics*, pages 781–793, 2004.

[AL10] Nuttapong Attrapadung and Benoît Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In *PKC*, pages 384–402. Springer, 2010.

[ALDP11] Nuttapong Attrapadung, Benoît Libert, and Elie De Panafieu. Expressive key-policy attribute-based encryption with constant-size ciphertexts. In *PKC*, pages 90–108. Springer, 2011.

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In *CRYPTO*, pages 333–362. Springer, 2016.

[ARU14] Andris Ambainis, Ansis Rosmanis, and Dominique Unruh. Quantum attacks on classical proof systems: The hardness of quantum rewinding. In *FOCS*, pages 474–483. IEEE, 2014.

[AS15] Jacob Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions. In *PKC*, pages 236–255. Springer, 2015.

[Att14] Nuttapong Attrapadung. Dual system encryption via doubly selective security: Framework, fully secure functional encryption for regular languages, and more. In *EUROCRYPT*, pages 557–577. Springer, 2014.

[Att16] Nuttapong Attrapadung. Dual system encryption framework in prime-order groups via computational pair encodings. In *ASIACRYPT*, pages 591–623. Springer, 2016.

[Bar89] David A Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in nc1. *Journal of Computer and System Sciences*, 38(1):150–164, 1989.

[BB04a] Dan Boneh and Xavier Boyen. Efficient selective-id secure identity-based encryption without random oracles. In *EUROCRYPT*, pages 223–238. Springer, 2004.

[BB04b] Dan Boneh and Xavier Boyen. Secure identity based encryption without random oracles. In *CRYPTO*, pages 443–459. Springer, 2004.

[BBG05] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In *EUROCRYPT*, pages 440–456. Springer, 2005.

[BBS98] Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In *EUROCRYPT*, pages 127–144. Springer, 1998.

[BCH86] Paul W Beame, Stephen A Cook, and H James Hoover. Log depth circuits for division and related problems. *SIAM Journal on Computing*, pages 994–1003, 1986.

[BCK$^+$14] Fabrice Benhamouda, Jan Camenisch, Stephan Krenn, Vadim Lyubashevsky, and Gregory Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *ASIACRYPT*, pages 551–572. Springer, 2014.

[BDF$^+$11] Dan Boneh, Özgür Dagdelen, Marc Fischlin, Anja Lehmann, Christian Schaffner, and Mark Zhandry. Random oracles in a quantum world. In *EUROCRYPT*, pages 41–69. Springer, 2011.

[BDOP16] Carsten Baum, Ivan Damgård, Sabine Oechsner, and Chris Peikert. Efficient commitments and zero-knowledge protocols from ring-sis with applications to lattice-based threshold cryptosystems. Cryptology ePrint Archive, Report 2016/997, 2016. https://eprint.iacr.org/2016/997.pdf.

[BDPMW16] Florian Bourse, Rafaël Del Pino, Michele Minelli, and Hoeteck Wee. Fhe circuit privacy almost for free. In *CRYPTO*, pages 62–89. Springer, 2016.

[BF01] Dan Boneh and Matt Franklin. Identity-based encryption from the weil pairing. In *CRYPTO*, pages 213–229. Springer, 2001.

[BF11] Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In *PKC*, pages 1–16. Springer, 2011.

[BF14] Mihir Bellare and Georg Fuchsbauer. Policy-based signatures. In *PKC*, pages 520–537. Springer, 2014.

[BFM88] Manuel Blum, Paul Feldman, and Silvio Micali. Non-interactive zero-knowledge and its applications. In *STOC*, pages 103–112. ACM, 1988.

[BFW16] David Bernhard, Marc Fischlin, and Bogdan Warinschi. On the hardness of proving cca-security of signed elgamal. In *PKC*, pages 47–69. Springer, 2016.

[BG14] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In *CT-RSA*, pages 28–47. Springer, 2014.

[BGG$^+$14a] 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*, pages 533–556. Springer, 2014.

[BGG$^+$14b] 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*, pages 533–556. Springer, 2014.

[BGH07] Dan Boneh, Craig Gentry, and Michael Hamburg. Space-efficient identity based encryptionwithout pairings. In *FOCS*, pages 647–657. IEEE, 2007.

[BGJS17] Saikrishna Badrinarayanan, Vipul Goyal, Aayush Jain, and Amit Sahai. A note on vrfs from verifiable functional encryption. Cryptology ePrint Archive, Report 2017/051, 2017. https://eprint.iacr.org/2017/051.pdf.

[BGW05] Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In *CRYPTO*, pages 258–275. Springer, 2005.

[BH08] Dan Boneh and Michael Hamburg. Generalized identity based and broadcast encryption schemes. In *ASIACRYPT*, pages 455–470. Springer, 2008.

[Bit17] Nir Bitansky. Verifiable random functions from non-interactive witness-indistinguishable proofs. In *TCC*, pages 567–594. Springer, 2017.

[BKLP15] Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In *ESORICS*, pages 305–325. Springer, 2015.

[BKOS00] Mark De Berg, Marc Van Kreveld, Mark Overmars, and Otfried Cheong Schwarzkopf. Computational geometry. In *Computational geometry*. Springer, 2000.

[BKPW12] Mihir Bellare, Eike Kiltz, Chris Peikert, and Brent Waters. Identity-based (lossy) trapdoor functions and applications. In *EUROCRYPT*, pages 228–245. Springer, 2012.

[BL16] Xavier Boyen and QinYi Li. Towards tightly secure lattice short signature and id-based encryption. In *ASIACRYPT*, pages 404–434. Springer, 2016.

[BLL$^+$15] Shi Bai, Adeline Langlois, Tancrède Lepoint, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: using the rényi divergence rather than the statistical distance. In *ASIACRYPT*, pages 3–24. Springer, 2015.

[BLP$^+$13] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584, 2013.

[BLS01] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the weil pairing. In *ASIACRYPT*, pages 514–532. Springer, 2001.

[BMR10] Dan Boneh, Hart William Montgomery, and Ananth Raghunathan. Algebraic pseudorandom functions with improved efficiency from the augmented cascade. In *CCS*, pages 131–140. ACM, 2010.

[BN06] Mihir Bellare and Gregory Neven. Multi-signatures in the plain public-key model and a general forking lemma. In *CCS*, pages 390–399. ACM, 2006.

[Boy10] Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *PKC*, pages 499–517. Springer, 2010.

[BPR12] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737. Springer, 2012.

[BPVY00] Ernest Brickell, David Pointcheval, Serge Vaudenay, and Moti Yung. Design validations for discrete logarithm based signature schemes. In *PKC*, pages 276–292. Springer, 2000.

[BPW12] David Bernhard, Olivier Pereira, and Bogdan Warinschi. How not to prove yourself: Pitfalls of the fiat-shamir heuristic and applications to helios. In *ASIACRYPT*, pages 626–643. Springer, 2012.

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *CCS*, pages 62–73. ACM, 1993.

[BR09] Mihir Bellare and Thomas Ristenpart. Simulation without the artificial abort: Simplified proof and improved concrete security for waters ' ibe scheme. In *EUROCRYPT*, pages 407–424. Springer, 2009.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *TCC*, pages 253–273. Springer, 2011.

[BV11] Zvika Brakerski and Vinod Vaikuntanathan. Efficient fully homomorphic encryption from (standard) lwe. In *FOCS*, pages 97–106. IEEE, 2011.

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554. Springer, 2007.

[CCZ11] Yu Chen, Liqun Chen, and Zongyang Zhang. Cca secure ib-kem from the computational bilinear diffie-hellman assumption in the standard model. In *Information Security and Cryptology–ICISC*, pages 275–301. Springer, 2011.

[CGGI16] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachene. Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In *ASIACRYPT*, pages 3–33. Springer, 2016.

[CGW15] Jie Chen, Romain Gay, and Hoeteck Wee. Improved dual system abe in prime-order groups via predicate encodings. In *EUROCRYPT*, pages 595–624. Springer, 2015.

[Che06] Jung Hee Cheon. Security analysis of the strong diffie-hellman problem. In *EUROCRYPT*, pages 1–11. Springer, 2006.

[CHKP10] David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In *EUROCRYPT*, pages 523–552. Springer, 2010.

[CLR16] Jie Chen, Benoît Libert, and Somindu C. Ramanna. Non-zero inner product encryption with short ciphertexts and private keys. In *SCN*, pages 23–41. Springer, 2016.

[Coc01] Clifford Cocks. An identity based encryption scheme based on quadratic residues. In *Cryptography and Coding*, pages 360–363. Springer, 2001.

[Cor09] Jean-Sébastien Coron. A variant of boneh-franklin ibe with a tight reduction in the random oracle model. *Designs, Codes and Cryptography*, pages 115–133, 2009.

[CS98] Ronald Cramer and Victor Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO*, pages 13–25. Springer, 1998.

[CVH91] David Chaum and Eugène Van Heyst. Group signatures. In *EUROCRYPT*, pages 257–265. Springer, 1991.

[CW13] Jie Chen and Hoeteck Wee. Fully,(almost) tightly secure ibe and dual system groups. In *CRYPTO*, pages 435–460. Springer, 2013.

[CW14] Jie Chen and Hoeteck Wee. Doubly spatial encryption from DBDH. *Theory of Computer Science*, pages 79–89, 2014.

[DDN91] Danny Dolev, Cynthia Dwork, and Moni Naor. Nonmalleable cryptography. In *STOC*, pages 542–552. ACM, 1991.

[DF89] Yvo Desmedt and Yair Frankel. Threshold cryptosystems. In *CRYPTO*, pages 307–315. Springer, 1989.

[DF02] Ivan Damgård and Eiichiro Fujisaki. A statistically-hiding integer commitment scheme based on groups with hidden order. In *ASIACRYPT*, pages 77–85. Springer, 2002.

[DG17] Nico Döttling and Sanjam Garg. Identity-based encryption from the diffie-hellman assumption. In *CRYPTO*, pages 537–569. Springer, 2017.

[DGK$^+$10] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, pages 361–381. Springer, 2010.

[DH76] Whitfield Diffie and Martin Hellman. New directions in cryptography. *IEEE transactions on Information Theory*, 22(6):644–654, 1976.

[DKNY18] Alex Davidson, Shuichi Katsumata, Ryo Nishimaki, and Shota Yamada. Constrained prfs for bit-fixing from owfs with constant collusion resistance. Cryptology ePrint Archive, Report 2018/982, 2018. https://eprint.iacr.org/2018/982.pdf.

[DLP14] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over ntru lattices. In *ASIACRYPT*, pages 22–41. Springer, 2014.

[DM14] Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO*, pages 335–352. Springer, 2014.

[DM15] Léo Ducas and Daniele Micciancio. Fhew: Bootstrapping homomorphic encryption in less than a second. In *EUROCRYPT*, pages 617–640. Springer, 2015.

[DORS04] Yevgeniy Dodis, Rafail Ostrovsky, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In *EUROCRYPT*, pages 523–540. Springer, 2004.

[DY05] Yevgeniy Dodis and Aleksandr Yampolskiy. A verifiable random function with short proofs and keys. In *PKC*, pages 416–431. Springer, 2005.

[EE16] Rachid El Bansarkhani and Ali El Kaafarani. Post-quantum attribute-based signatures from lattice assumptions. Cryptology ePrint Archive, Report 2016/823, 2016. http://eprint.iacr.org/2016/823.

[EGK14] Ali El Kaafarani, Essam Ghadafi, and Dalia Khader. Decentralized traceable attribute-based signatures. In *CT-RSA*, pages 327–348. Springer, 2014.

[EK18] Ali El Kaafarani and Shuichi Katsumata. Attribute-based signatures for unbounded circuits in the rom and efficient instantiations from lattices. In *PKC*, pages 89–119. Springer, 2018.

[EKS17] Ali El Kaafarani, Shuichi Katsumata, and Ravital Solomon. Anonymous reputation systems achieving full dynamicity from lattices. To appear in FC, Springer, 2017.

[ElG84] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. In *CRYPTO*, pages 10–18. Springer, 1984.

[FHPS13] Eduarda SV Freire, Dennis Hofheinz, Kenneth G Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In *CRYPTO*, pages 513–530. Springer, 2013.

[FKMV12] Sebastian Faust, Markulf Kohlweiss, Giorgia Azzurra Marson, and Daniele Venturi. On the non-malleability of the fiat-shamir transform. In *INDOCRYPT*, pages 60–79. Springer, 2012.

[FLS99] Uriel Feige, Dror Lapidot, and Adi Shamir. Multiple noninteractive zero knowledge proofs under general assumptions. *SIAM Journal on Computing*, pages 1–28, 1999.

[FN93] Amos Fiat and Moni Naor. Broadcast encryption. In *CRYPTO*, pages 480–491. Springer, 1993.

[FO97] Eiichiro Fujisaki and Tatsuaki Okamoto. Statistical zero knowledge protocols to prove modular polynomial relations. In *CRYPTO*, pages 16–30. Springer, 1997.

[FS86] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194. Springer, 1986.

[Gal10] David Galindo. Chosen-ciphertext secure identity-based encryption from computational bilinear diffie-hellman. In *Pairing-Based Cryptography–Pairing*, pages 367–376. Springer, 2010.

[GDCC16] Junqing Gong, Xiaolei Dong, Jie Chen, and Zhenfu Cao. Efficient ibe with tight reduction to standard assumption in the multi-challenge setting. In *ASIACRYPT*, pages 624–654. Springer, 2016.

[Gen06] Craig Gentry. Practical identity-based encryption without random oracles. In *EUROCRYPT*, pages 445–464. Springer, 2006.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–169. ACM, 2009.

[GGH$^+$13] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49. IEEE, 2013.

[GGM86] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to construct random functions. *Journal of the ACM*, pages 792–807, 1986.

[GHKW17] Rishab Goyal, Susan Hohenberger, Venkata Koppula, and Brent Waters. A generic approach to constructing and proving verifiable random functions. Cryptology ePrint Archive, Report 2017/021, 2017. https://eprint.iacr.org/2017/021.pdf.

[GKPV10] Shafi Goldwasser, Yael Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. *Innovations in Computer Science*, pages 230–240, 2010.

[GKW17] Rishab Goyal, Venkata Koppula, and Brent Waters. Lockable obfuscation. In *FOCS*, pages 612–621. IEEE, 2017.

[GL89] Oded Goldreich and Leonid A Levin. A hard-core predicate for all one-way functions. In *STOC*, pages 25–32. ACM, 1989.

[GM82] Shafi Goldwasser and Silvio Micali. Probabilistic encryption & how to play mental poker keeping secret all partial information. In *STOC*, pages 365–377. ACM, 1982.

[GM84] Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of computer and system sciences*, pages 270–299, 1984.

[GMW15] Romain Gay, Pierrick Méaux, and Hoeteck Wee. Predicate encryption for multi-dimensional range queries from lattices. In *PKC*, pages 752–776. Springer, 2015.

[Gol86] Oded Goldreich. Two remarks concerning the goldwasser-micali-rivest signature scheme. In *CRYPTO*, pages 104–110. Springer, 1986.

[Gol08] Oded Goldreich. *Computational Complexity: A Conceptual Perspective*. Cambridge University Press, 2008.

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *CCS*, pages 89–98. ACM, 2006.

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. ACM, 2008.

[GSW13] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, pages 75–92. Springer, 2013.

[GV15] Sergey Gorbunov and Dhinakaran Vinayagamurthy. Riding on asymmetry: Efficient ABE for branching programs. In *ASIACRYPT*, pages 550–574. Springer, 2015.

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554. ACM, 2013.

[GVW15a] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Predicate encryption for circuits from lwe. In *CRYPTO*, pages 503–523. Springer, 2015.

[GVW15b] Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, pages 469–477. ACM, 2015.

[HJ12] Dennis Hofheinz and Tibor Jager. Tightly secure signatures and public-key encryption. In *CRYPTO*, pages 590–607. Springer, 2012.

[HJ16] Dennis Hofheinz and Tibor Jager. Verifiable random functions from standard assumptions. In *TCC*, pages 336–362. Springer, 2016.

[HJKS10] Kristiyan Haralambiev, Tibor Jager, Eike Kiltz, and Victor Shoup. Simple and efficient public-key encryption from computational diffie-hellman in the standard model. In *PKC*, pages 1–18. Springer, 2010.

[HK08] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In *CRYPTO*, pages 21–38. Springer, 2008.

[HKS15] Dennis Hofheinz, Jessica Koch, and Christoph Striecks. Identity-based encryption with (almost) tight security in the multi-instance, multi-ciphertext setting. In *PKC*, pages 799–822. Springer, 2015.

[HLLR12] Javier Herranz, Fabien Laguillaumie, Benoît Libert, and Carla Ràfols. Short attribute-based signatures for threshold predicates. In *CT-RSA*, pages 51–67. Springer, 2012.

[HW10] Susan Hohenberger and Brent Waters. Constructing verifiable random functions with large input spaces. In *EUROCRYPT*, pages 656–672. Springer, 2010.

[IK00] Yuval Ishai and Eyal Kushilevitz. Randomizing polynomials: a new representation with applications to round-efficient secure computation. In *FOCS*, pages 294–304. IEEE, 2000.

[IK02] Yuval Ishai and Eyal Kushilevitz. Perfect constant-round secure computation via perfect randomizing polynomials. In *ICALP*, pages 244–256. Springer, 2002.

[IR89] Russell Impagliazzo and Steven Rudich. Limits on the provable consequences of one-way permutations. In *STOC*, pages 44–61. ACM, 1989.

[Jag15] Tibor Jager. Verifiable random functions from weaker assumptions. In *TCC*, pages 121–143. Springer, 2015.

[JR13] Charanjit S Jutla and Arnab Roy. Shorter quasi-adaptive nizk proofs for linear subspaces. In *ASIACRYPT*, pages 1–20. Springer, 2013.

[Kat10] Jonathan Katz. *Digital signatures.* Springer Science & Business Media, 2010.

[Kat17] Shuichi Katsumata. On the untapped potential of encoding predicates by arithmetic circuits and their applications. In *ASIACRYPT*, pages 95–125. Springer, 2017.

[KLS18] Eike Kiltz, Vadim Lyubashevsky, and Christian Schaffner. A concrete treatment of fiat-shamir signatures in the quantum random-oracle model. In *EUROCRYPT*, pages 552–586. Springer, 2018.

[KMT19] Shuichi Katsumata, Takahiro Matsuda, and Atsushi Takayasu. Lattice-based revocable (hierarchical) ibe with decryption key exposure resistance. To appear in PKC, Springer, 2019.

[KNYY19] Shuichi Katsumata, Ryo Nishimaki, Shota Yamada, and Takashi Yamakawa. Designated verifier/prover and preprocessing nizks from diffie-hellman assumptions. Under Submission, 2019.

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, pages 146–162. Springer, 2008.

[KTX08] Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *ASIACRYPT*, pages 372–389. Springer, 2008.

[KW03] Jonathan Katz and Nan Wang. Efficiency improvements for signature schemes with tight security reductions. In *CCS*, pages 155–164. ACM, 2003.

[KY16] Shuichi Katsumata and Shota Yamada. Partitioning via non-linear polynomial functions: more compact ibes from ideal lattices and bilinear maps. In *ASIACRYPT*, pages 682–712. Springer, 2016.

[KY19a] Shuichi Katsumata and Shota Yamada. Group signatures without nizk: From lattices in the standard model. Under Submission, 2019.

[KY19b] Shuichi Katsumata and Shota Yamada. Non-zero inner product encryption schemes from various assumptions: Lwe, ddh and dcr. To appear in PKC, Springer, 2019.

[KYY18] Shuichi Katsumata, Shota Yamada, and Takashi Yamakawa. Tighter security proofs for gpv-ibe in the quantum random oracle model. In *ASIACRYPT*, pages 253–282. Springer, 2018.

[LLM$^+$16] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In *ASIACRYPT*, pages 101–131. Springer, 2016.

[LLNW14] Adeline Langlois, San Ling, Khoa Nguyen, and Huaxiong Wang. Lattice-based group signature scheme with verifier-local revocation. In *PKC*, pages 345–361. Springer, 2014.

[LNSW13] San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang. Improved zero-knowledge proofs of knowledge for the isis problem, and applications. In *PKC*, pages 107–124, 2013.

[LNW15] San Ling, Khoa Nguyen, and Huaxiong Wang. Group signatures from lattices: simpler, tighter, shorter, ring-based. In *PKC*, pages 427–449. Springer, 2015.

[LOS$^+$10] Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In *EUROCRYPT*, pages 62–91. Springer, 2010.

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, pages 1–23. Springer, 2010.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. In *EUROCRYPT*, pages 35–54. Springer, 2013.

[LPRTJ05] Alexander E Litvak, Alain Pajor, Mark Rudelson, and Nicole Tomczak-Jaegermann. Smallest singular value of random matrices and geometry of random polytopes. *Advances in Mathematics*, pages 491–523, 2005.

[LS15] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, pages 565–599, 2015.

[LSSS17] Benoît Libert, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. All-but-many lossy trapdoor functions and selective opening chosen-ciphertext security from lwe. In *CRYPTO*, pages 332–364. Springer, 2017.

[LW10] Allison Lewko and Brent Waters. New techniques for dual system encryption and fully secure hibe with short ciphertexts. In *TCC*, pages 455–479. Springer, 2010.

[LW11] Allison B. Lewko and Brent Waters. Unbounded HIBE and attribute-based encryption. In *EUROCRYPT*, pages 547–567, 2011.

[Lys02] Anna Lysyanskaya. Unique signatures and verifiable random functions from the dh-ddh separation. In *CRYPTO*, pages 597–612. Springer, 2002.

[Lyu09] Vadim Lyubashevsky. Fiat-shamir with aborts: applications to lattice and factoring-based signatures. In *ASIACRYPT*, pages 598 – 616. Springer, 2009.

[Lyu12] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, pages 738–755. Springer, 2012.

[MH78] Ralph Merkle and Martin Hellman. Hiding information and signatures in trapdoor knapsacks. *IEEE transactions on Information Theory*, pages 525–530, 1978.

[Mic04] Daniele Micciancio. Almost perfect lattices, the covering radius problem, and applications to ajtai's connection factor. *SIAM Journal on Computing*, pages 118–169, 2004.

[MM11] Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of lwe search-to-decision reductions. In *CRYPTO*, pages 465–484. Springer, 2011.

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718. Springer, 2012.

[MP13] Daniele Micciancio and Chris Peikert. Hardness of sis and lwe with small parameters. In *CRYPTO*, pages 21–39. Springer, 2013.

[MPR11] Hemanta K Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In *CT-RSA*, pages 376–392. Springer, 2011.

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *FOCS*, pages 372–381. IEEE, 2004.

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM Journal on Computing*, pages 267–302, 2007.

[MRV99] Silvio Micali, Michael Rabin, and Salil Vadhan. Verifiable random functions. In *FOCS*, pages 120–130. IEEE, 1999.

[MSK02] Shigeo Mitsunari, Ryuichi Sakai, and Masao Kasahara. A new traitor tracing. *transactions on fundamentals of electronics, communications and computer sciences*, pages 481–484, 2002.

[Nac07] David Naccache. Secure and practical identity-based encryption. *IET Information Security*, pages 59–64, 2007.

[Nao91] Moni Naor. Bit commitment using pseudorandomness. *Journal of cryptology*, pages 151–158, 1991.

[NC00] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[NR97] Moni Naor and Omer Reingold. Number-theoretic constructions of efficient pseudo-random functions. In *FOCS*, pages 458–467. IEEE, 1997.

[O'N10] Adam O'Neill. Definitional issues in functional encryption. *IACR Cryptology ePrint Archive*, 2010:556, 2010.

[OP01] Tatsuaki Okamoto and David Pointcheval. The gap-problems: A new class of problems for the security of cryptographic schemes. In *PKC*, pages 104–118. Springer, 2001.

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In *CCS*, pages 195–203. ACM, 2007.

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010.

[OT11] Tatsuaki Okamoto and Katsuyuki Takashima. Efficient attribute-based signatures for non-monotone predicates in the standard model. In *PKC*, pages 35–52. Springer, 2011.

[OT13] Tatsuaki Okamoto and Katsuyuki Takashima. Decentralized attribute-based signatures. In *PKC*, pages 125–142. Springer, 2013.

[OT15] Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. *Designs, Codes and Cryptography*, pages 725–771, 2015.

[Pei07] Chris Peikert. Limits on the hardness of lattice problems in ell _p norms. In *Conference on Computational Complexity*, pages 333–346. IEEE, 2007.

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. ACM, 2009.

[Pei10] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In *CRYPTO*, pages 80–97. Springer, 2010.

[PR06] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, pages 145–166. Springer, 2006.

[PRS17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-lwe for any ring and modulus. In *STOC*, pages 461–473. ACM, 2017.

[PS00] David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of cryptology*, pages 361–396, 2000.

[PSV06] Rafael Pass, Abhi Shelat, and Vinod Vaikuntanathan. Construction of a non-malleable encryption scheme from any semantically secure one. In *CRYPTO*, pages 271–289. Springer, 2006.

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571. Springer, 2008.

[RAD78] Ronald L Rivest, Len Adleman, and Michael L Dertouzos. On data banks and privacy homomorphisms. *Foundations of Secure Computation*, 1978.

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93. ACM Press, 2005.

[Reg10] Oded Regev. The learning with errors problem. *Invited survey in CCC*, 2010.

[Rom90] John Rompel. One-way functions are necessary and sufficient for secure signatures. In *STOCS*, pages 387–394. ACM, 1990.

[RS91] Charles Rackoff and Daniel R Simon. Non-interactive zero-knowledge proof of knowledge and chosen ciphertext attack. In *CRYPTO*, pages 433–444. Springer, 1991.

[RSA78] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, pages 120–126, 1978.

[RST01] Ronald L Rivest, Adi Shamir, and Yael Tauman. How to leak a secret. In *ASIACRYPT*, pages 552–565. Springer, 2001.

[RW04] Renato Renner and Stefan Wolf. Smooth rényi entropy and applications. In *ISIT*, pages 233–233. IEEE, 2004.

[SAH16] Yusuke Sakai, Nuttapong Attrapadung, and Goichiro Hanaoka. Attribute-based signatures for circuits from bilinear map. In *PKC*, pages 283–300. Springer, 2016.

[SBC$^+$07] Elaine Shi, John Bethencourt, TH Hubert Chan, Dawn Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In *S&P*, pages 350–364. IEEE, 2007.

[Sha79] Adi Shamir. How to share a secret. *Communications of the ACM*, pages 612–613, 1979.

[Sha82] Adi Shamir. A polynomial time algorithm for breaking the basic merkle-hellman cryptosystem. In *FOCS*, pages 145–152. IEEE, 1982.

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In *CRYPTO*, pages 47–53. Springer, 1985.

[Sho94a] Peter W Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *FOCS*, pages 124–134. IEEE, 1994.

[Sho94b] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *FOCS*, pages 124–134. IEEE, 1994.

[SKAH18] Yusuke Sakai, Shuichi Katsumata, Nuttapong Attrapadung, and Goichiro Hanaoka. Attribute-based signatures for unbounded languages from standard assumptions. In *ASIACRYPT*, pages 493–522. Springer, 2018.

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairings. In *SCIS*, 2000. (In Japanese).

[SPB12] Kunwar Singh, C Pandurangan, and AK Banerjee. Adaptively secure efficient lattice (h) ibe in standard model with short public parameters. In *SPACE*, pages 153–172. Springer, 2012.

[SS11] Damien Stehlé and Ron Steinfeld. Making ntru as secure as worst-case problems over ideal lattices. In *EUROCRYPT*, pages 27–47. Springer, 2011.

[SSTX09] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In *ASIACRYPT*, pages 617–635. Springer, 2009.

[STW96] Michael Steiner, Gene Tsudik, and Michael Waidner. Diffie-hellman key distribution extended to group communication. In *CCS*, pages 31–37. ACM, 1996.

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473. Springer, 2005.

[SXY18] Tsunekazu Saito, Keita Xagawa, and Takashi Yamakawa. Tightly-secure key-encapsulation mechanism in the quantum random oracle model. In *EUROCRYPT*, pages 520–551. Springer, 2018.

[SY10] Amir Shpilka and Amir Yehudayoff. Arithmetic circuits: A survey of recent results and open questions. *Foundations and Trends® in Theoretical Computer Science*, pages 207–388, 2010.

[Tsa17] Rotem Tsabary. An equivalence between attribute-based signatures and homomorphic signatures, and new constructions for both. In *TCC*, pages 489–518. Springer, 2017.

[TU16] Ehsan Ebrahimi Targhi and Dominique Unruh. Post-quantum security of the fujisaki-okamoto and OAEP transforms. In *TCC*, pages 192–216. Springer, 2016.

[Unr14a] Dominique Unruh. Quantum position verification in the random oracle model. In *CRYPTO*, pages 1–18. Springer, 2014.

[Unr14b] Dominique Unruh. Revocable quantum timed-release encryption. In *EUROCRYPT*, pages 129–146. Springer, 2014.

[Unr15] Dominique Unruh. Non-interactive zero-knowledge proofs in the quantum random oracle model. In *EUROCRYPT*, pages 755–784. Springer, 2015.

[Unr17] Dominique Unruh. Post-quantum security of fiat-shamir. In *ASIACRYPT*, pages 65–95. Springer, 2017.

[Ver11] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. Lecture Notes, 2011. Available at http://www-personal.umich.edu/romanv/papers/ non-asymptotic-rmt-plain.pdf.

[Wat05] Brent Waters. Efficient identity-based encryption without random oracles. In *EUROCRYPT*, pages 114–127. Springer, 2005.

[Wat09] Brent Waters. Dual system encryption: Realizing fully secure ibe and hibe under simple assumptions. In *CRYPTO*, pages 619–636. Springer, 2009.

[Wee09] Hoeteck Wee. Zero knowledge in the random oracle model, revisited. In *ASIACRYPT*, pages 417–434. Springer, 2009.

[Wee14] Hoeteck Wee. Dual system encryption via predicate encodings. In *TCC*, pages 616–637. Springer, 2014.

[WZ17] Daniel Wichs and Giorgos Zirdelis. Obfuscating compute-and-compare programs under lwe. In *FOCS*, pages 600–611. IEEE, 2017.

[Xag13] Keita Xagawa. Improved (hierarchical) inner-product encryption from lattices. In *PKC*, pages 235–252. Springer, 2013.

[XXW13] Xiang Xie, Rui Xue, and Minqian Wang. Zero knowledge proofs from ring-lwe. In *CANS*, pages 57–73. Springer, 2013.

[YAHK14] Shota Yamada, Nuttapong Attrapadung, Goichiro Hanaoka, and Noboru Kunihiro. A framework and compact constructions for non-monotonic attribute-based encryption. In *PKC*, pages 275–292, 2014.

[YAL$^+$17] Rupeng Yang, Man Ho Au, Junzuo Lai, Qiuliang Xu, and Zuoxia Yu. Lattice-based techniques for accountable anonymity: Composition of abstract stern ’s protocols and weak prf with efficient protocols from lwr. Cryptology ePrint Archive, Report 2017/781, 2017. https://eprint.iacr.org/2017/781.pdf.

[Yam16] Shota Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In *EUROCRYPT*, pages 32–62. Springer, 2016.

[Yam17] Shota Yamada. Asymptotically compact adaptively secure lattice ibes and verifiable random functions via generalized partitioning techniques. In *CRYPTO*, pages 161–193. Springer, 2017.

[YKHK10] Shota Yamada, Yutaka Kawai, Goichiro Hanaoka, and Noboru Kunihiro. Public key encryption schemes from the (b) cdh assumption with better efficiency. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, pages 1984–1993, 2010.

[ZCZ16] Jian Zhang, Yu Chen, and Zhenfeng Zhang. Programmable hash functions from latties: Short signatures and ibes with small key sizes. In *CRYPT0*, pages 303–332. Springer, 2016.

[Zha12a] Mark Zhandry. How to construct quantum random functions. In *FOCS*, pages 679–687. IEEE, 2012.

[Zha12b] Mark Zhandry. Secure identity-based encryption in the quantum random oracle model. In *CRYPTO*, pages 758–775. Springer, 2012.