Non-Zero Inner Product Encryption Schemes from Various Assumptions: LWE, DDH and DCR

Shuichi Katsumata[1,2] and Shota Yamada[2]

[1]The University of Tokyo, Tokyo ,Japan
shuichi_katsumata@it.k.u-tokyo.ac.jp
[2]National Institute of Advanced Industrial Science and Technology (AIST), Tokyo, Japan
yamada-shota@aist.go.jp

Abstract

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 Introduction

1.1 Background

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 paper, 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 direct first construction of an NIPE scheme using bilinear maps.[1] 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$^+$14] 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 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 on 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. Due to its usefulness in practice, negated policies in the area of ABE have 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, where it currently seems that stronger assumptions such as the DBDH or DLIN assumption is required to construct its counterpart — IPE schemes. Therefore, although an NIPE scheme may be simply understood as an IPE scheme in the opposite flavor, our result

[1] We note that Goyal et al. [GPSW06] propose an ABE scheme for $\mathbf{NC}^1$ circuit, which in turn implies a NIPE scheme, since the computation of inner products can be performed in $\mathbf{NC}^1$. However, the resulting construction is highly inefficient.

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.

1.2 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)[2] . 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 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 our lattice-based NIPE schemes and the generic construction of NIPE schemes from LinFE. We believe the techniques we utilized for the lattice-based direct NIPE construction to be of independent interest.

Lattice-Based Constructions. We propose two NIPE schemes built directly from lattices.

[2] 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.

At a high level, our two NIPE constructions share many similarities; both constructions highly depart from the previous lattice-based ABE constructions [GVW13, BGG$^+$14, 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 $\mathbf{T_A} \in \mathbb{Z}^{m\times m}$ 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 knowledgeable readers of lattice-based cryptography, 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 $\mathbf{G}$ [MP12] is a special matrix with a publicly known trapdoor $\mathbf{T_G}$.

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 $\mathbf{B}_i = \mathbf{AR}_i \mod q$. 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$^+$14], 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 $\mathbf{c}_0 = \mathbf{s}^\top\mathbf{A} + \mathbf{z}_0$. Here, $\mathbf{s}$ is a randomly sampled vector in $\mathbb{Z}_q^n$ and $\{\mathbf{z}_i\}_{i\in[0,\ell]}$ are short vectors in $\mathbb{Z}^m$ sampled from a particular discrete Gaussian distribution. Then, for decryption, a user with predicate vector $\vec{y}$ computes the following:

$$\sum_{i=1}^{\ell} y_i \cdot \mathbf{c}_i = \mathbf{s}^\top(\sum_{i=1}^{\ell} y_i \mathbf{B}_i + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}) + \mathsf{noise} = \mathbf{s}^\top(\mathbf{AR}_{\vec{y}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}) + \mathsf{noise}.$$

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 $[\mathbf{A}|\mathbf{AR}_{\vec{y}} + \langle \vec{x}, \vec{y} \rangle \cdot \mathbf{G}]\mathbf{e} = \mathbf{u} \mod q$. 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 for 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 $\mathbf{AR}_i^{\mathsf{SIM}} - x_i^* \cdot \mathbf{G}$. 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 $\{\mathbf{R}^{\mathsf{SIM}}_{\vec{y}^{(j)}}\}_{j \in [Q]}$ 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)[3] 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

[3] 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.

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.

2 Preliminaries

Notation. We treat vectors in their column form. For a vector $\mathbf{v} \in \mathbb{R}^n$, denote $\|\mathbf{v}\|$ as 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 (spectral norm). We denote $[\cdot|\cdot]$ as the horizontal 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 denote $[a, b]$ as the set $\{a, a+1, \ldots, b-1, b\}$ for any integers $a, b \in \mathbb{N}$ satisfying $a \leq b$, and for simplicity write $[b]$ for the special case $a = 1$. Statistical distance between two random variables X and Y with support Ω is defined as $\Delta(X; Y) = \frac{1}{2} \sum_{s \in \Omega} |\Pr[X = s] - \Pr[Y = s]|$. 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.

2.1 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 outputs a private key $\mathsf{sk}_{\vec{y}}$ and a updated state st. We assume that $\vec{y}$ is implicitly included in $\mathsf{sk}_{\vec{y}}$.

$\mathsf{Encrypt}(\mathsf{MPK}, \vec{x}, \mathsf{M}) \to C$: 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 C.

$\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, (\vec{x}, C)) \to \mathsf{M}$ or $\perp$: The decryption algorithm takes as input the master public key MPK, a private key $\mathsf{sk}_{\vec{y}}$, and a ciphertext C 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 $\Pr[\mathsf{Dec}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Enc}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \mathsf{M}] = 1 - \mathsf{negl}(\lambda)$
- if $\langle \vec{x}, \vec{y} \rangle = 0$, then $\Pr[\mathsf{Dec}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Enc}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \perp] = 1 - \mathsf{negl}(\lambda)$,

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 $C^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ 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.

2.2 Lattices

A (full-rank-integer) m-dimensional lattice Λ in $\mathbb{Z}^m$ is a set of the form $\{\sum_{i \in [m]} x_i \mathbf{b}_i | x_i \in \mathbb{Z}\}$, where $\mathbf{B} = \{\mathbf{b}_1, \cdots, \mathbf{b}_m\}$ are m linearly independent vectors in $\mathbb{Z}^m$. We call $\mathbf{B}$ the basis of the lattice Λ. For any positive integers n, m and $q \geq 2$, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and a vector $\mathbf{u} \in \mathbb{Z}_q^n$, we define $\Lambda^\perp(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m | \mathbf{A}\mathbf{z} = \mathbf{0} \mod q\}$ and $\Lambda^\perp_{\mathbf{u}}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}^m | \mathbf{A}\mathbf{z} = \mathbf{u} \mod q\}$.

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 $\{\mathbf{V}\mathbf{M} | \mathbf{V} \in \Lambda^k\} \subseteq \mathbb{Z}^{m \times \ell}$.[4]

Gaussian Measures. For any vector $\mathbf{c} \in \mathbb{R}^m$ and positive real $\sigma > 0$, the m-dimensional Gaussian function over $\mathbb{R}^m$ centered at $\mathbf{c}$ with parameter s is defined as $\rho_{\sigma,\mathbf{c}}(\mathbf{x}) = \exp(-\pi \|\mathbf{x} - \mathbf{c}\|^2 / \sigma^2)$. The continuous Gaussian distribution D_σ over $\mathbb{R}^m$ centered at $\mathbf{c}$ with parameter σ is defined as $D_{\sigma,\mathbf{c}}(\mathbf{x}) = \rho_{\sigma,\mathbf{c}}(\mathbf{x}) / \sigma^m$. For an m-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 $\rho_{\sigma,\mathbf{c}}(\Lambda) = \sum_{\mathbf{x} \in \Lambda} \rho_{\sigma,\mathbf{c}}(\mathbf{x})$. Finally, for an m-dimensional shifted lattice $\Lambda + \mathbf{t}$,

[4] In Appendix B, we provide a more general definition of multi lattices. We omit this from the main body for simplicity as it only comes up during in the security proof.

we define the Gaussian distribution $D_{\Lambda+\mathbf{t},\sigma}$ with center $\mathbf{c} = 0$ 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.

Lemma 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*[5] *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.

Lemma 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.*

Lemma 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, $\mathbf{T}_\mathbf{A}$ be a basis for $\Lambda^\perp(\mathbf{A})$, and $\sigma > \|\mathbf{T}_\mathbf{A}\| \cdot \omega(\sqrt{\log m})$. Then, if we sample a vector $\mathbf{x} \leftarrow D_{\Lambda^\perp_\mathbf{u}(\mathbf{A}),\sigma}$, we have $\Pr[\|\mathbf{x}\| > \sqrt{m}\sigma] < \mathsf{negl}(n)$.*

Lemma 4 (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 $\mathbf{z}$ chosen from $D_{\mathbb{Z}^m,r}$. Then for any $\mathbf{V} \in \mathbb{Z}^{m\times\ell}$ and positive real $\sigma > s_1(\mathbf{V})$, there exists a PPT algorithm $\mathsf{ReRand}(\mathbf{V}, \mathbf{b}+\mathbf{z}, r, \sigma)$ that outputs $\mathbf{b}'^\top = \mathbf{b}^\top\mathbf{V} + \mathbf{z}'^\top \in \mathbb{Z}_q^\ell$ where $\mathbf{z}'$ is distributed statistically close to $D_{\mathbb{Z}^\ell,2r\sigma}$.*

Analogously to above, 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 the distribution of the sum of discrete Gaussians plays a central roll in our security proof. The proof of the theorem is given in the Appendix B with a more formal treatment on the output distribution.

Theorem 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 $\mathbf{X}\mathbf{M} \in \mathbb{Z}^{m\times(\ell-t)}$ is statistically close to to a distribution parameterized by $\Lambda^\perp(\mathbf{A}), \sigma, \mathbf{M}, (\mathbf{T}\mathbf{M} \mod \Lambda^\perp(\mathbf{A})^\ell\mathbf{M})$.

Remark 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,*

[5] 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.

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 $\mathbf{X}\mathbf{M}$, the distribution of $\mathbf{X}\bar{\mathbf{M}}$ is parameterized solely by the distribution of $\Lambda, \sigma, \mathbf{M}, (\mathbf{T}\mathbf{M} \mod \Lambda^k \mathbf{M})$. (For further discussion see Appendix B of [BF11].

Sampling Algorithms. The following lemma states useful algorithms for sampling short vectors from lattices.

Lemma 5. *Let $n, m, q > 0$ be integers with $m > n$. Then:*

- *([GPV08])* $\mathsf{SamplePre}(\mathbf{A}, \mathbf{u}, \mathbf{T}_{\mathbf{A}}, \sigma) \to \mathbf{e}$: *There exists a randomized algorithm that, given a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, a vector $\mathbf{u} \in \mathbb{Z}_q^n$, a basis $\mathbf{T}_{\mathbf{A}}$ for $\Lambda^{\perp}(\mathbf{A})$, and a Gaussian parameter $\sigma > \|\mathbf{T}_{\mathbf{A}}\|_{\mathsf{GS}} \cdot \omega(\sqrt{\log m})$, outputs a vector $\mathbf{e} \in \mathbb{Z}^m$ sampled from a distribution which is* $\mathsf{negl}(n)$*-close to $D_{\Lambda^{\perp}_{\mathbf{u}}(\mathbf{A}), \sigma}$.*
- *([ABB10])* $\mathsf{SampleRight}(\mathbf{A}, \mathbf{G}, \mathbf{R}, t, \mathbf{u}, \mathbf{T}_{\mathbf{G}}, \sigma) \to \mathbf{e}$*: 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 $D_{\Lambda^{\perp}_{\mathbf{u}}([\mathbf{A}|\mathbf{AR}+t\mathbf{G}]), \sigma}$.*
- *([MP12]) Let $m \geq n\lceil \log q \rceil$. Then, there exists a fixed full-rank matrix $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ such that the lattice $\Lambda^{\perp}(\mathbf{G})$ has publicly known basis $\mathbf{T}_{\mathbf{G}} \in \mathbb{Z}^{m \times m}$ with $\|\mathbf{T}_{\mathbf{G}}\|_{\mathsf{GS}} \leq \sqrt{5}$.*

Observe that even if we are in possession of a "nice" trapdoor matrix $\mathbf{R}$, we can not use the $\mathsf{SampleRight}$ 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 $\mathsf{SampleRight}$ 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 $\mathbb{Z}_q$. The proof is provided in Appendix A.

Lemma 6 (Algorithm $\mathsf{SampleSkewed}$)**.** *Let $q = p^d$ for a prime p and positive integer d. Then, there exists a polynomial time algorithm* $\mathsf{SampleSkewed}$ *with the following property.*

$\mathsf{SampleSkewed}(\mathbf{A}, \mathbf{G}, \mathbf{R}, t, p^{d-1}\mathbf{u}, \mathbf{T}_{\mathbf{G}}) \to \mathbf{e}$*: 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 $\|\mathbf{e}\| \leq s_1(\mathbf{R})\sqrt{m} \cdot \omega(\sqrt{\log n})$ with all but negligible probability.*

Hardness Assumptions. 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]. Looking ahead, FE.LWE will be used for our lattice-based NIPE construction over $\mathbb{Z}_p$.

Definition 1 (LWE and FE.LWE)**.** *For integers $n = n(\lambda), m = m(n), q = q(n) > 2$, an error distribution over $\chi = \chi(n)$ 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:*

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

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 (LWE to 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, MP12, BLP+13, PRSD17] handling the case of non-prime modulus q have appeared in the literatures.

Theorem 3 (Hardness of LWE. [Reg05, PRSD17])**.** *Let* n, m, q *be positive integers, and let* $\alpha \in (0,1)$ *be a positive real such that* $\alpha q \geq 2\sqrt{n}$. *Then, there exists a probabilistic polynomial-time quantum reduction from* $\mathsf{GapSVP}_{\tilde{O}(n/\alpha)}$ *to* $\mathsf{LWE}_{n,m,q,\chi}$ *with* $\chi = D_{\mathbb{Z},\alpha q}$.

3 Construction from Lattices with Inner Product over $\mathbb{Z}$

3.1 Constructions

Here we construct a *stateless* NIPE scheme with inner product space $\mathbb{Z}$. We consider the predicate space $\mathcal{P} = \{-P+1, \ldots, P-2, P-1\}^\ell \subset \mathbb{Z}^\ell$ and attribute space $\mathcal{I} = \{-I+1, \ldots, I-2, I-1\}^\ell \subset \mathbb{Z}^\ell$ 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., $\ell P I < q$. 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 Sec. 3.4.

$\mathsf{Setup}(1^n, 1^\ell)$: 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 $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i \mod q$. 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 $w = c - \mathbf{e}^\top [\mathbf{c}_0^\top | \mathbf{c}_{\vec{y}}^\top]^\top \in \mathbb{Z}_q$. Finally, it returns 1 if $|w - \lceil q/2 \rceil| < \lceil q/4 \rceil$ and 0 otherwise.

3.2 Correctness and Parameter Selection

The following lemma guarantees correctness of the scheme.

Lemma 7 (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 $\langle \vec{x}, \vec{y} \rangle \neq 0 \in \mathbb{Z}$. 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,

$$s_1(\mathbf{R}_{\vec{y}}) \ = \ s_1\big(\sum_{i=1}^{\ell} y_i \mathbf{R}_i \big) \ \leq \ C \cdot \sqrt{\ell} P \sigma \cdot \sqrt{m} \tag{1}$$

where, the inequality follows from Lemma 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,

$$\begin{aligned} |z - z''| &\le |z| + |\mathbf{e}^\top [\mathbf{z}_0^\top | \mathbf{z}'^\top]^\top| \\ &\le |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\mathbf{e}^\top \mathbf{z}'\| \qquad (2) \\ &= |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\sum_{i=1}^{\ell} y_i \cdot \mathbf{e}^\top \mathbf{z}_i\| \\ &\le \Big(\alpha q + P \cdot s_1(\mathbf{R}_{\vec{y}})\sqrt{m\ell}\alpha' q\Big)\omega(\sqrt{\log n}) \qquad (3) \\ &\le \Big(\alpha q + \ell P^2 \sigma m \alpha' q\Big)\omega(\sqrt{\log n}) \qquad (4) \end{aligned}$$

where Eq.(2) follows from the sub-additivity of the square root function $\sqrt{\cdot}$, Eq.(3) follows from the linear structure of subgaussian random variables, Lemma 5, Lemma 3 and the fact that $\vec{y} \in \mathcal{P}$, Eq.(4) follows from Eq.(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 Dec 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 7),
- the gadget matrix $\mathbf{G}$ is well defined (i.e., $m \ge n\lceil \log q \rceil$. See Lemma 5.),
- σ is sufficiently large so that $\mathbf{R}_i$'s are samplable, and Theorem 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 5),
- 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 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 4),
- the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{n}$). See Theorem 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. Notably, in case for the standard setting where all ℓ, P, and I are polynomial in n, the modulus q is also polynomial.

3.3 Security Proof

Theorem 4. *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 $\mathbf{B}_i = \mathbf{A}\mathbf{R}_i - x_i^*\mathbf{G} \mod q$ 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 $\vec{y} \in \mathcal{P}$ 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 $C^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ 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

$$\left\{ \{\mathbf{B}_i\}_{i\in[\ell]}, \quad \{\mathbf{R}_{\vec{y}^{(k)}}\}_{k\in[\ell-1]} \right\}. \tag{5}$$

Hence, we aim at proving that the view of Eq.(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 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 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:

$$\underbrace{[\mathbf{R}_{\vec{y}^{(1)}}|\mathbf{R}_{\vec{y}^{(2)}}|\cdots|\mathbf{R}_{\vec{y}^{(\ell-1)}}]}_{:=\vec{\mathbf{R}}_{\mathsf{sk}}\ \in\mathbb{Z}^{m\times m(\ell-1)}} = \underbrace{[\mathbf{R}_1|\mathbf{R}_2|\cdots|\mathbf{R}_\ell]}_{=\vec{\mathbf{R}}\ \in\mathbb{Z}^{m\times m\ell}} \underbrace{\left[\begin{array}{c|c|c|c} y_1^{(1)}\mathbf{I}_m & y_1^{(2)}\mathbf{I}_m & \cdots & y_1^{(\ell-1)}\mathbf{I}_m \\ y_2^{(1)}\mathbf{I}_m & y_2^{(2)}\mathbf{I}_m & & y_2^{(\ell-1)}\mathbf{I}_m \\ \vdots & \vdots & \cdots & \vdots \\ y_\ell^{(1)}\mathbf{I}_m & y_\ell^{(2)}\mathbf{I}_m & & y_\ell^{(\ell-1)}\mathbf{I}_m \end{array}\right]}_{:=\mathbf{M}=\mathbf{Y}\otimes\mathbf{I}_m\ \in\mathbb{Z}^{m\ell\times m(\ell-1)}}, \tag{6}$$

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 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 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) 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 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 are statistically 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

$$c = \mathbf{u}^\top\mathbf{s}+z+\mathsf{M}_b\lfloor q/2\rceil,\quad \mathbf{c}_0 = \mathbf{A}^\top\mathbf{s}+\mathbf{z}_0,\quad (\mathbf{c}_i = (\mathbf{A}\mathbf{R}_i)^\top\mathbf{s}+\mathbf{z}_i)_{i\in[\ell]} \tag{7}$$

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

$$\mathsf{ReRand}\Big([\mathbf{I}_m|\vec{\mathbf{R}}],\mathbf{v},\alpha q,\frac{\alpha'}{2\alpha}\Big)\to\mathbf{c}\in\mathbb{Z}_q^{m(\ell+1)}$$

from Lemma 4, 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

$$C^* = \Big(c = v+\mathsf{M}_b\lfloor q/2\rceil,\quad \mathbf{c}_0,\quad (\mathbf{c}_i)_{i\in[\ell]}\Big)\in\mathbb{Z}_q\times\mathbb{Z}_q^m\times(\mathbb{Z}_q^m)^\ell, \tag{8}$$

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.(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 4, 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.(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

$$\mathsf{ReRand}\Big([\mathbf{I}_m|\vec{\mathbf{R}}], \mathbf{v}, \alpha q, \frac{\alpha'}{2\alpha}\Big) \rightarrow \mathbf{c} \in \mathbb{Z}_q^{m(\ell+1)},$$

where $\mathbf{v} = \mathbf{v}' + \mathbf{z}$. Then, the challenge ciphertext is set as in Eq.(8). We show in Lemma 8 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

$$\begin{aligned}\left|\Pr[S_0] - \frac{1}{2}\right| &= \left|\sum_{i=0}^{2}(\Pr[S_i] - \Pr[S_{i+1}]) + \Pr[S_3] - \frac{1}{2}\right| \\ &\le \sum_{i=0}^{2}|\Pr[S_i] - \Pr[S_{i+1}]| + \left|\Pr[S_3] - \frac{1}{2}\right| \le \mathsf{negl}(n).\end{aligned}$$

Therefore, the probability that $\mathcal{A}$ wins Game_0 is negligible. Now, to complete the proof of Theorem 4, it remains to prove the following Lemma 8.

Lemma 8. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[S_2] - \Pr[S_3]| \le \mathsf{Adv}_{\mathcal{B}}^{\mathsf{LWE}_{n,m+1,q,\chi}}.$$

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 $\{\mathbf{a}_i, v_i\}_{i=0}^m \in \left(\mathbb{Z}_q^n \times \mathbb{Z}_q\right)^{m+1}$ 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.(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. □

□

3.4 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 4. We note that the same parameters as in the single-bit variant 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.

4 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 Sec. 3, 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 on 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 Sec. 3.4.

$\mathsf{Setup}(1^n, 1^\ell)$: 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

$$\mathsf{MPK} = \Big(\mathbf{A}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{u}\Big) \quad \text{and} \quad \mathsf{MSK} = \Big(\mathsf{st}, \{\mathbf{R}_i\}_{i \in [\ell]}\Big).$$

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 $(\vec{h}^{(i)}, \mathsf{h}^{(i)}, \mathsf{sk}_{\vec{h}^{(i)}}) \in \mathbb{Z}_p^\ell \times \mathbb{Z}^\ell \times \mathbb{Z}^{m \times m}$, 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 $\mathsf{h}^{(j')} = \vec{h}^{(j')}$, interpreted as a vector in $\mathbb{Z}^{\ell}$, and sets $\mathsf{sk}_{\vec{h}^{(j')}}$ as

$$\mathbf{R}_{\vec{h}^{(j')}} = \sum_{i=1}^{\ell} \mathsf{h}_i^{(j')} \mathbf{R}_i \in \mathbb{Z}^{m \times m}, \tag{9}$$

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.[6] 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.

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 $\mathsf{SampleSkewed}(\mathbf{A}, \mathsf{sk}_{\vec{y}} = \mathbf{R}_{\vec{y}}, \langle \vec{x}, \vec{y} \rangle, p^{d-1}\mathbf{u}, \mathbf{T}_{\mathbf{G}})$. 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] 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 $\vec{h}^{(j')}$.

4.1 Correctness and Parameter Selection

The following lemma guarantees correctness of the scheme.

Lemma 9 (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 $\langle \vec{x}, \vec{y} \rangle \neq 0 \in \mathbb{Z}_p$. First, notice that

$$\begin{aligned}
\mathbf{c}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{c}_i &= \sum_{i=1}^{\ell} \mathsf{y}_i \Big((\mathbf{B}_i + p^{d-1} \cdot x_i \mathbf{G})^\top \mathbf{s} + \mathbf{z}_i \Big) \\
&= \Bigg(\mathbf{A} \underbrace{\Big(\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \Big)}_{= \mathbf{R}_{\vec{y}}\ (= \mathsf{sk}_{\vec{y}})} + p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle \mathbf{G} \Bigg)^\top \mathbf{s} + \underbrace{\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{z}_i}_{:= \mathbf{z}'\ \text{(noise)}} \\
&= \Big(\mathbf{A} \mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{\mathsf{y}} \rangle \mathbf{G} \Big)^\top \mathbf{s} + \mathbf{z}' \mod q, \qquad (10)
\end{aligned}$$

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.(10) is equivalent to

$$\mathbf{c}_{\vec{y}} = \Big(\mathbf{A} \mathbf{R}_{\vec{y}} + p^{d-1} \cdot \langle \vec{x}, \vec{y} \rangle \mathbf{G} \Big)^\top \mathbf{s} + \mathbf{z}' \in \mathbb{Z}_q^m.$$

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,

$$s_1(\vec{\mathbf{R}}_{\vec{y}}) \ = \ s_1\Big(\sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \Big) \ \leq \ C \cdot \sqrt{\ell} p \sigma \cdot \sqrt{m} \qquad (11)$$

where, the inequality follows from Lemma 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,

$$|z - z''| \leq |z| + |\mathbf{e}^\top [\mathbf{z}_0^\top | \mathbf{z}'^\top]^\top|$$

$$\leq |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\mathbf{e}^\top \mathbf{z}'\| \tag{12}$$

$$= |z| + \|\mathbf{e}^\top \mathbf{z}_0\| + \|\sum_{i=1}^{\ell} \mathsf{y}_i \cdot \mathbf{e}^\top \mathbf{z}_i\|$$

$$\leq \left(\alpha q + p \cdot s_1(\vec{\mathbf{R}}_{\vec{y}}) \sqrt{m\ell} \alpha' q\right) \omega(\sqrt{\log n}) \tag{13}$$

$$\leq \left(\alpha q + \ell p^2 \sigma m \alpha' q\right) \omega(\sqrt{\log n}) \tag{14}$$

where Eq.(12) follows from the sub-additivity of the square root function $\sqrt{\cdot}$, Eq.(13) follows from the linear structure of subgaussian random variables, Lemma 3, Lemma 5 and the fact that $\vec{\mathsf{y}} \in \mathbb{Z}_p^\ell$, Eq.(14) follows from Eq.(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 Dec 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., $\left(\alpha q + \ell p^2 \sigma m \alpha' q\right) \omega(\sqrt{\log n}) < q/5$. See Lemma 7),
- the gadget matrix $\mathbf{G}$ is well defined (i.e., $m \geq n \lceil \log q \rceil$. See Lemma 5.),
- σ is sufficiently large so that $\mathbf{R}_i$'s are samplable and Theorem 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 5 and Lemma 10),
- 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 4),
- the worst case to average case reduction works (i.e., $\alpha q > 2\sqrt{n}$). (See Theorem 2 and Theorem 3).

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.

4.2 Security Proof

Theorem 5. *The above NIPE scheme with inner product space $\mathbb{Z}_p$ is selectively 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:

$$\mathbf{R}_{\vec{y}} = \sum_{i=1}^{\ell} \mathsf{y}_i \mathbf{R}_i \overset{\text{(i)}}{=} \sum_{i=1}^{\ell} \Big(\sum_{j \in \mathsf{list}} \lambda_j \mathsf{h}_i^{(j)} \Big) \mathbf{R}_i = \sum_{j \in \mathsf{list}} \lambda_j \Big(\sum_{i=1}^{\ell} \mathsf{h}_i^{(j)} \mathbf{R}_i \Big) \overset{\text{(ii)}}{=} \sum_{j \in \mathsf{list}} \lambda_j \mathbf{R}_{\vec{h}^{(j)}} \in \mathbb{Z}^{m \times m},$$

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. (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

$$\Big\{ \mathsf{MPK} = \Big\{ \mathbf{A}, \{\mathbf{B}_i\}_{i \in [\ell]}, \mathbf{u} \Big\}, \quad \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j \in \mathsf{list}}, \quad C^* \Big\},$$

where $C^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, \vec{x}^*, \mathsf{M}_b)$ 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

$$\Big\{ \{\mathbf{B}_i\}_{i\in[\ell]}, \quad \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}} \Big\}. \tag{15}$$

Hence, we aim at proving that the view of Eq.(15) in both games are statistically close to the adversary. More specifically, we compare the following probability of each game:

$$\begin{aligned}
&\Pr\left[\Big\{ \{\mathbf{B}_i\}_{i\in[\ell]}, \ \{\mathbf{R}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}} \Big\} = \Big\{ \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]}, \ \{\widehat{\mathbf{R}}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}} \Big\} \right] \\
=&\underbrace{\Pr\left[\{\mathbf{R}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}} = \{\widehat{\mathbf{R}}_{\vec{h}^{(j)}}\}_{j\in\mathsf{list}} \ \Big| \ \{\mathbf{B}_i\}_{i\in[\ell]} = \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} \right]}_{\text{(A)}} \cdot \underbrace{\Pr\left[\{\mathbf{B}_i\}_{i\in[\ell]} = \{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} \right]}_{\text{(B)}},
\end{aligned}$$

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 $i \in [\ell]$, $\mathbf{AR}_i$ 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 1 and our parameter selections. Concretely, since $\mathbf{B}_i = \mathbf{AR}_i$ and $\mathbf{B}_i = \mathbf{AR}_i - p^{d-1}\cdot x_i^*\mathbf{G}$ 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 1 and the conditions on $\{\widehat{\mathbf{B}}_i\}_{i\in[\ell]} = \{\mathbf{AR}_i\}_{i\in[\ell]}$, 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:

$$\underbrace{[\mathbf{R}_{\vec{h}^{(1)}}|\mathbf{R}_{\vec{h}^{(2)}}|\cdots|\mathbf{R}_{\vec{h}^{(\ell-1)}}]}_{:=\vec{\mathbf{R}}_{\mathsf{sk}}\ \in\mathbb{Z}^{m\times m(\ell-1)}} = \underbrace{[\mathbf{R}_1|\mathbf{R}_2|\cdots|\mathbf{R}_\ell]}_{=\vec{\mathbf{R}}\ \in\mathbb{Z}^{m\times m\ell}} \underbrace{\left[\begin{array}{c|c|c|c|c|c}
\mathsf{h}_1^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_1^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_1^{(\ell-1)}\mathbf{I}_m \\
\mathsf{h}_2^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_2^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_2^{(\ell-1)}\mathbf{I}_m \\
\vdots & & \vdots & \vdots & & \vdots \\
\mathsf{h}_\ell^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_\ell^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(\ell-1)}\mathbf{I}_m
\end{array}\right]}_{:=\mathbf{M}\ \in\mathbb{Z}^{m\ell\times m(\ell-1)}}, \tag{16}$$

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. We show in Lemma 10 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 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 postpone the proof of Lemma 10 to the end so as not to interrupt the proof.

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 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.(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 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

$$\begin{aligned}
\vec{\mathbf{E}}_{\vec{x}^*}\mathbf{M} &= p^{d-1}\cdot\mathbf{E}\cdot[x_1^*\mathbf{I}_m|x_2^*\mathbf{I}_m|\cdots|x_\ell^*\mathbf{I}_m]\cdot\left[\begin{array}{c|c|c|c|c|c}
\mathsf{h}_1^{(1)}\mathbf{I}_m & \dots & \mathsf{h}_1^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_1^{(j^*+1)}\mathbf{I}_m & \dots & \mathsf{h}_1^{(\ell-1)}\mathbf{I}_m \\
\mathsf{h}_2^{(1)}\mathbf{I}_m & \dots & \mathsf{h}_2^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_2^{(j^*+1)}\mathbf{I}_m & \dots & \mathsf{h}_2^{(\ell-1)}\mathbf{I}_m \\
\vdots & & \vdots & \vdots & & \vdots \\
\mathsf{h}_\ell^{(1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(j^*-1)}\mathbf{I}_m & \mathsf{h}_\ell^{(j^*+1)}\mathbf{I}_m & \cdots & \mathsf{h}_\ell^{(\ell-1)}\mathbf{I}_m
\end{array}\right], \\
&= p^{d-1}\cdot\mathbf{E}\cdot[\langle\vec{x}^*,\vec{\mathsf{h}}^{(1)}\rangle\mathbf{I}_m|\cdots|\langle\vec{x}^*,\vec{\mathsf{h}}^{(j^*-1)}\rangle\mathbf{I}_m|\langle\vec{x}^*,\vec{\mathsf{h}}^{(j^*+1)}\rangle\mathbf{I}_m|\cdots|\langle\vec{x}^*,\vec{\mathsf{h}}^{(\ell-1)}\rangle\mathbf{I}_m] \\
&= q\cdot\mathbf{E}\cdot[n_1\mathbf{I}_m|\cdots|n_{j^*-1}\mathbf{I}_m|n_{j^*+1}\mathbf{I}_m|\cdots|n_{\ell-1}\mathbf{I}_m]\in q\mathbb{Z}^{m\times m(\ell-1)},
\end{aligned}$$

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 Sec. 4.1) 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 $\mathbf{V}\mathbf{M} = \mathbf{Z}$ (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:

$$\mathbf{PHQ} = \left[\begin{array}{ccccc}
\star & & \cdots & \star & \star \\
\hline
1 & 0 & \cdots & \cdots & 0 \\
\star & 1 & \ddots & & \vdots \\
\vdots & \star & \ddots & \ddots & \vdots \\
\vdots & & \ddots & 1 & 0 \\
\star & \star & \cdots & \star & 1
\end{array}\right] = \begin{bmatrix}\mathbf{a}^\top \\ \mathbf{U}\end{bmatrix}\in\mathbb{Z}^{\ell\times(\ell-1)}, \tag{17}$$

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

$\mathbf{V} = [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{Q}\mathbf{U}^{-1}\otimes\mathbf{I}_m)]\cdot(\mathbf{P}\otimes\mathbf{I}_m) \in \mathbb{Z}^{m\times m\ell}$ satisfies the above condition, i.e., $\mathbf{V} \in \Lambda^{\perp}(\mathbf{A})^{m\ell}$ and $\mathbf{V}\mathbf{M} = \mathbf{Z}$ (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

$$\begin{aligned}
\mathbf{V}\mathbf{M} &= \Big([\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{Q}\mathbf{U}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)\Big)\cdot(\mathbf{H}\otimes\mathbf{I}_m) \\
&= \Big([\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{Q}\mathbf{U}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)\Big)\cdot\left(\mathbf{P}^{\top}\begin{bmatrix}\mathbf{a}^{\top}\\ \mathbf{U}\end{bmatrix}\mathbf{Q}^{\top}\right)\otimes\mathbf{I}_m \qquad (18)\\
&= [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{Q}\mathbf{U}^{-1}\otimes\mathbf{I}_m)](\mathbf{P}\otimes\mathbf{I}_m)(\mathbf{P}^{\top}\otimes\mathbf{I}_m)\left(\begin{bmatrix}\mathbf{a}^{\top}\mathbf{Q}^{\top}\\ \mathbf{U}\mathbf{Q}^{\top}\end{bmatrix}\otimes\mathbf{I}_m\right) \qquad (19)\\
&= [\mathbf{0}_{m\times m} \mid \mathbf{Z}\cdot(\mathbf{Q}\mathbf{U}^{-1}\otimes\mathbf{I}_m)]\begin{bmatrix}\mathbf{a}^{\top}\mathbf{Q}^{\top}\otimes\mathbf{I}_m\\ \mathbf{U}\mathbf{Q}^{\top}\otimes\mathbf{I}_m\end{bmatrix} \qquad (20)\\
&= \mathbf{Z}, \qquad (21)
\end{aligned}$$

where Eq. (18) follows from Eq. (17), Eq. (19) follows from the fact that $(\mathbf{A}\mathbf{B}\otimes\mathbf{I}_m) = (\mathbf{A}\otimes\mathbf{I}_m)(\mathbf{B}\otimes\mathbf{I}_m)$ and Eq. (20),(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 statistically 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

$$c = p^{d-1}\cdot\Big(\mathbf{u}^{\top}\mathbf{s} + \mathsf{M}_b\lfloor p/2\rceil\Big), \quad \mathbf{c}_0 = \mathbf{A}^{\top}\mathbf{s} + \mathbf{z}_0, \quad (\mathbf{c}_i = (\mathbf{A}\mathbf{R}_i)^{\top}\mathbf{s} + \mathbf{z}_i)_{i\in[\ell]} \qquad (22)$$

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 4, 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

$$C^* = \Big(c = v + \mathsf{M}_b\cdot p^{d-1}\lfloor p/2\rceil, \quad \mathbf{c}_0, \quad (\mathbf{c}_i)_{i\in[\ell]}\Big) \in \mathbb{Z}_q \times \mathbb{Z}_q^m \times (\mathbb{Z}_q^m)^{\ell}, \qquad (23)$$

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.(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 4, 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.(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.(23). We show below in Lemma 11 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. Now, to complete the proof of Theorem 4, it remains to prove the following two lemmas Lemma 10 and 11.

Lemma 10. *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. (16).*

Proof. Here, we use the matrices and vector $\mathbf{P}, \mathbf{Q}, \mathbf{U}, \mathbf{a}$ defined in Eq. (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

$$\mathbf{W}^\top\mathbf{M} = \big((\mathbf{P}^\top\mathbf{w}) \otimes \mathbf{I}_m\big)^\top \cdot \mathbf{M} = \big((\mathbf{w}^\top\mathbf{P}) \otimes \mathbf{I}_m\big) \cdot (\mathbf{H} \otimes \mathbf{I}_m) = \big((\mathbf{w}^\top\mathbf{PHQ}) \cdot \mathbf{Q}^\top\big) \otimes \mathbf{I}_m = \mathbf{0},$$

where we have $\mathbf{QQ}^\top = \mathbf{I}_{\ell-1}$ 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 $\|\mathbf{w}\| \leq 3p^\ell$. Recalling Eq. (17), we have

$$(\mathbf{w}^\top \mathbf{PHQ})^\top = \left[\begin{array}{c|ccccc} a_1 & 1 & u_{1,1} & \cdots & \cdots & u_{1,\ell-1} \\ a_2 & 0 & 1 & u_{2,2} & \cdots & u_{2,\ell-1} \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ a_{\ell-2} & \vdots & & \ddots & 1 & u_{\ell-1,\ell-1} \\ a_{\ell-1} & 0 & \cdots & \cdots & 0 & 1 \end{array}\right] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_{\ell-1} \\ w_\ell \end{bmatrix} = \mathbf{0} \in \mathbb{Z}^{\ell-1}, \tag{24}$$

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. (24), if we set $w_1 = 1$ we can solve for the other $\{w_i\}_{i=2}^{\ell}$ terms recursively as follows:

$$\begin{cases} w_\ell & = -a_{\ell-1} w_1 \\ w_{\ell-1} & = -a_{\ell-2} w_1 - u_{\ell-1,\ell-1} w_{\ell-1} \\ & \vdots \\ w_2 & = -a_1 w_1 - \sum_{i=1}^{\ell-1} u_{1,i} w_{i+1} \end{cases}$$

Since, $w_1 = 1$ and $|a_i|, |u_{i,k}| < p$ for all $i \in [\ell - 1], k \in [i]$, we have $|w_i| \leq \sum_{t=1}^{\ell+1-i} p|w_t| \leq p(p+1)^{\ell+1-i}$. Therefore, we have

$$\mathbf{w}^\top \mathbf{w} = \sum_{i=1}^{\ell} w_i^2 \leq \sum_{i=1}^{\ell} p^2 (p+1)^{2(\ell+1-i)} \leq (p+1)^{2(\ell+2)}.$$

Thus, we conclude that $\sqrt{s_1(\mathbf{W}^\top \mathbf{W})} = \|\mathbf{w}\| \leq (p+1)^{\ell+2}$. □

Lemma 11. *For any PPT adversary $\mathcal{A}$, there exists another PPT adversary $\mathcal{B}$ such that*

$$|\Pr[S_2] - \Pr[S_3]| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{FE.LWE}_{n,m+1,q,\chi}}.$$

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.(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. □

□

5 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.

5.1 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 $\mathsf{sk}_{\vec{y}}$.

$\mathsf{Encrypt}(\mathsf{MPK}, \vec{x}) \to C$: The encryption algorithm takes as input a master public key MPK and attribute vector $\vec{x} \in \mathcal{J}$. It outputs a ciphertext C.

$\mathsf{Decrypt}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, C) \to \langle \vec{x}, \vec{y} \rangle$or $\perp$: The decryption algorithm takes as input the master public key MPK, a private key $\mathsf{sk}_{\vec{y}}$, and a ciphertext C. 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

$$\Pr[\mathsf{Dec}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, \mathsf{Enc}(\mathsf{MPK}, \vec{x}, \mathsf{M})) = \langle \vec{x}, \vec{y} \rangle] = 1 - \mathsf{negl}(\lambda)$$

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 $\mathsf{Enc}(\mathsf{MPK}, \vec{x})$.

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 $C^* \leftarrow \mathsf{Enc}(\mathsf{MPK}, \vec{x}_b^*)$ 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.

5.2 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 $C \leftarrow \mathsf{Enc}(\mathsf{MPK}, \mathsf{M} \cdot \vec{x})$ and outputs C.

$\mathsf{Dec}'(\mathsf{MPK}, (\vec{y}, \mathsf{sk}_{\vec{y}}), (\vec{x}, C))$: To decrypt a ciphertext C 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 $z = \mathsf{Dec}(\mathsf{MPK}, \mathsf{sk}_{\vec{y}}, C)$. 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. *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 $\mathcal{B}$. $\mathcal{B}$ then passes 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^*. $\mathcal{B}$ then passes 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 7. *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. □

5.3 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.

Here, we compare our direct construction in Section 4 with the scheme obtained via the above conversion. 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 $(m + \ell)\ell_M$. 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 $\ell < m, \lambda$, this condition 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.

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 5.2. 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.

Acknowledgement. The first author was partially supported by JST CREST Grant Number JPMJCR1302 and JSPS KAKENHI Grant Number 17J05603. The second author was supported by JST CREST Grant Number JPMJCR1688 and JSPS KAKENHI Grant Number 16K16068.

References

[ABB10] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h) ibe in the standard model. In *EUROCRYPT*, pages 553–572. Springer, 2010. 4, 8, 9, 17

[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. 3, 5, 31

[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. 5, 28

[ABS17] Miguel Ambrona, Gilles Barthe, and Benedikt Schmidt. Generic transformations of predicate encodings: Constructions and applications. In *Crypto*, pages 36–66. Springer, 2017. 2

[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. 10

[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. 4

[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. 2

[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. 1, 2

[ALS16] Shweta Agrawal, Benoît Libert, and Damien Stehlé. Fully secure functional encryption for inner products, from standard assumptions. In *Crypto 2016*, volume 9816, pages 333–362. Springer, 2016. 3, 5, 7, 8, 31, 32

[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. 2

[BCH86] Paul W Beame, Stephen A Cook, and H James Hoover. Log depth circuits for division and related problems. *SIAM Journal on Computing*, 15(4):994–1003, 1986. 2

[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. 4, 9, 38, 39

[BGG$^+$14] 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. 2, 4

[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. 9, 10

[BW07] Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In *TCC*, pages 535–554. Springer, 2007. 2

[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, 2016. 2

[CW14] Jie Chen and Hoeteck Wee. Doubly spatial encryption from DBDH. *Theor. Comput. Sci.*, 543:79–89, 2014. 2

[DG17] Nico Döttling and Sanjam Garg. Identity-based encryption from the diffie-hellman assumption. In *CRYPTO*, pages 537–569. Springer, 2017. 3, 6

[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. 1, 2

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. ACM, 2008. 8, 9, 40

[GV15] Sergey Gorbunov and Dhinakaran Vinayagamurthy. Riding on asymmetry: Efficient ABE for branching programs. In *ASIACRYPT*, pages 550–574. Springer, 2015. 2, 4

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554. ACM, 2013. 1, 2, 4

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. *EUROCRYPT*, pages 146–162, 2008. 2

[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. 8

[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. 1

[LW11] Allison B. Lewko and Brent Waters. Unbounded HIBE and attribute-based encryption. In *Eurocrypt*, pages 547–567, 2011. 1

[MP12] Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718. Springer, 2012. 4, 8, 9, 10, 35

[MR04] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. In *Symposium on Foundations of Computer Science–FOCS*, pages 372–381, 2004. 38

[OSW07] Rafail Ostrovsky, Amit Sahai, and Brent Waters. Attribute-based encryption with non-monotonic access structures. In *CCS*, pages 195–203. ACM, 2007. 2

[OT10] Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In *CRYPTO*, pages 191–208, 2010. 1, 2

[OT15] Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. *Designs, Codes and Cryptography*, 77(2-3):725–771, 2015. 2

[Pei09] Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. ACM, 2009. 10

[PRSD17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-lwe for any ring and modulus. In *STOC*, pages 461–473. ACM, 2017. 10

[PVW08] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. A framework for efficient and composable oblivious transfer. In *Crypto*, pages 554–571. Springer, 2008. 17

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93. ACM Press, 2005. 9, 10

[SW05] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473. Springer, 2005. 1

[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. 2

[Yam16] Shota Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In *EUROCRYPT*, pages 32–62. Springer, 2016. 17

A Omitted Proof from Sec. 2.2

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 $\mathbf{G}\mathbf{z} = t^{-1}\mathbf{u} \mod q$ 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

$$\begin{aligned} [\mathbf{A}|\mathbf{A}\mathbf{R} + p^{d-1} \cdot t \cdot \mathbf{G}]\mathbf{e} &= [\mathbf{A}|\mathbf{A}\mathbf{R} + p^{d-1} \cdot t \cdot \mathbf{G}] \begin{bmatrix} -\mathbf{R} \\ \mathbf{I}_m \end{bmatrix} \mathbf{z} \\ &= p^{d-1} \cdot t \cdot \mathbf{G}\mathbf{z} \\ &= p^{d-1}\mathbf{u} \mod q. \end{aligned}$$

Finally, we have $\|\mathbf{e}\| \leq (s_1(\mathbf{R})+1)\|z\| \leq s_1(\mathbf{R})\sqrt{m}\cdot\omega(\sqrt{\log n})$, since we have $\|z\| \leq \sqrt{m}\omega(\sqrt{\log n})$ from Lemma 3 and Lemma 5. This completes the proof. □

B A Note on Sum of Discrete Gaussians

In this section, we provide some discussions on the main tool of our paper — *multi-dimensional lattices*. We believe the new definitions and developed techniques to be of interest to applications elsewhere.

B.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 supplemental materials 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 $s_{\max}(\mathbf{R})$.

B.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[7] $\phi : \mathbb{R}^{m\times\ell} \to \mathbb{R}^{m\ell}$ 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,

$$\begin{aligned}
\bar{\rho}_{\sqrt{\Sigma}}(\mathbf{X}) &= \exp(-\pi \cdot \mathrm{tr}(\mathbf{X}\Sigma^{-1}\mathbf{X}^\top)) \\
&= \exp\big(-\pi \cdot \phi(\mathbf{X})^\top(\Sigma^{-1} \otimes \mathbf{I}_m)\phi(\mathbf{X})\big) && (25)\\
&= \exp\big(-\pi \cdot \phi(\mathbf{X})^\top(\Sigma \otimes \mathbf{I}_m)^{-1}\phi(\mathbf{X})\big) && (26)\\
&= \rho_{\sqrt{\Sigma\otimes\mathbf{I}_m}}(\phi(\mathbf{X})), && (27)
\end{aligned}$$

where Eq.(25) follows from the second formula, Eq.(26) follows from $\mathbf{A}^{-1}\otimes\mathbf{B}^{-1} = (\mathbf{A}\otimes\mathbf{B})^{-1}$, and Eq.(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 2. *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 $\rho_{1/s}(\Lambda^*\backslash\{\mathbf{0}\}) \leq \epsilon$. Furthermore, let $\Sigma > \mathbf{0}$ be any symmetric positive-definite matrix in $\mathbb{R}^{m\times m}$. We say $\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$ if $\rho_{\sqrt{\Sigma^{-1}}}(\Lambda^*\backslash\{\mathbf{0}\}) \leq \epsilon$.*

It is informative to observe that, if $\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$, then $\sqrt{\lambda_{\mathsf{max}}} \geq \eta_\epsilon(\Lambda)$ where λ_{max} denotes the largest eigenvalue of Σ. Equivalently, since for symmetric positive-definite matrices eigenvalues equal their singular values, $\sqrt{s_{\mathsf{max}}} \geq \eta_\epsilon(\Lambda)$ where s_{max} denotes the largest singular value. On the other hand, if $\sqrt{\lambda_{\mathsf{min}}} = \sqrt{s_{\mathsf{min}}} \geq \eta_\epsilon(\Lambda)$, then $\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$ 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.

[7] In linear algebra, this isomorphism is sometimes called the *vectorization* of matrices.

Definition 3. *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 12. *[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 13. *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, $\eta_{\epsilon'}(\Lambda) \geq \bar{\eta}_\epsilon(\Lambda^\ell)$. □

B.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 8 (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 $\mathbf{XM}$ is statistically close to $D_{\bar{\Lambda}\mathbf{M}+\mathbf{TM},\sqrt{\mathbf{M}^\top\Sigma\mathbf{M}}}$.

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,

$$\begin{aligned} \Pr[\mathbf{XM} = \mathbf{V}] = \Pr[\mathbf{X} \in S_{\mathbf{V}}] &= \sum_{\mathbf{U} \in L} \Pr[\mathbf{X} = \mathbf{X}_0 + \mathbf{UW}^\top] \\ &= \frac{1}{\bar{\rho}_{\sqrt{\Sigma}}(\bar{\Lambda} + \mathbf{T})} \sum_{\mathbf{U} \in L} \exp\left(-\pi \cdot \operatorname{tr}\left((\mathbf{X}_0 + \mathbf{UW}^\top)\Sigma^{-1}(\mathbf{X}_0 + \mathbf{UW}^\top)^\top\right)\right) \qquad (28) \end{aligned}$$

To properly decouple the terms in $\operatorname{tr}(\cdot)$, we use the following fact on linear algebra.

Fact 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.(28) can be expressed as follows:

$$\begin{aligned} &\operatorname{tr}\left((\mathbf{X}_0 + \mathbf{UW}^\top)\Sigma^{-1}(\mathbf{X}_0 + \mathbf{UW}^\top)^\top\right) \\ =&\operatorname{tr}\left(\mathbf{UW}^\top \Sigma^{-1} \mathbf{WU}^\top\right) + 2 \cdot \operatorname{tr}\left(\mathbf{X}_0 \Sigma^{-1} \mathbf{WU}^\top\right) + \operatorname{tr}\left(\mathbf{X}_0 \Sigma^{-1} \mathbf{X}_0^\top\right) \\ =&\operatorname{tr}\left((\mathbf{U} + \mathbf{Y})\left(\mathbf{W}^\top \Sigma^{-1} \mathbf{W}\right)(\mathbf{U} + \mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{X}_0 \Sigma^{-1} \mathbf{X}_0^\top\right) - \operatorname{tr}\left(\mathbf{Y}\left(\mathbf{W}^\top \Sigma^{-1} \mathbf{W}\right)\mathbf{Y}^\top\right) \qquad (29) \\ =&\operatorname{tr}\left((\mathbf{U} + \mathbf{Y})\left(\mathbf{W}^\top \Sigma^{-1} \mathbf{W}\right)(\mathbf{U} + \mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{X}_0 \mathbf{M}\left(\mathbf{M}^\top \Sigma \mathbf{M}\right)^{-1} \mathbf{M}^\top \mathbf{X}_0^\top\right) \qquad (30) \\ =&\operatorname{tr}\left((\mathbf{U} + \mathbf{Y})\left(\mathbf{W}^\top \Sigma^{-1} \mathbf{W}\right)(\mathbf{U} + \mathbf{Y})^\top\right) + \operatorname{tr}\left(\mathbf{V}\left(\mathbf{M}^\top \Sigma \mathbf{M}\right)^{-1} \mathbf{V}^\top\right) \qquad (31) \end{aligned}$$

where in Eq.(29) we substitute $\mathbf{Y} = \mathbf{X}_0 \Sigma^{-1} \mathbf{W}\left(\mathbf{W}^\top \Sigma^{-1} \mathbf{W}\right)^{-1}$, in Eq.(30) we use Fact 1, and in Eq.(31) we use the equality $\mathbf{V} = \mathbf{X}_0 \mathbf{M}$. Then plugging Eq.(31) back in Eq.(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 12, 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 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 8 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

$$L := \{\mathbf{U} \in \mathbb{R}^{m\times t} \mid \mathbf{UW}^\top \in \Lambda^\perp(\mathbf{A})^\ell\}.$$

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 13 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. □