Efficient Inner Product Encryption from LWE for fine-grained access control

No Institute Given

Abstract. Inner product encryption (IPE) introduced by Katz, Sahai, and Waters [KSW08] is a generalization of identity-based encryption in which ciphertexts and secret keys are associated to a vectors in some finite field. In an IPE scheme, a ciphertext can only be decrypted by a secret key if the vector associated with the latter is orthogonal to that of the ciphertext, which provides fine-grained access control and has attractive applications

In this work, we propose a lattice-based inner product encryption(IPE) scheme whose security follows from the $Learning\ with\ errors$(LWE) assumption, which has a compact public-key and cipertexts (with size similar to known lattice-based PKE schemes). This improves over previous IPE schemes from lattices, which have a public key and cipertexts that grow at least linearly with the length of the predicate vectors.

Additionally, we construct a hierarchical IPE (HIPE) scheme and propose a fuzzy IBE scheme by using our techniques. Our HIPE scheme and fuzzy IBE scheme are more efficient than prior works in terms of public key and cipertexts size.

Keywords: predicate encryption, (hierarchical) inner-product encryption, learning with errors, fuzzy identity-based encryption.

1 Introduction

As the Internet and applications evolve, more complex types of data will be stored in distributed settings. These will present a different set of demands for the security of stored data. In many cases, the sender want access to encrypted data to be governed by more complex rules; the sender may no longer intend the encrypted data for any particular user, but might instead want to enable decryption by any user satisfying a certain sender-specified policy. Traditional encryption techniques supporting limited access control can not always meet the requirements. $predicate\ encryption$(PE) [BW07], [KSW08] is an emerging asymmetric encryption allowing fine-grained access control over encrypted data. In PE systems, each ciphertext c is associated with an attribute a and each secret key s is associated with predicate f. A user holding the key s can decrypt c if and only if $f(a) = 1$.

In this paper, we focus on the notation of inner-product encryption(IPE), introduced by Katz, Sahai, and Waters [KSW08], which is a PE scheme that supports the inner-product predicate: attributes a and predicates f are expressed as vectors $\overrightarrow{v}_a$ and $\overrightarrow{w}_f$. We say $f(a) = 1$ if and only if $\langle \overrightarrow{v}_a, \overrightarrow{w}_f \rangle = 0$. As point out in [KSW08], inner-product predicates can also support conjunction, subset and range queries on encrypted date[BW07] as well as disjunctions, polynomial evaluation, and $\mathbf{CNF}$ and $\mathbf{DNF}$ formulas [KSW08].

In recent years, there were a number of IPE schemes [KSW08],[OT09],[LOS$^+$10], [OT10],[AL10],[Par11], [OT11], [OT12] have been proposed, the security of these schemes is based on composite-number or prime order groups. while assumption such as these can often be shown to hold in a suitable generic group model, to obtain more confidence in security; what's more, large-scale quantum computers will able to efficiently solve these problems by Shor's quantum algorithm [Sho94], posing a serious threat to cryptography based on these assumption. Therefore, we would like to build IPE scheme based on computational problems such as lattice-based hardness problems, whose complexity is better understood and considered to be quantum attack-resistant.

Agrawal, Freeman, and Vaikuntanathan [AFV11] proposed the first IPE scheme based on the LWE assumption, and Xagawa [Xag13] improved the efficiency of [AFV11]'s scheme. While the scheme of [Xag13] has public parameters of size $O(\ell n^2 \lg^2 q)$ and ciphertexts of size $O(\ell n \lg^2 q)$, where ℓ is the length of predicate vector, n is the security parameter, and q is the modulus. which still makes the scheme impractical.

1.1 Our Contributions

In this work, we construct an inner-product encryption scheme (IPE) from the standard Learning with Error [Reg05] assumption with a compact public key of size $2nm \log q$ and ciphertexts of size $(2m + 1) \log q$. This asymptotically matches the public key space and ciptertext space complexity of known lattice-based PKE schemes. and points us toward real-world-efficient implementations of lattice-based IPE. We emphasize that prior lattice-based IPE schemes
[AFV11], [Xag13] have public keys and cipertexts whose size grow at least linearly with ℓ, the length of predicate vectors(attribute vector), whereas our IPE's PK and cipertexts size are (at least) asymptotically sublinear in the predicate vectors length. We provide a detail comparison with several recent work in section 1.3.

Theorem 1 (Main). *Under the standard Learning with Error (LWE) assumption, there is an inner-product encryption (IPE) scheme with weakly attribute hiding security supporting predicates and attributes of length $\ell = O(\log^2 n)$, where*

- *the modulus q is a prime of size polynomial in the security parameter n,*
- *cipertexts consist of a vector in $\mathbb{Z}_q^{2m+1}$, and*
- *the public key consists of two matrices in $\mathbb{Z}_q^{n \times m}$ and one vector in $\mathbb{Z}_q^n$.*

We note that we can extend the attribute domain by assuming an exponentially-secure collision resistant hash function from $\{0,1\}^n \to \{0,1\}^{\log^2 n}$.

In addition, We have two side contributions: One is an extension of hierarchical inner-product encryption (HIPE)
[OT09], which implies spatial encryption(SE) [BH08]. We apply our techniques to drastically improve the existing HIPE scheme from lattices [ADM12],[Xag13]. The other is a fuzzy IBE (FIBE) scheme over a small universe $\{0,1\}$ from IPE under the LWE assumption with compact public key size and cipertexts size, whereas the existing fuzzy IBE schemes from lattices have public key whose size grows linearly with the length of identities ℓ [ABV$^+$12],[Xag13].

1.2 Our Techniques

Our high-level approach to compact inner-product encryption from LWE begins by revisiting the lattice based IPE scheme proposed by Agrawal, Freeman, and Vaikuntanathan [AFV11] and Xagawa [Xag13].

In the [AFV11] IPE scheme, the encryption lattice for ciphertext-attribute vector $\boldsymbol{w} \in \mathbb{Z}_q^\ell$ is defined as

$$\Lambda_{\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\mathbf{A}_1 + w_1\mathbf{B}|...|\mathbf{A}_\ell + w_\ell\mathbf{B}).$$

The cipertext is a vector $\boldsymbol{c} = (\boldsymbol{c}_0, ..., \boldsymbol{c}_\ell) \in \mathbb{Z}_q^{m\times(\ell+1)}$ close to $\Lambda_{\boldsymbol{w}}$.

They define the mapping $F_{\boldsymbol{v}} : \mathbb{Z}_q^{m\times(\ell+1)} \to \mathbb{Z}_q^{2m}$ as

$$F_{\boldsymbol{v}}(\boldsymbol{c}_0, \boldsymbol{c}_1..., \boldsymbol{c}_\ell) = (\boldsymbol{c}_0, \Sigma_{i=1}^{\ell} v_i\boldsymbol{c}_i) \in \mathbb{Z}_q^{2m}$$

for decryption. If $\boldsymbol{v}$ is a short vector, e.g., $\boldsymbol{v} \in \{0,1\}^\ell \subset \mathbb{Z}_q^\ell$, then $F_{\boldsymbol{v}}(\boldsymbol{c})$ is a vector close to the lattice

$$\Lambda_{\boldsymbol{v},\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\Sigma_{i=1}^{\ell} v_i(\mathbf{A}_i + w_i\mathbf{B})) = \Lambda_q(\mathbf{A}_0|\Sigma_{i=1}^{\ell} v_i\mathbf{A}_i + (\boldsymbol{w}\boldsymbol{v})\mathbf{B}).$$

So if $\boldsymbol{w}^T\boldsymbol{v} = 0$ then the masking term $(\boldsymbol{w}^T\boldsymbol{v})\mathbf{B}$ vanishes. the secret key for $\boldsymbol{v} \in \mathbb{Z}_q^\ell$ is defined as a short basis of $\Lambda_q^\perp(\mathbf{A}_0|\Sigma_{i=1}^{\ell} v_i\mathbf{A}_i)$.

In the [Xag13] IPE scheme, it changed the domain of attributes $\boldsymbol{w}$ from $\mathbb{Z}_q^\ell$ to $GF(q^n)^\ell$, while $\boldsymbol{v}$'s domain is the same as the original $\mathbb{Z}_q^\ell$. It also changed the way of encoding the attribute vectors $\boldsymbol{w}$, he defined a mapping $H : GF(q^n) \to \mathbb{Z}_q^{n\times n}$, and naturally defined $\Lambda_{\boldsymbol{w}}$ for $\boldsymbol{w} = (w_1, ..., w_\ell)^T \in GF(q^n)^\ell$ as

$$\Lambda_{\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\mathbf{A}_1 + H(w_1)\mathbf{G}|...|\mathbf{A}_\ell + H(w_\ell)\mathbf{G}).$$

For a predicate vector $\boldsymbol{v} \in \mathbb{Z}_q^\ell$, the decryption mapping $F_{\boldsymbol{v}}^{'}(\boldsymbol{c}_0, ..., \boldsymbol{c}_\ell) = (\boldsymbol{c}_0, \Sigma_{i=1}^{\ell} H(v_i)\cdot \boldsymbol{c}_i))$ is close to the lattice

$$\Lambda_{\boldsymbol{v},\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\Sigma_{i=1}^{\ell} H(v_i)\cdot\mathbf{A}_i + H(\boldsymbol{w}^T\boldsymbol{v})\mathbf{G})$$

if $\boldsymbol{w}^T\boldsymbol{v} = 0$ then $H(\boldsymbol{w}^T\boldsymbol{v}) = 0$. Using the property of the structured matrix $\mathbf{G}$, this scheme doesn't need to use the binary decomposition technique as [AFV11], but the size of public key and cipertexts still grow with the length of attribute/predicate vectors.

Our new encoding techniques. Here, we present our new encoding technical. We draw a connection between the public of our IPE scheme and the GSW-FHE [GSW13]. To be more exact, we recall the FHE scheme proposed by Hiromasa, Abe, and Okamoto [HAO15], which shows that by applying SIMD (Single Instruction Multiple Data) message-packing techniques to the GSW-FHE, the result scheme is $matrix - multiplication - homomorphic$.

The cipertexts in [HAO15] have the form $\mathbf{AR} + \mathbf{MG}$, where $\mathbf{A} \in \mathbb{Z}_q^{n\times m}, \mathbf{R} \in \mathbb{Z}_q^{m\times m}$, and $\mathbf{M} \in \mathbb{Z}_q^{n\times n}$ contains a message vector μ along the diagonal, and where $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ is the "gadget matrix" that first introduced in [MP12]. The key point is that there is an efficient computable function $\mathbf{G}^{-1}$, so that

1. The multiplication homomorphism holds ($\mathsf{ct}_1 \cdot \mathbf{G}^{-1}(\mathsf{ct}_2) = \mathsf{ct}_\times$), and
2. $\mathbf{G}^{-1}(\mathbf{AR} + \mathbf{MG})$ is a matrix in $\{0,1\}^{m\times m}$, and has small norm.

The scheme is also naturally equipped with a matrix-multiplication-by-a-constant operation of the following form:

$$(\mathbf{AR} + \mathbf{MG}) \cdot \mathbf{G}^{-1}(\mathbf{CG}) = \mathbf{AR} \cdot small + \mathbf{MCG},$$

this leads us to consider a new encoding technique where the public key matrices $\mathbf{A}_i \in \mathbb{Z}_q^{n\times m}$ are "multiplied by a constant" $\mathbf{W}_i \in \mathbb{Z}_q^{n\times n}$, where the matrix $\mathbf{W}_i$ contains the i-th attribute coordinate w_i on each coordinate of its diagonal. So the encryption lattice is defined as

$$\Lambda_{\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\Sigma_i(\mathbf{A}_i + \mathbf{W}_i\mathbf{G})),$$

and define the mapping $F_{\boldsymbol{v}} : \mathbb{Z}_q^{m\times(i+1)} \to \mathbb{Z}_q^{2m}$ as

$$F_{\boldsymbol{v}}(\boldsymbol{c}_0, \boldsymbol{c}_1 ..., \boldsymbol{c}_i) = (\boldsymbol{c}_0, \Sigma_i \boldsymbol{c}_i \mathbf{G}^{-1}(\mathbf{V}_i\mathbf{G})) \in \mathbb{Z}_q^{2m},$$

where the $\mathbf{V}_i$ is the matrix contains the i-th predicate coordinate v_i on each coordinate of its diagonal. Then, $F_{\boldsymbol{v}}(\boldsymbol{c})$ is close to the lattice

$$\Lambda_{\boldsymbol{v},\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\Sigma_i(\mathbf{G}^{-1}(\mathbf{V}_i\mathbf{G})\mathbf{A}_i + \mathbf{V}_i\mathbf{W}_i\mathbf{G})),$$

if $\boldsymbol{w}^T\boldsymbol{v} = 0$, $\mathbf{V}_i\mathbf{W}_i\mathbf{G}$ vanishes.

In order to achieve the public key of two matrices, we need to shrink the matrices $\mathbf{W}_i$ into one matrix. So we extend the "gadget matrix" $\mathbf{G}$ and function $\mathbf{G}^{-1}$ to other powers, i.e., $\mathbf{G}_{n\ell,\ell',m}$ (which can be padded by $\mathbf{G}_{n\ell,\ell'}$ with zerro) and $\mathbf{G}^{-1}_{n\ell,\ell',m}$. Then we pad all the matrices $\mathbf{W}_i$ into one matrix $\mathbf{W}^{'} = [\mathbf{W}_1, ..., \mathbf{W}_\ell]$, and set $\mathbf{W} = \mathbf{W}^{'} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}^{n\times m}$, whose rows and columns are equal to $\mathbf{A}$, so we change $\Lambda_{\boldsymbol{w}}$ into

$$\Lambda_{\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\mathbf{A}_1 + \mathbf{W}).$$

Similarly, we define $\mathbf{V}^{'} = [\mathbf{V}_1, ..., \mathbf{V}_\ell]^T$, and set $\mathbf{V} = \mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{V}^{'}\mathbf{G}_{n,2,m}) \in [\ell^{'}]^{m\times m}$. Then $F_{\boldsymbol{v}}$ becomes

$$F_{\boldsymbol{v}}(\boldsymbol{c}) = (\boldsymbol{c}_0, \boldsymbol{c}_1\mathbf{V}),$$

so the lattice $\Lambda_{\boldsymbol{v},\boldsymbol{w}}$ becomes

$$\Lambda_{\boldsymbol{v},\boldsymbol{w}} = \Lambda_q(\mathbf{A}_0|\mathbf{A}_1\mathbf{V} + \mathbf{WV}),$$

it's easy to check that $\mathbf{WV} = \mathbf{0}$ if $\boldsymbol{w}^T\boldsymbol{v} = 0$, and the norm of $\mathbf{V}$ is small. The secret key is a short basis of lattice $\Lambda_q^{\perp}(\mathbf{A}_0|\mathbf{A}_1\mathbf{V})$. Therefore, we yield a compact IPE from LWE.

1.3 Comparison with prior work

In this section, we first provide a detailed comparison with the first lattice-based IPE construction [AFV11] and the follow-up work [Xag13], then we compare the HIPE scheme and FIBE scheme with prior works.

We start from the scheme [AFV11],for the use of the binary decomposition technique, there are $O(\ell \log q)$ matrices in public parameters in their construction. Then, [Xag13] improved the scheme by removing the binary decomposition technique, resulting a scheme with $O(\ell)$ matrices in public parameters.

In the following Table 1, we compare the current state-of-the-arts constructions that are plain LWE-based and achieve weakly attribute hiding security with polynomial key queries-the work [AFV11,Xag13] and this work. We set the length of the predicate/sttribute space to be $\log^2 \lambda$, and security level to be super-poly. We also

Scheme Type	Security Level	Predicate/Attribute Length	# matrices in mpk
[AFV11]	super-poly	$\log^2 \lambda$	$\log^2 \lambda \cdot \log q$
[Xag13]	super-poly	$\log^2 \lambda$	$\log^2 \lambda$
This paper	super-poly	$\log^2 \lambda$	2

Table 1. Comparison of our IPE and prior schemes

note that, assuming the exiting of a collision resistant hash function from $\{0,1\}^\lambda$ to $\{0,1\}^{\log^2 \lambda}$, we can extend the predicate length to λ, then the number of the matrices in mpk in [AFV11] is $\log^2 \lambda \cdot \log q$, and $\log^2 \lambda$ in [Xag13], and 2 in our scheme.

Next, we compare our HIPE scheme with [ADM12,Xag13]. The number of matrices in [ADM12]'s public key is $O(\ell d \log q)$, and [Xag13]'s is $O(d + \ell \log q)$, whereas our scheme enjoys compact a public key with just 2 matrices. We omit the detail, which can be obtained in the full version of the paper.

Finally, we sketch the comparison of our FIBE scheme with some related works. The number of public matrices in [ABV$^+$12] is 2ℓ, and 2ℓ trapdoor matrices in msk. [Xag13] proposed a FIBE scheme with $\ell + 1$ public matrices and one trapdoor matrix. However, in our construction, these are 2 and 1 respectively. We describe concrete schemes in full version paper.

2 Preliminaries

Notation. Let λ be the security parameter, and let PPT denote probabilistic polynomial time. We use bold uppercase letters to denote matrices $\mathbf{M}$, and bold lowercase letters to denote vectors $\boldsymbol{v}$. We write $\widetilde{\mathbf{M}}$ to denote the Gram-Schmidt orthogonalization of $\mathbf{M}$. We write $[n]$ to denote the set $\{1, ..., n\}$, and $|\boldsymbol{t}|$ to denote the number of bits in the string $\boldsymbol{t}$. We denote the i-th bit $\boldsymbol{s}$ by $\boldsymbol{s}[i]$. We say a function $\mathsf{negl}(\cdot) : \mathbb{N} \to (0,1)$ is negligible, if for every constant $c \in \mathbb{N}$, $\mathsf{negl}(n) < n^{-c}$ for sufficiently large n.

2.1 Inner Product Encryption

We recall the syntax and security definition of *inner product encryption* (IPE) [KSW08]. IPE can be regarded as a generalization of predicate encryption. An IPE scheme Π consists of the 4-tuple $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec})$ algorithms with details as follows:

$\mathsf{Setup}(1^\lambda)$: On input the security parameter λ, the setup algorithm outputs public parameters pp and master secret key msk.
$\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{v})$: On input the master secret key msk and a predicate vector $\boldsymbol{v}$, the key generation algorithm outputs a secret key $\mathsf{sk}_{\boldsymbol{v}}$ for vector $\boldsymbol{v}$.
$\mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}, \mu)$: On input the public parameter pp and an attribute/message pair $(\boldsymbol{w}, \mu)$, it outputs a ciphertext $\mathsf{ct}_{\boldsymbol{w}}$.
$\mathsf{Dec}(\mathsf{sk}_{\boldsymbol{v}}, c_{\boldsymbol{w}})$: On input the secret key $\mathsf{sk}_{\boldsymbol{v}}$ and a ciphertext $c_{\boldsymbol{w}}$, it outputs the corresponding plaintext μ if $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$; otherwise, it outputs $\perp$.

Correctness. We say the IPE scheme described above is correct, if for any $(\mathsf{msk}, \mathsf{pp}) \leftarrow \mathsf{Setup}(1^\lambda)$, any message μ, predicate vector $\boldsymbol{v}$, and any attribute vector $\boldsymbol{w}$ such that $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$, we have $\mathsf{Dec}(\mathsf{sk}_{\boldsymbol{v}}, \mathsf{ct}_{\boldsymbol{w}}) = \mu$, where $\mathsf{sk}_{\boldsymbol{w}} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{v})$ and $\mathsf{ct}_{\boldsymbol{v}} \leftarrow \mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}, \mu)$.

Security. For the security definition of IPE, we use the following experiment to describe it. Formally, for any PPT adversary $\mathcal{A}$, we consider the experiment $\mathbf{Expt}_{\mathcal{A}}^{\mathsf{IPE}}(1^\lambda)$:

- **Setup**: An adversary $\mathcal{A}$ outputs two attribute vectors $\boldsymbol{w}_0, \boldsymbol{w}_1$, A challenger runs the $\mathsf{Setup}(1^\lambda)$ algorithm, and sends the master public key pp to the adversary.
- **Query Phase I**: Proceeding adaptively, the adversary $\mathcal{A}$ queries a sequence of predicate vectors $(\boldsymbol{v}_1, ..., \boldsymbol{v}_m)$ subject to the restriction that $\langle \boldsymbol{v}_i, \boldsymbol{w}_0 \rangle \neq 0$ and $\langle \boldsymbol{v}_i, \boldsymbol{w}_1 \rangle \neq 0$. On the i-th query, the challenger runs $\mathsf{KeyGen}(\mathsf{msk}, \boldsymbol{v}_i)$, and sends the result $\mathsf{sk}_{\boldsymbol{v}_i}$ to $\mathcal{A}$.
- **Challenge**: Once adversary $\mathcal{A}$ decides that Query Phase I is over, it outputs two length-equal messages (μ_0^*, μ_1^*) and send them to challenger. In response, the challenger selects random $b \in \{0, 1\}$, and sends the ciphertext $\mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}_b, \mu_b^*)$ to $\mathcal{A}$.
- **Query Phase II**: Adversary $\mathcal{A}$ continues to issue identity queries $(\boldsymbol{v}_{m+1}, ..., \boldsymbol{v}_n)$ adaptively, under the restriction that $\langle \boldsymbol{v}_i, \boldsymbol{w}_0 \rangle \neq 0$ and $\langle \boldsymbol{v}_i, \boldsymbol{w}_1 \rangle \neq 0$. The challenger responds by issuing keys $\mathsf{sk}_{\boldsymbol{v}_i}$ as in Query Phase I.
- **Guess**: Adversary $\mathcal{A}$ outputs a guess $b' \in \{0, 1\}$.

We define the advantage of adversary $\mathcal{A}$ in attacking an IPE scheme Π as:

$$\mathbf{Adv}_{\mathcal{A}}(1^\lambda) = \left| \Pr[b = b'] - \frac{1}{2} \right|,$$

where the probability is over the randomness of the challenger and adversary.

Definition 1. *We say an IPE scheme Π is weakly attribute hiding against chosen-plaintext attacks in selective attribute setting, if for all* PPT *adversaries $\mathcal{A}$, we have*

$$\mathbf{Adv}_{\mathcal{A}}(1^\lambda) \leq \mathsf{negl}(\lambda).$$

2.2 Lattice Background

A full-rank m-dimensional integer lattice $\Lambda \subset \mathbb{Z}^m$ is a discrete additive subgroup whose linear span is $\mathbb{R}^m$. The basis of Λ is a linearly independent set of vectors whose linear combinations are exactly Λ. Every integer lattice is generated as the $\mathbb{Z}$-linear combination of linearly independent vectors $\mathbf{B} = \{\boldsymbol{b}_1, ..., \boldsymbol{b}_m\} \subset \mathbb{Z}^m$. For a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, we define the "$q$-ary" integer lattices:

$$\Lambda_q^{\perp} = \{\boldsymbol{e} \in \mathbb{Z}^m | \mathbf{A}\boldsymbol{e} = \mathbf{0} \bmod q\}, \qquad \Lambda_q^{\mathbf{u}} = \{\boldsymbol{e} \in \mathbb{Z}^m | \mathbf{A}\boldsymbol{e} = \boldsymbol{u} \bmod q\}$$

It is obvious that $\Lambda_q^{\boldsymbol{u}}$ is a coset of $\Lambda_q^{\perp}$.

Let Λ be a discrete subset of $\mathbb{Z}^m$. For any vector $\boldsymbol{c} \in \mathbb{R}^m$, and any positive parameter $\sigma \in \mathbb{R}$, let $\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x}) = \exp(-\pi||\boldsymbol{x} - \boldsymbol{c}||^2/\sigma^2)$ be the Gaussian function on $\mathbb{R}^m$ with center $\boldsymbol{c}$ and parameter σ. Next, we let $\rho_{\sigma,\boldsymbol{c}}(\Lambda) = \sum_{\boldsymbol{x} \in \Lambda} \rho_{\sigma,\boldsymbol{c}}(\boldsymbol{x})$ be the discrete integral of $\rho_{\sigma,\boldsymbol{x}}$ over Λ, and let $\mathcal{D}_{\Lambda,\sigma,\boldsymbol{c}}(\boldsymbol{y}) := \frac{\rho_{\sigma,\boldsymbol{c}}(\boldsymbol{y})}{\rho_{\sigma,\boldsymbol{c}}(\Lambda)}$. We abbreviate this as $\mathcal{D}_{\Lambda,\sigma}$ when $\boldsymbol{c} = \mathbf{0}$.

Let S^m denote the set of vectors in $\mathbb{R}^{m+1}$ whose length is 1. Then the norm of a matrix $\mathbf{R} \in \mathbb{R}^{m \times m}$ is defined to be $\mathsf{sup}_{\boldsymbol{x} \in S^m} ||\mathbf{R}\boldsymbol{x}||$. Then we have the following lemma, which bounds the norm for some specified distributions.

Lemma 1 ([ABB10]). *Regarding the norm defined above, the following bounds hold:*

- *Let* $\mathbf{R} \in \{-1, 1\}^{m \times m}$ *be chosen at random, then we have* $\mathsf{Pr}[||\mathbf{R}|| > 12\sqrt{2m}] < e^{-2m}$.
- *Let* $\mathbf{R}$ *be sampled from* $\mathcal{D}_{\mathbb{Z}^{m \times m},\sigma}$*, then we have* $\mathsf{Pr}[||\mathbf{R}|| > \sigma\sqrt{m}] < e^{-2m}$.

2.3 Randomness Extraction

We will use the following lemma to argue the indistinghishability of two different distributions, which is a generalization of the leftover hash lemma proposed by Dodis et al. [DRS04].

Lemma 2 ([ABB10]). *Suppose that* $m > (n+1)\log q + \omega(\log n)$*. Let* $\mathbf{R} \in \{-1, 1\}^{m \times k}$ *be chosen uniformly at random for some polynomial* $k = k(n)$*. Let* $\mathbf{A}, \mathbf{B}$ *be matrix chosen randomly from* $\mathbb{Z}_q^{n \times m}, \mathbb{Z}_q^{n \times k}$ *respectively. Then, for all vectors* $\boldsymbol{w} \in \mathbb{Z}^m$*, the two following distributions are statistically close:*

$$(\mathbf{A}, \mathbf{AR}, \mathbf{R}^T\boldsymbol{w}) \approx (\mathbf{A}, \mathbf{B}, \mathbf{R}^T\boldsymbol{w})$$

2.4 Learning With Errors

The LWE problem was introduced by Regev [Reg05], who showed that solving it *on average* is as hard as (quantumly) solving several standard lattice problems *in the worst case*.

Definition 2 (LWE). *For an integer* $q = q(n) \geq 2$*, and an error distribution* $\chi = \chi(n)$ *over* $\mathbb{Z}_q$*, the* Learning With Errors problem $\mathsf{LWE}_{n,m,q,\chi}$ *is to distinguish between the following pairs of distributions (e.g. as given by a sampling oracle* $\mathcal{O} \in \{\mathcal{O}_{\boldsymbol{s}}, \mathcal{O}_{\$}\}$*):*

$$\{\mathbf{A}, \mathbf{A}^T\boldsymbol{s} + \boldsymbol{x}\} \text{ and } \{\mathbf{A}, \boldsymbol{u}\}$$

where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$*,* $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$*,* $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_q^m$*, and* $\boldsymbol{x} \xleftarrow{\$} \chi^n$.

Theorem 2 ([Reg05,Pei09,MM11,MP12]). *Let $q = q(n) \in \mathbb{N}$ be either a prime power or a product of small (size $poly(n)$) distinct primes, and let $B \geq \omega(\log n) \cdot \sqrt{n}$. Then there exists an efficient sampleable B-bounded distribution χ such that if there is an efficient algorithm that solves the average-case* LWE *problem for parameters n, q, χ, then:*

- *There is an efficient quantum algorithm that solves $\mathsf{GapSVP}_{\tilde{O}(nq/B)}$ on any n-dimensional lattice.*
- *If $q \geq \tilde{O}(2^{n/2})$, then there is an efficient classical algorithm for $\mathsf{GapSVP}_{\tilde{O}(nq/B)}$ on any n-dimensional lattice.*

In both cases, if one also considers distinguishers with sub-polynomial advantage, then we require $B \geq \tilde{O}(n)$ and the resulting approximation factor is slightly lager than $\tilde{O}(n^{1.5}q/B)$.

Theorem 3 ([BLP+13]). *Solving n-dimensional* LWE *with $poly(n)$ modulus implies an equally efficient solution to a worst-case lattice problem (e.g.,* GapSVP*) in dimension $\sqrt{n}$.*

2.5 Two-Sided Trapdoors and Sampling Algorithms

We will use the following algorithms to sample short vectors from specified lattices.

Lemma 3 ([GPV08,AP10]). *Let q, n, m be positive integers with $q \geq 2$ and sufficiently large $m = \Omega(n \log q)$. There exists a* PPT *algorithm $\mathsf{TrapGen}(q, n, m)$ that with overwhelming probability outputs a pair $(\mathbf{A} \in \mathbb{Z}_q^{n \times m}, \mathbf{T_A} \in \mathbb{Z}^{m \times m})$ such that $\mathbf{A}$ is statistically close to uniform in $\mathbb{Z}_q^{n \times m}$ and $\mathbf{T_A}$ is a basis for $\Lambda_q^{\perp}(\mathbf{A})$ satisfying*

$$||\mathbf{T_A}|| \leq O(n \log q) \quad \textit{and} \quad ||\widetilde{\mathbf{T_A}}|| \leq O(\sqrt{n \log q})$$

except with $\mathsf{negl}(n)$ probability.

Lemma 4 ([GPV08,CHKP10,ABB10]). *Let $q > 2, m > n$. There are five algorithms as follows:*

- *There is a* PPT *algorithm $\mathsf{SamplePre}(\mathbf{A}, \mathbf{T_A}, \boldsymbol{u}, \sigma)$: It takes as input*
  - *a rank-n matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and a vector $\boldsymbol{u} \in \mathbb{Z}_q^n$,*
  - *a "short" basis $\mathbf{T_A}$ for lattice $\Lambda_q^{\perp}(\mathbf{A})$, and a Gaussian parameter $\sigma > ||\widetilde{\mathbf{T_A}}|| \cdot \omega(\sqrt{\log m})$,*

  then outputs a vector $\boldsymbol{x} \in \Lambda_q^{\perp}(\mathbf{A})$ distributed statistically close to $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{A}),\sigma}$, whenever $\Lambda_q^{\boldsymbol{u}}(\mathbf{A})$ is not empty.

- *There is a deterministic polynomial algorithm $\mathsf{ExtBasis}(\mathbf{S}, \mathbf{A}' = \mathbf{A}\|\bar{\mathbf{A}})$: It takes as input*
  - *an arbitrary $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ whose columns generate the entire group $\mathbb{Z}_q^n$, and an arbitrary $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times \bar{m}}$,*
  - *an arbitrary basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\Lambda^{\perp}(\mathbf{A})$,*

then outputs a basis $\mathbf{S}'$ *of* $\Lambda^{\perp}(\mathbf{A}')$ *such that* $\|\tilde{\mathbf{S}'}\| = \|\tilde{\mathbf{S}}\|$. *Moreover, the same holds even for any given permutation of the columns of* $\mathbf{A}'$ *(e.g., if columns of* $\bar{\mathbf{A}}$ *are both appended and prepended to* $\mathbf{A}$*).*

- *There is a* PPT *algorithm* $\mathsf{RandBasis}(\mathbf{S}, s)$ *:It takes as input*
  - *a basis* $\mathbf{S}$ *of an* m*-dimensional integer lattice* Λ*,*
  - *a parameter* $s \geq \|\tilde{\mathbf{S}}\| \cdot \omega(\sqrt{\log n})$*,*

  With overwhelming probability, it outputs a basis $\mathbf{S}'$ *of* Λ*, which is (essentially) statistically independent of the original basis, and* $\|\mathbf{S}'\| \leq s\sqrt{m}$.

- *There is a* PPT *algorithm* $\mathsf{SampleLeft}(\mathbf{A}, \mathbf{B}, \mathbf{T_A}, \boldsymbol{u}, s)$*: It takes as input*
  - *a rank-*n *matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$*, and any matrix* $\mathbf{B} \in \mathbb{Z}_q^{n \times m_1}$*,*
  - *a "short" basis* $\mathbf{T_A}$ *for lattice* $\Lambda_q^{\perp}(\mathbf{A})$*, a vector* $\boldsymbol{u} \in \mathbb{Z}_q^n$*,*
  - *and a Gaussian parameter* $s > \|\widetilde{\mathbf{T_A}}\| \cdot \omega(\sqrt{\log(m + m_1)})$*,*

  then outputs a vector $\boldsymbol{r} \in \mathbb{Z}^{m+m_1}$ *distributed statistically close to* $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{F}),s}$ *where* $\mathbf{F} := (\mathbf{A}|\mathbf{B})$.

- *There is a* PPT *algorithm* $\mathsf{SampleRight}(\mathbf{A}, \mathbf{B}, \mathbf{R}, \mathbf{T_B}, \boldsymbol{u}, s)$*: It takes as input*
  - *any matrix* $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$*, and a rank-*$n$ *matrix* $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$*,*
  - *a matrix* $\mathbf{R} \in \mathbb{Z}_q^{m \times m}$*, where* $s_{\mathbf{R}} := \|\mathbf{R}\| = \sup_{\boldsymbol{x}:\|\boldsymbol{x}\|=1} \|\mathbf{R}\boldsymbol{x}\|$*,*
  - *a "short" basis* $\mathbf{T_B}$ *for lattice* $\Lambda_q^{\perp}(\mathbf{B})$*, a vector* $\boldsymbol{u} \in \mathbb{Z}_q^n$*,*
  - *and a Gaussian parameter* $s > \|\widetilde{\mathbf{T_B}}\| \cdot s_{\mathbf{R}} \cdot \omega(\sqrt{\log m})$*,*

  then outputs a vector $\boldsymbol{r} \in \mathbb{Z}^{2m}$ *distributed statistically close to* $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{F}),s}$ *where* $\mathbf{F} := (\mathbf{A}|\mathbf{AR} + \mathbf{B})$.

3 Gadget Matrices and Gadget-Based Matrix Operations

In this section, we will show that the matrix $\mathbf{G}$ defined above can be used to support "invariant-preserving" pseudo-commutative matrix multiplication operations. We can also generalize matrix $\mathbf{G}$ and its trapdoor to other integer powers or mixed-integer products. lastly, we propose a new arrangement of matrix operations for the generalized gadget matrices that is critical to our main result.

3.1 Matrix Operations with G via the Flattening Function $\mathbf{G^{-1}}$

Over general lattices, the only matrices that commute with $\mathbf{G} \in \mathbb{Z}_q^{n \times m}$ are scaled identity matrices $\alpha\mathbf{I}$, in the sense that $\mathbf{G} \cdot (\alpha\mathbf{I}_m) = (\alpha\mathbf{I}_n) \cdot \mathbf{G}$. Note that if the $\mathbf{G}$ is padded, then $\alpha\mathbf{I}_m$ could alternatively be the block matrix $\mathbf{A}$ containing $\alpha\mathbf{I}_n$ in the appropriate with zeroes everywhere else.

The works of Xagawa [Xag13] and Alperin-Sheriff and Peikert [AP14], among others, describe a technique that resolves this non-commutatively problem for $\mathbf{G}$. In particular, there is an efficiently computable function $\mathbf{G}^{-1}$, so that for any matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$so that $\mathbf{G}^{-1}(\mathbf{B}) = \mathbf{X} \in \{-1, 1\}^{m \times m}$ and $\mathbf{GX} = \mathbf{B}$. This allows for

"pseudo-commutative" multiplication of the gadget matrix $\mathbf{G}$ by any square matrix $\mathbf{M}$ of dimension n, by observing that

$$\mathbf{G} \cdot (\mathbf{G}^{-1}(\mathbf{MG})) = \mathbf{M} \cdot \mathbf{G} \tag{1}$$

The matrix $\mathbf{X} = \mathbf{G}^{-1}(\mathbf{B})$ has small norm independent of $\mathbf{B}$.

3.2 Non-Binary Gadgets $\mathbf{G}_{n,r,m}$ and Function $\mathbf{G}^{-1}_{n',r',m'}(.)$

The matrix $\mathbf{G}$ and its trapdoor can be extended to other integer powers or mixed-integer products by the results of [MP12]. Then, we can give a generalized notation for gadget matrices as follows:

For any modulus $q \geq 2$, for integer $2 \leq r \leq q$, let $\mathbf{g}_r^T = [1, r, r^2, ..., r^{k_r - 1}] \in \mathbb{Z}_q^{1 \times k_r}$ for $k_r = \lceil \log_r q \rceil$. We let $\mathbf{G}_{n,r} = \mathbf{g}_r^T \otimes \mathbf{I}_n \in \mathbb{Z}_q^{n \times nk_r}$. The public trapdoor basis can be given analogously. Similar to the above padding argument, $\mathbf{G}_{n,r} \in \mathbb{Z}_q^{n \times nk_r}$ can be padded into a matrix $\mathbf{G}_{n,r,m} \in \mathbb{Z}_q^{n \times m}$ for $m \geq nk_r$ without increasing the norm of $\widetilde{\mathbf{T}_{\mathbf{G}_{n,r,m}}}$ from that of $\widetilde{\mathbf{T}_{\mathbf{G}_{n,r}}}$.

In this paper, we do not need to use $\widetilde{\mathbf{T}_{\mathbf{G}_{n,r,m}}}$ or $\widetilde{\mathbf{T}_{\mathbf{G}_{n,r}}}$ at all, but keep the discussion for exposition. Under this notation, the matrix $\mathbf{G}$ in dimension n is either $\mathbf{G}_{n,2} \in \mathbb{Z}_q^{n \times n \log_2 q}$ or its padded version $\mathbf{G}_{n,2,m} \in \mathbb{Z}_q^{n \times m}$ depending on the setting.

We now introduce a related function - the Batch Change-of-Base function $\mathbf{G}^{-1}_{n',r',m'}(.)$ - as follows:

For any modulus $q \geq 2$, and any integer base $2 \leq r' \leq q$, let integer $k_{r'} = \lceil \log_{r'} q \rceil$. For any integer $N' \geq 2$ and $m' \geq n' k_{r'}$ the function $\mathbf{G}^{-1}_{n',r',m'}(.)$ takes as input a matrix from $\mathbb{Z}_q^{n' \times m'}$, first computes a matrix in $\{0, 1, ..., r' - 1\}^{n' \log_{r'} q \times m'}$ using the $\mathbf{G}^{-1}$, then pads with rows of zeroes needed to form a matrix in $\{0, 1, ..., r' - 1\}^{m' \times m'}$. For example, the typical base-2 $\mathbf{G}^{-1} = \mathbf{G}^{-1}_{n,2,m}$ takes $\mathbb{Z}_q^{n \times m}$ to $\{0,1\}^{m \times m}$ as expected.

3.3 Further Gadget-Based Matrix Multiplication Operations

In this part, we propose a new arrangement of matrix operations(pseudo-commutative and non-commutative) that is critical to our main construction in this paper. First, fix any integer n, for some sufficiently small $\ell = \omega(1) \ll \log_2 q \approx \log_2 n$, define $\ell' = 2^\ell$ such that $\ell' = \omega(1) < \log_2 q \approx \log_2 n$. Finally, fix any integer $m \geq n\ell k_{\ell'}$.

Then for any two matrices $\mathbf{H} \in \mathbb{Z}_q^{n \times n\ell}, \mathbf{X} \in \mathbb{Z}_q^{\ell n \times n}$, consider the following terms, in order:

1. First, consider the base-2, dimension-n "gadget-encoding" of $\mathbf{X} \in \mathbb{Z}_q^{\ell n \times n}$, i.e. the matrix
$$\mathbf{X} \cdot \mathbf{G}_{n,2,m} \in \mathbb{Z}_q^{\ell n \times m} = \mathbb{Z}^{n \log_2(\ell') \times m}. \tag{2}$$
2. Next, consider the base-ℓ', dimension-$(n\ell)$ flatting(with zero-row padding) of the above:
$$\mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{X} \cdot \mathbf{G}_{n,2,m}) \in \{0, 1, ..., \ell' - 1\}^{m \times m} \subsetneq \mathbb{Z}_q^{m \times m}. \tag{3}$$

3. Then, consider the base-ℓ', dimension-$(n\ell)$ gadget-encoding of $\mathbf{H} \in \mathbb{Z}_q^{n \times n\ell}$, i.e. the matrix
$$\mathbf{H} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n \times m}. \tag{4}$$
4. Finally – for sufficient large m, we find the following relationship holds:
$$(\mathbf{H} \cdot \mathbf{G}_{n\ell,\ell',m}) \cdot (\mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{X} \cdot \mathbf{G}_{n,2,m})) = (\mathbf{H} \cdot \mathbf{X}) \cdot \mathbf{G}_{n,2,m} \in \mathbb{Z}_q^{n \times m} \tag{5}$$

with $\|\mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{X} \cdot \mathbf{G}_{n,2,m})\| =$ **small**, conditioned on the sufficiently small choice of $\ell = \omega(1)$. We emphasize that only public information, n, m, ℓ, is required to perform this batch base-change-then-multiply operation, when given as input any $\mathbf{M} \in \mathbb{Z}_q^{n \times m}$(equal to $\mathbf{H} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n \times m}$) and any $\mathbf{X} \in \mathbb{Z}_q^{\ell n \times n}$.

Definition 3. *We refer to the matrix $\mathbf{H} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n \times m}$ as the predicate-encoding of $\mathbf{H} \in \mathbb{Z}_q^{n \times n\ell}$, and to the matrix $\mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{X} \cdot \mathbf{G}_{n,2,m}) \in \{0, 1, ..., \ell' - 1\}^{m \times m} \subsetneq \mathbb{Z}_q^{m \times m}$ as the input-encoding of $\mathbf{X} \in \mathbb{Z}_q^{\ell n \times n}$.*

4 Our Main Construction

We describe our IPE scheme that each secret key will be associated with a predicate vector $\boldsymbol{v} \in \mathbb{Z}_q^{\ell}$(for some fixed $\ell \geq 2$) and each ciphertext will be associated with an attribute vector $\boldsymbol{w} \in \mathbb{Z}_q^{\ell}$. Decryption may succeed if and only if $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0 (mod\ q)$.

4.1 Construction

Let $\lambda \in \mathbb{Z}^+$ be the security parameter and $\ell = \log(n)$ be the length of predicate and attribute vectors. Let $q = \mathsf{poly}(n)$ and $m = O(n \log(q))$ be positive integers. Let σ and α be positive real Gaussian parameters. Define $k = \lfloor \log_2 q \rfloor$.

$\mathsf{Setup}(1^\lambda, 1^\ell)$: On input a security parameter λ and a parameter ℓ denoting the length of predicate and
attribute vectors, do:

1. Use the algorithm $\mathsf{TrapGen}\ (1^\lambda, q, n, m)$ to generate a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ together with a trapdoor $\mathbf{T_A}$.
2. Choose a random matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times m}$.
3. Choose a random vector $\boldsymbol{u} \in \mathbb{Z}_q^n$.

Output $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \boldsymbol{u}), \mathsf{msk} = \mathbf{T_A}$.

$\mathsf{KeyGen}(\mathsf{pp}, \mathsf{msk}, \boldsymbol{v})$: On input the public parameters pp, the master secret key msk, and a predicate vector
$\boldsymbol{v} = (v_1, ..., v_\ell) \in \mathbb{Z}_q^{\ell}$, do:

1. Define the input-encoding matrices

$$\mathbf{V}^{'} = \begin{bmatrix} v_1\mathbf{I}_n \\ v_2\mathbf{I}_n \\ \vdots \\ v_\ell\mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\ell n \times n}, \quad \mathbf{V} = \mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{V}^{'} \cdot \mathbf{G}_{n,2,m}) \in [\ell^{'}]^{m\times m}. \tag{6}$$

2. Define the matrix $\mathbf{U} = \mathbf{BV} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{A}_{\boldsymbol{v}} = [\mathbf{A}|\mathbf{U}]$.

3. Using the master secret key $\mathsf{msk} = (\mathbf{T}_{\mathbf{A}}, \sigma)$, compute $\boldsymbol{r} \leftarrow \mathsf{SampleLeft}\,(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{U}, \boldsymbol{u}, \sigma)$. Then $\boldsymbol{r}$ is a vector in $\mathbb{Z}^{2m}$ satisfying $\mathbf{A}_{\boldsymbol{v}} \cdot \boldsymbol{r} = \boldsymbol{u}$ (mod q).

Output the secret key $\mathsf{sk}_{\boldsymbol{v}} = \boldsymbol{r}$.

$\mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}, m)$: On input public parameters pp, an attribute vectors $\boldsymbol{w}$, and a message $m \in \{0,1\}$, do:

1. Choose a uniformly random $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$.

2. Choose a noise vector $\boldsymbol{e}_0 \leftarrow D_{\mathbb{Z}_q^m,\alpha}$ and a noise term $e \leftarrow D_{\mathbb{Z}_q,\alpha}$.

3. Compute $\boldsymbol{c}_0 = \boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T$.

4. Define the matrix

$$\mathbf{W}^{'} = \left[w_1\mathbf{I}_n \; w_2\mathbf{I}_n \; \ldots \; w_\ell\mathbf{I}_n\right] \in \mathbb{Z}_q^{n\times \ell n}, \quad \mathbf{W} = \mathbf{W}^{'} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n\times m}. \tag{7}$$

Pick a random matrix $\mathbf{R} \xleftarrow{\$} \{-1,1\}^{m\times m}$, define error vector $\boldsymbol{e}_1^T = \boldsymbol{e}_0^T\mathbf{R}$. Set

$$\boldsymbol{c}_1 = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}) + \boldsymbol{e}_1^T, \quad c_2 = \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor\frac{q}{2}\rfloor m. \tag{8}$$

Output ciphertext $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1, c_2)$.

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_{\boldsymbol{v}}, \mathsf{ct}, \boldsymbol{v})$: On input public parameters pp ,a secret key $\mathsf{sk}_{\boldsymbol{v}}$ for predicate vector $\boldsymbol{v}$, and a ciphertext
ct, do:

1. Compute the vector $\boldsymbol{c}_v = \boldsymbol{c}_1 \cdot \mathbf{V}$.

2. Let $\boldsymbol{c} = [\boldsymbol{c}_0|\boldsymbol{c}_v]$.

3. Compute $z \leftarrow c_2 - \boldsymbol{cr}$(mod q).

Output 0 if $|z| < q/4$ and 1 otherwise.

4.2 Correctness

We prove correctness of our scheme as follow.

Lemma 5. *For certain parameter choices, our scheme is correct.*

Proof. Recall that the ciphertext $\mathsf{ct} = (\boldsymbol{c}_0, \boldsymbol{c}_1, c_2)$. For which $\boldsymbol{c}_0 = \boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T, \boldsymbol{c}_1 = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}) + \boldsymbol{e}_1^T, c_2 = \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor \frac{q}{2} \rfloor m$, so

$$\boldsymbol{c}_v = \boldsymbol{c}_1 \cdot \mathbf{V} = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}) \cdot \mathbf{V} + \boldsymbol{e}_1^T \cdot \mathbf{V} = \boldsymbol{s}^T \cdot \mathbf{B} \cdot \mathbf{V} + \boldsymbol{s}^T \cdot \mathbf{W} \cdot \mathbf{V} + \boldsymbol{e}_1^T \cdot \mathbf{V}. \quad (9)$$

It's easy to check that if $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$, then $\mathbf{W} \cdot \mathbf{V} = \mathbf{W}' \cdot \mathbf{V}' \cdot \mathbf{G}_{n,2,m} = \mathbf{0}$. Then $\boldsymbol{c}_v = \boldsymbol{s}^T \cdot \mathbf{B} \cdot \mathbf{V} + \boldsymbol{e}_1^T \cdot \mathbf{V}$, so

$$\begin{aligned} \boldsymbol{c} &= [\boldsymbol{c}_0 | \boldsymbol{c}_v] = [\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T | \boldsymbol{s}^T \cdot \mathbf{B} \cdot \mathbf{V} + \boldsymbol{e}_1^T \cdot \mathbf{V}] = \boldsymbol{s}^T[\mathbf{A}|\mathbf{B} \cdot \mathbf{V}] + [\boldsymbol{e}_0^T | \boldsymbol{e}_1^T \cdot \mathbf{V}] \\ &= \boldsymbol{s}^T[\mathbf{A}|\mathbf{U}] + [\boldsymbol{e}_0^T | \boldsymbol{e}_1^T \cdot \mathbf{V}] = \boldsymbol{s}^T\mathbf{A}_v + [\boldsymbol{e}_0^T | \boldsymbol{e}_1^T \cdot \mathbf{V}]. \end{aligned} \quad (10)$$

And

$$\begin{aligned} z = c_2 - \boldsymbol{c}\boldsymbol{r} (mod\, q) &= \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor \frac{q}{2} \rfloor m - \boldsymbol{s}^T\boldsymbol{u} - [\boldsymbol{e}_0^T | \boldsymbol{e}_1^T \cdot \mathbf{V}]\boldsymbol{r} \\ &= \lfloor \frac{q}{2} \rfloor m + \underbrace{e - [\boldsymbol{e}_0^T | \boldsymbol{e}_1^T \cdot \mathbf{V}]\boldsymbol{r}}_{\hat{\boldsymbol{e}}} (mod\, q) \end{aligned} \quad (11)$$

If the norm $\|\hat{\boldsymbol{e}}\| < q/4$, then $|z| < q/4$ if and only if $m = 0$, or else $m = 1$. In this condition, our scheme is correct. □

4.3 Security

In this part, we show the security proof of our IPE scheme as follows:

Theorem 4. *Assuming the hardness of the standard LWE assumption, Our IPE scheme described above is weakly attribute hiding.*

Proof: To prove the theorem we define a series games against adversary $\mathcal{A}$ that play the weak attribute hiding game. The adversary $\mathcal{A}$ outputs two attribute vectors $\boldsymbol{w}_0$ and $\boldsymbol{w}_1$ at the beginning of each game, and at some point output two messages m_0, m_1. The first and last games correspond to real security game with challenge ciphertexts $\mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}_0, m_0)$ and $\mathsf{Enc}(\mathsf{pp}, \boldsymbol{w}_1, m_1)$ respectively. In the intermediate games we use the "alternative" simulation algorithms $\mathsf{Sim.Setup}, \mathsf{Sim.KeyGen}, \mathsf{Sim.Enc}$. During the course of the game the adversary can only request keys for predicate vector $\boldsymbol{v}$ such that $\langle \boldsymbol{v}, \boldsymbol{w}_0 \rangle \neq 0$ and $\langle \boldsymbol{v}, \boldsymbol{w}_1 \rangle \neq 0$.

Game 0: The challenger runs Setup, answers $\mathcal{A}$'s secret key queries using KeyGen, and generates the challenge ciphertext using Enc with attribute $\boldsymbol{w}_0$ and message m_0.

Game 1: The challenger runs $\mathsf{Sim.Setup}$ with $\boldsymbol{w}^* = \boldsymbol{w}_0$, and answers $\mathcal{A}$'s secret key queries using $\mathsf{Sim.KeyGen}$. The challenger generates the challenge ciphertext using $\mathsf{Sim.Enc}$ with attribute $\boldsymbol{w}_0$ and message m_0.

Game 2: The challenger runs $\mathsf{Sim.Setup}$ with $\boldsymbol{w}^* = \boldsymbol{w}_0$, and answers $\mathcal{A}$'s secret

key queries using Sim.KeyGen. The challenger generates the challenge ciphertext by choosing a uniformly random element of the ciphertext space.

Game 3: The challenger runs Sim.Setup with $\boldsymbol{w}^* = \boldsymbol{w}_1$, and answers $\mathcal{A}$'s secret key queries using Sim.KeyGen. The challenger generates the challenge ciphertext by choosing a uniformly random element of the ciphertext space.

Game 4: The challenger runs Sim.Setup with $\boldsymbol{w}^* = \boldsymbol{w}_1$, and answers $\mathcal{A}$'s secret key queries using Sim.KeyGen. The challenger generates the challenge ciphertext using Sim.Enc with attribute $\boldsymbol{w}_1$ and message m_1.

Game 5: The challenger runs Setup, answers $\mathcal{A}$'s secret key queries using KeyGen, and generates the challenge ciphertext using Enc with attribute $\boldsymbol{w}_1$ and message m_1.

Now we define the "alternative" simulation algorithm as follows:

Sim.Setup $(1^\lambda, 1^\ell, \boldsymbol{w}^*)$: On input a security parameter λ, a parameter ℓ denoting the length of predicate(and the attribute) vector, and an attribute vector $\boldsymbol{w}^* \in \mathbb{Z}_q^\ell$, do the following:

1. Choose a random matrix $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times m}$ and a random vector $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_q^n$.
2. Choosing a random matrix $\mathbf{R}^* \xleftarrow{\$} \{-1, 1\}^{m\times m}$.
3. Define the matrix

$$\mathbf{W}^{'} = \left[w_1^*\mathbf{I}_n \; w_2^*\mathbf{I}_n \ldots w_\ell^*\mathbf{I}_n\right] \in \mathbb{Z}_q^{n\times \ell n}, \tag{12}$$

   and set $\mathbf{B} = \mathbf{A}\mathbf{R}^* - \mathbf{W}^{'} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n\times m}$.

Output $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \boldsymbol{u}), \mathsf{msk} = (\mathbf{R}^*, \mathsf{pp}, \mathbf{T}_{\mathbf{G}_{n,2,m}})$.

Sim.KeyGen $(\mathsf{pp}, \mathsf{msk}, \boldsymbol{v})$: On input a master key msk and a vector $\boldsymbol{v} \in \mathbb{Z}_q^\ell$, do the following:

1. Check $\langle \boldsymbol{v}, \boldsymbol{w}^* \rangle = 0$, if so ,output $\perp$.
2. Define the matrix

$$\mathbf{V}^{'} = \begin{bmatrix} v_1\mathbf{I}_n \\ v_2\mathbf{I}_n \\ \vdots \\ v_\ell\mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\ell n\times n}, \;\; \mathbf{V} = \mathbf{G}^{-1}_{n\ell,\ell',m}(\mathbf{V}^{'} \cdot \mathbf{G}_{n,2,m}). \tag{13}$$

   Set $\mathbf{U} = \mathbf{B}\mathbf{V}, \mathbf{A}_{\boldsymbol{v}} = [\mathbf{A}|\mathbf{U}] = [\mathbf{A}|\mathbf{A}\mathbf{R}^*\mathbf{V} - \mathbf{W}^{'}\mathbf{V}^{'}\mathbf{G}_{n,2,m}]$.
3. Let $\boldsymbol{r} \leftarrow$ SampleRight $(\mathbf{A}, \mathbf{R}^*\mathbf{V}, \mathbf{W}^{'}\mathbf{V}^{'}, \mathbf{T}_{\mathbf{G}_{n,2,m}}, \boldsymbol{u}, \sigma)$, so that $\mathbf{A}_{\boldsymbol{v}} \cdot \boldsymbol{r} = \boldsymbol{u}$.

Output the secret key $\mathsf{sk}_{\boldsymbol{v}} = \boldsymbol{r}$.

$\mathsf{Sim.Enc}\ (\mathsf{pp}, \boldsymbol{w}, m, \mathsf{msk})$: This algorithm is the same as the Enc algorithm, except that $\mathbf{R}^*$ is used instead of the random matrix $\mathbf{R}$ in step 4.

We show that each pair of games($\mathbf{Game}\ i$, $\mathbf{Game}\ i+1$) are either statistically indistinguishable or computationally indistinguishable under the DLWE assumption.

Lemma 6. *The view of the adversary $\mathcal{A}$ in* $\mathbf{Game\ 0}$ *is statistically closed to the view of $\mathcal{A}$ in* $\mathbf{Game\ 1}$*, similarly,* $\mathbf{Game\ 4}$ *is statistically closed to* $\mathbf{Game\ 5}$.

Proof. We just prove the case for $\mathbf{Game\ 0}$ and $\mathbf{Game\ 1}$.

First, The matrix $\mathbf{A}$ in pp is chosen by running $\mathsf{TrapGen}$ in $\mathbf{Game\ 0}$, whereas it is a uniformly random matrix in $\mathbb{Z}_q^{n\times m}$ in $\mathbf{Game\ 1}$. Since $m = \Omega(n\log q)$, by $\mathbf{Lemma 2.5}$, the matrix $\mathbf{A}$ output by $\mathsf{TrapGen}$ is statistically indistinguishable from a uniformly random matrix, and thus the distribution of $\mathbf{A}$ in $\mathbf{Game\ 0}$ and $\mathbf{Game\ 1}$ are statistically close.

Next, We show that the distribution of $\mathbf{B}$ in pp and $\boldsymbol{c}_1$ in ciphertext in $\mathbf{Game\ 0}$ and $\mathbf{Game\ 1}$ are statistically indistinguishable. The difference between $(\mathbf{B}, \boldsymbol{c}_1)$ in two games is as follows:

- In $\mathbf{Game\ 0}$ the matrix $\mathbf{B}$ is uniformly random in $\mathbb{Z}_q^{n\times m}$. In $\mathbf{Game\ 1}$, $\mathbf{B} = \mathbf{AR}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}$. The matrix $\mathbf{R}^*$ is uniformly chosen from $\{-1,1\}^{m\times m}$.

- In $\mathbf{Game\ 0}$ the challenge ciphertext components $\boldsymbol{c}_1$ are computed as $\boldsymbol{c}_1 = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}) + \boldsymbol{e}_1^T = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}) + \boldsymbol{e}_0^T\mathbf{R}^*$, where $\mathbf{B}$ is uniformly random in $\mathbb{Z}_q^{n\times m}$ and the matrix $\mathbf{R}^*$ is uniformly chosen from $\{-1,1\}^{m\times m}$.
  In $\mathbf{Game\ 1}$ the challenge ciphertext components $\boldsymbol{c}_1$ is $\boldsymbol{c}_1 = \boldsymbol{s}^T(\mathbf{AR}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m} + \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}) + \boldsymbol{e}_0^T\mathbf{R}^* = \boldsymbol{s}^T\mathbf{AR}^* + \boldsymbol{e}_0^T\mathbf{R}^* = (\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T)\mathbf{R}^*$. Where $\mathbf{R}^*$ are the same matrix used to compute the public parameters $\mathbf{B}$. So the main difference between the two games is that the matrix $\mathbf{R}^*$ is chosen by Enc and used only in the ciphertext $\boldsymbol{c}_1$ in $\mathbf{Game\ 0}$, whereas in $\mathbf{Game\ 1}$, it plays a double role: it's used to construct the matrix $\mathbf{B}$ in pp as well as the ciphertext $\boldsymbol{c}_1$.

Now we show that the distributions $(\mathbf{A}, \mathbf{B}, \boldsymbol{c}_1)$ in $\mathbf{Game\ 0}$ and $\mathbf{Game\ 1}$ are statistically indistinguishable. for $\mathbf{B}$ is uniformly random and $\mathbf{R}^*$ is uniformly random in $\{-1,1\}$, so by the $generalized$ leftover hash lemma, the following two distributions are statistically indistinguishable:

$$(\mathbf{A}, \mathbf{AR}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}, \boldsymbol{e}_0^T\mathbf{R}^*) \approx_s (\mathbf{A}, \mathbf{B}, \boldsymbol{e}_0^T\mathbf{R}^*) \tag{14}$$

We add the same quantity to both sides of Equation(14), we see that the following two distributions are statistically close:

$$\begin{aligned}(\mathbf{A}, \mathbf{AR}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}, \underbrace{\boldsymbol{s}^T(\mathbf{AR}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m} + \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m})}_{added\ term} + \boldsymbol{e}_0^T\mathbf{R}^*) \\ \approx_s (\mathbf{A}, \mathbf{B}, \underbrace{\boldsymbol{s}^T(\mathbf{B} + \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m})}_{added\ term} + \boldsymbol{e}_0^T\mathbf{R}^*)\end{aligned} \tag{15}$$

The distribution on the left hand side of (15) is the distribution of $(\mathbf{A}, \mathbf{B}, \boldsymbol{c}_1)$ in $\mathbf{Game\ 1}$, while the right hand side is the distribution in $\mathbf{Game\ 0}$. So the two distributions are statistically indistinguishable. □

Next,we show the secret keys output by Sim.KeyGen are statistically indistinguishable from those output by KeyGen. Assuming σ is sufficiently large, it follows from the properties of the algorithm SampleLeft and SampleRight, in both games the secret key is chosen from $\mathcal{D}_{\Lambda_q^{\boldsymbol{u}}(\mathbf{A}_v),\sigma}$ with overwhelming probability, so the two keys are statistically indistinguishable.

Lemma 7. *If the DLWE assumption holds, then the view of $\mathcal{A}$ in* $\mathbf{Game\ 1}$ *is computationally indistinguishable from the view of $\mathcal{A}$ in* $\mathbf{Game\ 2}$*, similarly,* $\mathbf{Game\ 3}$ *is computationally indistinguishable from* $\mathbf{Game\ 4}$*.*

Proof. We just prove the case for $\mathbf{Game\ 1}$ and $\mathbf{Game\ 2}$.

Suppose we are given $m+1$ LWE instances $(\boldsymbol{a}_i, b_i)$ for $i = 0, ..., m$, where either $b_i = \boldsymbol{s}^T\boldsymbol{a}_i + e_i$ for some fixed random secret $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$ and Gaussian noise $e_i \leftarrow D_{\mathbb{Z}_q,\alpha}$ or b_i is uniformly random in $\mathbb{Z}_q$. Define the following variables:

$$\begin{aligned} &\mathbf{A} = \left[\boldsymbol{a}_1 \ldots \boldsymbol{a}_m\right] \in \mathbb{Z}_q^{n\times m} \qquad &&\boldsymbol{u} = \boldsymbol{a}_0 \\ &\boldsymbol{c}_0 = (b_1, ..., b_m) \in \mathbb{Z}_q^m \qquad &&c_2 = b_0 + \lfloor\frac{q}{2}\rfloor m \end{aligned} \tag{16}$$

We simulate the challenger as follows:

- $\mathbf{Setup}$: Run Sim.Setup with $\boldsymbol{w}^* = \boldsymbol{w}_0$. and let $\mathbf{A}$, $\boldsymbol{u}$ as defined above.
- $\mathbf{Secret\ key\ queries}$: Run Sim.KeyGen .
- $\mathbf{Challenge\ ciphertext}$: Let $\boldsymbol{c}_1 = \boldsymbol{c}_0\mathbf{R}^*$(using the $\mathbf{R}^* \in$ msk). Output $(\boldsymbol{c}_0, \boldsymbol{c}_1, c_2)$.

In the Sim.Enc algorithm it sets:

$$\begin{aligned} \boldsymbol{c}_1 &= \boldsymbol{s}^T(\mathbf{A}\mathbf{R}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m} + \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}) + \boldsymbol{e}_0^T\mathbf{R}^* = \boldsymbol{s}^T\mathbf{A}\mathbf{R}^* + \boldsymbol{e}_0^T\mathbf{R}^* \\ &= (\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T)\mathbf{R}^*. \end{aligned} \tag{17}$$

So if $b_i = \boldsymbol{s}^T\boldsymbol{a}_i + e_i$, then $\boldsymbol{c}_1 = \boldsymbol{c}_0\mathbf{R}^*$ and the simulator is the same as a $\mathbf{Game\ 1}$ challenger. Whereas if b_i is random in $\mathbb{Z}_q$, then simulated ciphertext is $(\boldsymbol{c}_0, \boldsymbol{c}_0\mathbf{R}^*, c_2)$. By the leftover hash lemma, $\boldsymbol{c}_0\mathbf{R}^*$ is uniformly random. Thus the ciphertext in this case is uniformly random and the simulator is identical to $\mathbf{Game\ 2}$'s challenger.

We conclude that any efficient adversary that can distinguish $\mathbf{Game\ 1}$ from $\mathbf{Game\ 2}$ can solve the DLWE problem. □

Lemma 8. *The view of $\mathcal{A}$ in* $\mathbf{Game\ 2}$ *is statistically indistinguishable from the view of $\mathcal{A}$ in* $\mathbf{Game\ 3}$*.*

Proof. Note that the only difference between the two games is that the attribute vector $\boldsymbol{w}^*$ is equal to $\boldsymbol{w}_0$ in $\mathbf{Game\ 2}$, whereas $\boldsymbol{w}^* = \boldsymbol{w}_0$ in $\mathbf{Game\ 3}$. The attribute vector $\boldsymbol{w}^*$ appears in the public parameter $\mathbf{B} = \mathbf{A}\mathbf{R}^* - \mathbf{W}^{'}\mathbf{G}_{n\ell,\ell',m}$, by the $generalized$

leftover hash lemma $(\mathbf{A}, \mathbf{AR}^*)$ is statistically indistinguishable from $(\mathbf{A},\mathbf{U}_{uni})$, where $\mathbf{U}_{uni}$ is uniformly random. So $\mathbf{AR}^* - \mathbf{X}$ for any fixed $\mathbf{X}$ is also uniformly random. It follows that the distribution of $\mathbf{B}$ in the two games are statistically close. □

Now we conclude the proof of the Theorem. Suppose that there is an efficient adversary $\mathcal{A}$ that can win the security game. Let $\mathcal{A}^{(i)}$ denote the output of $\mathcal{A}$ in $\mathbf{Game}\ i$. We have

$$|Pr[\mathcal{A}^{(0)} = 1] - Pr[\mathcal{A}^{(5)} = 1]| \geq \frac{1}{poly(n)}. \tag{18}$$

By a standard hybrid argument, this implies that

$$|Pr[\mathcal{A}^{(i)} = 1] - Pr[\mathcal{A}^{(i+1)} = 1]| \geq \frac{1}{poly(n)}. \tag{19}$$

for $i = 0, ..., 4$. Since $\mathcal{A}$ is polynomial time, $\mathbf{Lemma}(\mathbf{4.3})$ implies that (19) cannot hold for $i = 0$ or 4, $\mathbf{Lemma}(\mathbf{4.5})$ implies that (19) cannot hold for $i = 2$. So from $\mathbf{Lemma}(\mathbf{4.4})$ $\mathcal{A}$ can be used to solve the DLWE problem. □

4.4 Parameter Setting

We set the system parameters as follows, in which the $\delta > 0$ is an arbitrarily small constant:

parameters	Description	Setting
λ	security parameter	-
n	pp-lattice row dimension	$n = \lambda$
m	pp-lattice column dimension	$m = n^{1+\delta}$
q	modulus	$q = n^{4.5+4\delta}\log^{3.5+2\delta}(n)$
s	SampleLeft and SampleRight width	$s = n^{1.5+1.5\delta}\log^{1.5+\delta}(n)$
α	error width	$\alpha = \sqrt{n}\log^{1+\delta}(n)$
ℓ	predicate vectors and attribute vectors dimension	$\ell = \log\log(n)$
$\ell^{'}$	integer-base parameter	$\ell^{'} = \log(n)$

Table 2. Parameters and Example setting

We chose the parameters to satisfy the following constraints:

– To ensure correctness, it requires that $|e - [\boldsymbol{e}_0^T|\boldsymbol{e}_1^T \cdot \mathbf{V}]\boldsymbol{r}| < q/4$; we just need to bound the dominating term. So we parse the vector $\boldsymbol{r}$ into two equal-length vectors $\boldsymbol{r}_1$ and $\boldsymbol{r}_2$, then:

$$|\boldsymbol{e}_1^T.\mathbf{V}\cdot\boldsymbol{r}_2| \leq \|\boldsymbol{e}_0^T\|\cdot\|\mathbf{R}\|\cdot\|\mathbf{V}\|\cdot\|\boldsymbol{r}_2\| \approx \alpha\sqrt{m}\cdot\sqrt{m}\cdot\ell^{'}m\cdot s\sqrt{m} = m^{2.5}s\alpha\ell^{'} < q/4 \tag{20}$$

– For SampleLeft, we know $\|\widetilde{\mathbf{T}_\mathbf{A}}\| = O(\sqrt{n\log(q)})$, so it needs that the Gaussian parameter s satisfies

$$s > \sqrt{n\log(q)}.\omega(\sqrt{\log(m)}). \tag{21}$$

– For SampleRight, it's known that $\|\widetilde{\mathbf{T}_{\mathbf{A}}}\| \leq 5$ and in the proof of security

$$s_{\mathbf{R}} \leq \|\mathbf{R}^*\| \cdot \|\mathbf{V}\| \leq 12\sqrt{2m} \cdot (\ell' m) = O(2^{\ell} m^{1.5}) \tag{22}$$

Therefore, we need the Gaussian parameter s to satisfy a stronger constraint

$$s > 2^{\ell} m^{1.5} \omega(\sqrt{\log(m)}). \tag{23}$$

– To apply the Leftover Hash Lemma, we need $m \geq (n+1)\log(q) + \omega(\log(n))$.

– For the hardness of DLWE assumption, we apply Regev's reduction, then we need $\alpha > \sqrt{n}\omega(\log(n))$.

Remark. Regev [Reg05] showed that there exists an efficiently samplable B-bounded distribution χ for $B \geq \sqrt{n}.\omega(\log(n))$, so that if there is an efficient algorithm that solves the (average-case)$\mathsf{LWE}_{n,m,q,\chi}$ problem, then there is an efficient quantum algorithm that solves $\mathsf{GapSVP}_{\tilde{O}(n.q/B)}$ and $\mathsf{SIVP}_{\tilde{O}(n.q/B)}$ on any n-dimensional lattice. The work of Brakerski et al.[BLP+13] shows an analogous-but-classical result for any $\sqrt{n}$-dimensional lattice. For the case of our parameter setting ,the resulting lattice problems' approximation factor is $\widetilde{O}(n^{5+\epsilon})$, for $\epsilon > 0$.

Further Optimizations. We just use a trivial bound $\ell' m$ to bound the matrix $\mathbf{V}$. However, the $\mathbf{V}$ is highly sparse in fact, it's possible to optimize $\mathbf{V}$; e.g. about $\ell' \log_2(q)$. Then the parameters s and q can be selected more aggressively, e.g. $s \approx \sqrt{m}$ and $q \approx n^{2.5}$, this will result in a better approximation factor of about $\widetilde{O}(n^{3+\epsilon})$.

Using the tag-based sampling algorithms of [MP12] in place of [ABB10]'s SampleRight procedure leads to an approximation factor of $\widetilde{O}(n^{1.5})$, which matches the approximation factor of Dual Regev type $\mathbf{PKE}$, up to polylogarithmic factors in the security parameter n.

5 Conclusion and Open Problems

We presented a lattice-based inner product encryption scheme whose security follows from the difficulty of the $learning\ with\ errors$(LWE) problem.Our construction has more compact public key and ciphertexts size than precious lattice-based IPE scheme. We also used the techniques to construct a hierarchical inner product encryption scheme and a fuzzy identity-based encryption scheme, the efficiency of the two schemes is better than prior works. Our scheme is weakly attribute hiding in the selective security model, constructing a scheme that is adaptive (fully) security or fully attribute hiding are important open problems.

Acknowledgments.

References

ABB10. Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (H)IBE in the standard model. In Gilbert [Gil10], pages 553–572.

ABV+12. Shweta Agrawal, Xavier Boyen, Vinod Vaikuntanathan, Panagiotis Voulgaris, and Hoeteck Wee. Functional encryption for threshold functions (or fuzzy ibe) from lattices. In Marc Fischlin, Johannes Buchmann, and Mark Manulis, editors, *PKC 2012*, volume 7293 of *LNCS*, pages 280–297. Springer, Heidelberg, May 2012.

ADM12. Michel Abdalla, Angelo De Caro, and Karina Mochetti. Lattice-based hierarchical inner product encryption. In Alejandro Hevia and Gregory Neven, editors, *LATINCRYPT 2012*, volume 7533 of *LNCS*, pages 121–138. Springer, Heidelberg, October 2012.

AFV11. Shweta Agrawal, David Mandell Freeman, and Vinod Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 21–40. Springer, Heidelberg, December 2011.

AL10. Nuttapong Attrapadung and Benoît Libert. Functional encryption for inner product: Achieving constant-size ciphertexts with adaptive security or support for negation. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 384–402. Springer, Heidelberg, May 2010.

AP10. Joël Alwen and Chris Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48(3):535–553, 2010.

AP14. Jacob Alperin-Sheriff and Chris Peikert. Faster bootstrapping with polynomial error. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 297–314. Springer, Heidelberg, August 2014.

BH08. Dan Boneh and Michael Hamburg. Generalized identity based and broadcast encryption schemes. In Josef Pieprzyk, editor, *ASIACRYPT 2008*, volume 5350 of *LNCS*, pages 455–470. Springer, Heidelberg, December 2008.

BLP+13. Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013.

BW07. Dan Boneh and Brent Waters. Conjunctive, subset, and range queries on encrypted data. In Salil P. Vadhan, editor, *TCC 2007*, volume 4392 of *LNCS*, pages 535–554. Springer, Heidelberg, February 2007.

CHKP10. David Cash, Dennis Hofheinz, Eike Kiltz, and Chris Peikert. Bonsai trees, or how to delegate a lattice basis. In Gilbert [Gil10], pages 523–552.

DRS04. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 523–540. Springer, Heidelberg, May 2004.

Gil10. Henri Gilbert, editor. *EUROCRYPT 2010*, volume 6110 of *LNCS*. Springer, Heidelberg, May 2010.

GPV08. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

GSW13. Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

HAO15. Ryo Hiromasa, Masayuki Abe, and Tatsuaki Okamoto. Packing messages and optimizing bootstrapping in GSW-FHE. In Jonathan Katz, editor, *PKC 2015*, volume 9020 of *LNCS*, pages 699–715. Springer, Heidelberg, March / April 2015.

KSW08. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EU-*

ROCRYPT 2008, volume 4965 of *LNCS*, pages 146–162. Springer, Heidelberg, April 2008.

LOS+10. Allison B. Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. In Gilbert [Gil10], pages 62–91.

MM11. Daniele Micciancio and Petros Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In Phillip Rogaway, editor, *CRYPTO 2011*, volume 6841 of *LNCS*, pages 465–484. Springer, Heidelberg, August 2011.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In Pointcheval and Johansson [PJ12], pages 700–718.

OT09. Tatsuaki Okamoto and Katsuyuki Takashima. Hierarchical predicate encryption for inner-products. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 214–231. Springer, Heidelberg, December 2009.

OT10. Tatsuaki Okamoto and Katsuyuki Takashima. Fully secure functional encryption with general relations from the decisional linear assumption. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 191–208. Springer, Heidelberg, August 2010.

OT11. Tatsuaki Okamoto and Katsuyuki Takashima. Achieving short ciphertexts or short secret-keys for adaptively secure general inner-product encryption. In Dongdai Lin, Gene Tsudik, and Xiaoyun Wang, editors, *CANS 11*, volume 7092 of *LNCS*, pages 138–159. Springer, Heidelberg, December 2011.

OT12. Tatsuaki Okamoto and Katsuyuki Takashima. Adaptively attribute-hiding (hierarchical) inner product encryption. In Pointcheval and Johansson [PJ12], pages 591–608.

Par11. Jong Hwan Park. Inner-product encryption under standard assumptions. *Designs, Codes and Cryptography*, 58(3):235–257, 2011.

Pei09. Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In Michael Mitzenmacher, editor, *41st ACM STOC*, pages 333–342. ACM Press, May / June 2009.

PJ12. David Pointcheval and Thomas Johansson, editors. *EUROCRYPT 2012*, volume 7237 of *LNCS*. Springer, Heidelberg, April 2012.

Reg05. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

Sho94. Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. In *35th FOCS*, pages 124–134. IEEE Computer Society Press, November 1994.

SW05. Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *EUROCRYPT 2005*, volume 3494 of *LNCS*, pages 457–473. Springer, Heidelberg, May 2005.

Xag13. Keita Xagawa. Improved (hierarchical) inner-product encryption from lattices. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 235–252. Springer, Heidelberg, February / March 2013.

A Hierarchical Inner-Product Encryption

We use the definition proposed by Okamoto and Takashima [OT09]. We call a tuple of positive integers $\overrightarrow{\mu} := (n, d; \mu_1, ..., \mu_d)$ s.t $\mu_0 = 0 < \mu_1 < \mu_2 < ... < \mu_d = n$ a format of hierarchy of depth d attribute spaces. Let $\sum_\ell (\ell = 1, .., d)$ be the sets of attributes, where each $\sum_\ell := \mathbb{F}_q^{\mu_\ell - \mu_{\ell-1}} \backslash \{\overrightarrow{0}\}$. Let the hierarchical attributes $\sum := \bigcup_{\ell=1}^{d} (\sum_1 \times ... \times \sum_\ell)$, where the union is a disjoint union. Then, for $\overrightarrow{v}_i \in$

$\mathbb{F}_q^{\mu_\ell - \mu_{\ell-1}} \backslash \{\overrightarrow{0}\}$, the hierarchical predicate $f_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}(\overrightarrow{x}_1, ..., \overrightarrow{x}_h) = 1$ iff $\ell \leq h$ and $\overrightarrow{x}_i . \overrightarrow{v}_i = 0$ for all i s.t. $a \leq i \leq \ell$.

Let the space of hierarchical predicates $\mathcal{F} =: \{f_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)} | \overrightarrow{v}_i \in \mathbb{F}_q^{\mu_\ell - \mu_{\ell-1}} \backslash \{\overrightarrow{0}\}\}$. We call h (resp.ℓ) the level of $(\overrightarrow{x}_1, ..., \overrightarrow{x}_h)$ (resp.$(\overrightarrow{v}_1, ..., \overrightarrow{v}_h)$).

Definition 4. *Let $\overrightarrow{\mu} := (n, d; \mu_1, ..., \mu_d)$ s.t $\mu_0 = 0 < \mu_1 < \mu_2 < ... < \mu_d = n$ be a format of hierarchy of depth d attribute spaces. A hierarchical predicate encryption (HPE) scheme for the class of hierarchical inner-product predicates $\mathcal{F}$ over the set of hierarchical attributes $\sum$ consists of probabilistic polynomial time algorithms* $\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}$ *and* $\mathsf{Delegate}_\ell$ *for $\ell = 1, ..., d-1$. They are given as follows:*

- Setup *takes as input security parameter 1^λ and format of hierarchy $\overrightarrow{\mu}$, and outputs master public key* pp *and master secret key* msk.
- KeyGen *takes as input the* $\mathsf{pp}, \mathsf{msk}$*, and predicate vectors $(\overrightarrow{v}_1, ..., \overrightarrow{v}_\ell)$. It outputs a corresponding secret key* $\mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}$.
- Enc *takes as input the* pp*, attribute vectors $(\overrightarrow{x}_1, ..., \overrightarrow{x}_h)$, where $1 \leq h \leq d$, and plaintext m in some associated plaintext space,* $\mathbf{msg}$*. It returns ciphertext c.*
- Dec *takes as input the master public key* pp*, secret key* $\mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}$*, where $1 \leq \ell \leq d$, and ciphertext c. It outputs either plaintext m or the distinguished symbol $\perp$.*
- $\mathsf{Delegate}_\ell$ *takes as input the master public key* pp*, ℓ-th level secret key* $\mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}$*, and $(\ell+1)$-th level predicate vector $\overrightarrow{v}_{\ell+1}$. It returns $(\ell+1)$-th level secret key* $\mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_{\ell+1})}$.

Correctness. For all correctly generated pp and $\mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}$, generate $c \xleftarrow{R} \mathsf{Enc}(\mathsf{pp}, m, (\overrightarrow{x}_1, ..., \overrightarrow{x}_h))$ and $m' = \mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}, c)$. If $f_{(\overrightarrow{v}_1,...,\overrightarrow{v}_\ell)}(\overrightarrow{x}_1, ..., \overrightarrow{x}_h) = 1$, then $m' = m$. Otherwise, $m' \neq m$ except for negligible probability. For f and f' in $\mathcal{F}$, we denote $f' \leq f$ if the predicate vector for f is the prefix of that for f'.

Security. A hierarchical inner-product predicate encryption scheme for hierarchical predicates $\mathcal{F}$ over hierarchical attributes $\sum$ is $selectively\ attribute-hiding\ against\ plaintext\ attacks$ if for all probabilistic polynomial-time adversaries $\mathcal{A}$, the advantage of $\mathcal{A}$ in the following experiment is negligible in the security parameter.

1. $\mathcal{A}$ outputs challenge attribute vectors $X^{(0)} = (\overrightarrow{x}_1^{(0)}, ..., \overrightarrow{x}_h^{(0)}), X^{(1)} = (\overrightarrow{x}_1^{(1)}, ..., \overrightarrow{x}_h^{(1)})$.
2. Setup is run to generate keys pp and msk, and pp is given to $\mathcal{A}$.
3. $\mathcal{A}$ may adaptively makes a polynomial number of queries of the following type:
   - $\mathcal{A}$ asks the challenger to create a secret key for a predicate $f \in \mathcal{F}$. The challenger creates a key for f without giving it to $\mathcal{A}$.
   - $\mathcal{A}$ specifies a key for predicate f that has already been created, and asks the challenger to perform a delegation operation to create a child key for $f' \leq f$. The challenger computes the child key without giving it to the adversary.
   - $\mathcal{A}$ asks the challenger to reveal an already-created key for predicate f s.t. $f(X^{(0)}) = f(X^{(1)}) = 0$.

   Note that when key creation requests are made, $\mathcal{A}$ does not automatically see the created key. $\mathcal{A}$ see a key only when it makes a reveal key query.

4. $\mathcal{A}$ outputs challenge plaintexts m_0, m_1.
5. A random bit b is chosen. $\mathcal{A}$ is given $c \xleftarrow{R} \mathsf{Enc}(\mathsf{pp}, m_b, X^{(b)})$.
6. The adversary may continue to request keys for additional predicate vectors subject to the restrictions given in step 3.
7. $\mathcal{A}$ outputs a bit $b^{'}$, and succeeds if $b^{'} = b$.

We define the advantage of $\mathcal{A}$ as the quantity $\mathbf{Adv}_{\mathcal{A}}^{HIPE}(\lambda) = |Pr[b^{'} = b] - 1/2|$.

A.1 Our HIPE Scheme

We can apply the technical in our basic IPE scheme to optimize the hierarchical inner-product encryption scheme proposed by [Xag13]. For $i \in [d]$, we denote σ_i to be the Gaussian parameter used in the KeyGen and Delegate algorithm. Roughly speaking, we stack matrices $\mathbf{B}_{\boldsymbol{v}_i}$ to make IPE hierarchical.

$\mathsf{Setup}(1^\lambda, n, q, m, \boldsymbol{\mu})$. On input a security parameter λ, and an hierarchical format of depth d $\boldsymbol{\mu} = (\ell, d; \mu_1, ..., \mu_d)$:

1. $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, n, m)$.
2. Chose a random matrix $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and a random vector $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_q^n$.

Output $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \boldsymbol{u})$ and $\mathsf{msk} = \mathbf{T}_\mathbf{A}$.

$\mathsf{KeyGen}(\mathsf{pp}, \mathsf{msk}, \overrightarrow{\mathbf{V}})$: On input pp, msk and predicate vectors $\overrightarrow{\mathbf{V}} = (\boldsymbol{v}_1, ..., \boldsymbol{v}_j)$ where $\boldsymbol{v}_i = (v_{i,1}, ..., v_{i,\ell})$:

1. For all $i \in [j]$, define the matrix

$$\mathbf{V}_i^{'} = \begin{bmatrix} v_{i,1}\mathbf{I}_n \\ v_{i,2}\mathbf{I}_n \\ \vdots \\ v_{i,\ell}\mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\ell n \times n}, \quad \mathbf{V}_i = \mathbf{G}_{n\ell,\ell',m}^{-1}(\mathbf{V}_i^{'} \cdot \mathbf{G}_{n,2,m}) \in [\ell^{'}]^{m \times m}. \tag{24}$$

2. Set $\mathbf{A}_{\overrightarrow{\mathbf{V}}} = [\mathbf{A}|\mathbf{B}\mathbf{V}_1|.....|\mathbf{B}\mathbf{V}_j] \in \mathbb{Z}_q^{n \times (j+1)m}$.

3. Using the master secret key $\mathbf{T}_\mathbf{A}$ to construct short basis $\mathbf{E}_{\overrightarrow{\mathbf{V}}}^{'}$ for $\Lambda_q^{\perp}(\mathbf{A}_{\overrightarrow{\mathbf{V}}})$ by invoking the ExtBasis algorithm.

4. Using the algorithm $\mathsf{RandBasis}(\mathbf{E}_{\overrightarrow{\mathbf{V}}}^{'}, \sigma_j)$ to randomize $\mathbf{E}_{\overrightarrow{\mathbf{V}}}^{'}$, then outputs $\mathbf{E}_{\overrightarrow{\mathbf{V}}}$.

[Leo: since the basis of $\mathbf{A}_{\overrightarrow{\mathbf{V}}}$ is $2m \times 2m$, there might not be enough entropy to do the algorithm. But if you use the extendbasis introduced in the Bonsai tree paper, the trapdoor of $\mathbf{A}$ will be completely. Thus, I think this scheme is not correct.] Output $\mathsf{sk}_{\overrightarrow{\mathbf{V}}} = \mathbf{E}_{\overrightarrow{\mathbf{V}}}$.

[Wang: For your attack, since the definition of the HIPE needs that $\langle \boldsymbol{v}_i, \boldsymbol{w}_i \rangle = 0$ for all

$i \leq \ell$, so even if the adversary gets the secret key for $((1,...1),(-a_1b_1^{-1},...,-a_nb_n^{-1}))$, he still can not decrypt the ciphertext for attribute $((a_1,...,a_n),(b_1,...,b_n))$. The key point is that $\langle \boldsymbol{v}_i, \boldsymbol{w}_i \rangle = 0$ for all $i \leq \ell$ but not the summation of all the inner products of the predicate and attribute vectors equals to 0]

$\mathsf{Enc}(\mathsf{pp}, \overrightarrow{\mathbf{W}}, m)$: On input pp, a message $m \in \{0,1\}$, and attribute vectors $\overrightarrow{\mathbf{W}} = (\boldsymbol{w}_1,...,\boldsymbol{w}_h)$ where $\boldsymbol{w}_i = (w_{i,1},...,w_{i,\ell})$:

1. Choose a random vector $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$.
2. Set $\boldsymbol{c}_0 \leftarrow \boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T$, where $\boldsymbol{e}_0 \leftarrow \chi^m$.
3. Set $c \leftarrow \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor q/2 \rfloor m$, where $e \leftarrow \chi$.
4. For all $i \in [h]$, define the matrix

$$\mathbf{W}_i^{'} = \left[w_{i,1}\mathbf{I}_n \; w_{i,2}\mathbf{I}_n \; \dots \; w_{i,\ell}\mathbf{I}_n\right] \in \mathbb{Z}_q^{n \times \ell n}, \quad \mathbf{W}_i = \mathbf{W}_i^{'} \cdot \mathbf{G}_{n\ell,\ell',m} \in \mathbb{Z}_q^{n \times m}. \tag{25}$$

choose a random matrix $\mathbf{R}_i \xleftarrow{\$} \{-1,1\}^{m \times m}$ and set $\boldsymbol{c}_i \leftarrow \boldsymbol{s}^T(\mathbf{B} + \mathbf{W}_i^{'}) + \boldsymbol{e}_0^T\mathbf{R}_i$.

Output $\mathsf{ct} = (\boldsymbol{c}_0, \{\boldsymbol{c}_i\}, c)$.

$\mathsf{Dec}(\mathsf{pp}, \mathsf{sk}_{\overrightarrow{\mathbf{V}}}, \mathsf{ct})$: On input pp, a decryption key $\mathsf{sk}_{\overrightarrow{\mathbf{V}}}$ where $\overrightarrow{\mathbf{V}} = (\boldsymbol{v}_1,...,\boldsymbol{v}_j)$, and a ciphertext $\mathsf{ct} = (\boldsymbol{c}_0, \{\boldsymbol{c}_i\}, c)$:

1. For $i \in [j]$, set $\boldsymbol{c}_{\boldsymbol{v}_i} = \boldsymbol{c}_i \cdot \mathbf{V}_i$.
2. Let $\boldsymbol{c} \leftarrow [\boldsymbol{c}_0, \boldsymbol{c}_{\boldsymbol{v}_1},..., \boldsymbol{c}_{\boldsymbol{v}_j}] \in \mathbb{Z}_q^{(j+1)m}$.
3. Set $\tau_t = \sigma_t.\sqrt{(t+1)m}.\omega(\sqrt{(t+1)m})$, Then $\tau_t \geq \|\widetilde{\mathbf{E}_{\overrightarrow{\mathbf{V}}}}\| \cdot \omega(\sqrt{(t+1)m})$.
4. Compute $\boldsymbol{x}_{\overrightarrow{\mathbf{V}}} \leftarrow \mathsf{SamplePre}(\mathbf{A}_{\overrightarrow{\mathbf{V}}}, \mathbf{E}_{\overrightarrow{\mathbf{V}}}, \boldsymbol{u}, \tau_t)$.
5. Compute $z = c - \boldsymbol{c} \cdot \boldsymbol{x}_{\overrightarrow{\mathbf{V}}} \pmod q$, and output $\lfloor (2/q)z \rceil \in \{0,1\}$.

$\mathsf{Delegate}(\mathsf{pp}, \mathsf{sk}_{\overrightarrow{\mathbf{V}}}, \mathbf{V}^{'})$: On input pp, a decryption key $\mathsf{sk}_{\overrightarrow{\mathbf{V}}} = \mathbf{E}_{\overrightarrow{\mathbf{V}}}$, where $\overrightarrow{\mathbf{V}} = (\boldsymbol{v}_1,...,\boldsymbol{v}_j)$, and $\overrightarrow{\mathbf{V}}' = (\boldsymbol{v}_1,...,\boldsymbol{v}_j,\boldsymbol{v}_{j+1},...,\boldsymbol{v}_t)$, $\boldsymbol{v}_i = (v_{i,1},...,v_{i,\ell})$. do:

1. For all $i \in [t]$, define the matrices

$$\mathbf{V}_i^{'} = \begin{bmatrix} v_{i,1}\mathbf{I}_n \\ v_{i,2}\mathbf{I}_n \\ \vdots \\ v_{i,l}\mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\ell n \times n}, \quad \mathbf{V}_i = \mathbf{G}_{n\ell,\ell',m}^{-1}(\mathbf{V}_i^{'} \cdot \mathbf{G}_{n,2,m}) \in [\ell^{'}]^{m \times m}. \tag{26}$$

2. Set $\mathbf{A}_{\overrightarrow{\mathbf{V}}'} = [\mathbf{A}|\mathbf{B}\mathbf{V}_1|.....|\mathbf{B}\mathbf{V}_t] \in \mathbb{Z}_q^{n \times (t+1)m}$.

3. Recall that the secret key $\mathbf{E}_{\overrightarrow{\mathbf{V}}}$ is a short basis for $\Lambda_q^{\perp}(\mathbf{A}_{\overrightarrow{\mathbf{V}}})$. Using it to construct a short basis for $\Lambda_q^{\perp}(\mathbf{A}_{\overrightarrow{\mathbf{V}}'})$ by invoking

$$\mathbf{E}'_{\overrightarrow{\mathbf{V}}'} \leftarrow \mathsf{ExtBasis}(\mathbf{A}_{\overrightarrow{\mathbf{V}}}, [\mathbf{B}\mathbf{V}_{j+1}|...|\mathbf{B}\mathbf{V}_t], \mathbf{E}_{\overrightarrow{\mathbf{V}}}). \tag{27}$$

4. Using the algorithm $\mathsf{RandBasis}(\mathbf{E}'_{\overrightarrow{\mathbf{V}}'}, \sigma_t)$ to randomize $\mathbf{E}'_{\overrightarrow{\mathbf{V}}'}$, then outputs $\mathbf{E}_{\overrightarrow{\mathbf{V}}'}$.

Output $\mathsf{sk}_{\overrightarrow{\mathbf{V}}'} = \mathbf{E}_{\overrightarrow{\mathbf{V}}'}$ **Correctness.** It's easy to verify that $\boldsymbol{c}_{\boldsymbol{v}_i} = \boldsymbol{s}^T\mathbf{B}\mathbf{V}_i + \boldsymbol{e}_0^T\mathbf{R}_i\mathbf{V}_i$, so $\boldsymbol{c} = \boldsymbol{s}^T\mathbf{A}_{\overrightarrow{\mathbf{V}}} + [\boldsymbol{e}_0^T|\boldsymbol{e}_0^T\mathbf{R}_1\mathbf{V}_1|...|\boldsymbol{e}_0^T\mathbf{R}_i\mathbf{V}_i]$. Then $z = c - \boldsymbol{c}\cdot\boldsymbol{x}_{\overrightarrow{\mathbf{V}}} = \lfloor q/2 \rfloor m + \underbrace{e - [\boldsymbol{e}_0^T|\boldsymbol{e}_0^T\mathbf{R}_1\mathbf{V}_1|...|\boldsymbol{e}_0^T\mathbf{R}_i\mathbf{V}_i]\boldsymbol{x}_{\overrightarrow{\mathbf{V}}}}_{\boldsymbol{e}}$ (mod q). If the norm of the error term $\boldsymbol{e}$ is small, our HIPE scheme is correct.

Security. We omit the security proof of our HIPE scheme, since it's very similar to our basic IPE scheme and [ADM12]. We also omit the parameters setting.

B Fuzzy Identity-based Encryption

In this section, we construct a FIBE scheme from our IPE scheme, we first introduce the definition and the security model of Fuzzy IBE.

A Fuzzy Identity Based encryption scheme consists of the following four algorithms:

- $\mathsf{Fuzzy.Setup}(\lambda, \ell) \rightarrow (\mathsf{pp}, \mathsf{msk})$: The algorithm takes as input the security parameter λ and the maximum length of identities ℓ. It outputs the public parameters pp, and the master secret key msk.
- $\mathsf{Fuzzy.Extract}(\mathsf{msk}, \mathsf{pp}, id, k)$: This algorithm takes as input the master key msk, the public parameters pp, an identity id and the threshold $k \leq \ell$. It outputs a decryption key sk_{id}.
- $\mathsf{Fuzzy.Enc}(\mathsf{pp}, m, id') \rightarrow \mathsf{ct}_{id'}$: The algorithm takes as input: a message bit m, an identity id', and the public parameters pp. It outputs the ciphertext $\mathsf{ct}_{id'}$.
- $\mathsf{Fuzzy.Dec}(\mathsf{pp}, \mathsf{ct}_{id'}, \mathsf{sk}_{id}) \rightarrow m$: This algorithm takes as input the ciphertext $\mathsf{ct}_{id'}$, the decryption key sk_{id} and the public parameters pp. It outputs the message m if $|id \bigcap id'| \geq k$.

Security. We follow the Selective-ID model of security of Fuzzy Identity Based Encryption as given by Sahai and Waters[SW05].

- **Target**: The adversary declares the challenge identity, id^*, and he wishes to be challenged upon.
- **Setup**: The challenger runs the Setup algorithm of Fuzzy-IBE and gives the public parameters to the adversary.
- **Phase 1**: The adversary is allowed to issue queries for private keys for identities id_j of its choice, as long as $|id_j \bigcap id^*| < k; \forall j$

– **Challenge**: The adversary submits a message to encrypt. the challenger encrypts the message with the challenge id^* and then flips a random coin r. If $r = 1$, the ciphertext is given to the adversary, otherwise a random element of the ciphertext space is returned.
– **Phase 2**: Phase 1 is repeated.
– **Guess**: the adversary outputs a guess $r^{'}$ of r. the advantage of an adversary A in this game is defined as $|Pr[r^{'} = r] - 1/2|$.

A Fuzzy Identity Based Encryption scheme is secure in the Selective-Set model of security if all polynomial time adversaries have at most a negligible advantage in the Selective-Set game.

We introduce the embedding of exact threshold by Katz, Sahai, and Waters[KSW08].

Exact threshold: For binary vector $\boldsymbol{x} \in \{0,1\}^N$, $H_w(\boldsymbol{x}$ denotes the Hamming weight of $\boldsymbol{x}$. For binary vectors $\boldsymbol{a}, \boldsymbol{x} \in \{0,1\}^n$, the exact threshold predicate is denoted by $\mathcal{P}^{th}_{=t}(\boldsymbol{a}, \boldsymbol{x})$ and output 1 if and only if $H_w(\boldsymbol{a}\&\boldsymbol{x}) = t$, where $\&$ denotes the logical conjunction. Suppose that $t < q$. Set $\mu = N+1$, $\boldsymbol{v} = (\boldsymbol{a}, 1) \in \mathbb{Z}_q^\mu$, and $\boldsymbol{w} = (\boldsymbol{x}, -t) \in \mathbb{Z}_q^\mu$. We have that $\langle \boldsymbol{v}, \boldsymbol{w} \rangle = 0$ if and only if $H_w(\boldsymbol{a}\&\boldsymbol{x}) = t$.

B.1 Our FIBE scheme

Now, we use our basic IPE scheme to construct a FIBE scheme. Let $\{0,1\}^N$ be a space of identities. The threshold predicate over $\{0,1\}^N$ is defined by $\mathcal{P}^{th}_{\geq t}(\boldsymbol{a}, \boldsymbol{x})$ and output 1 if and only if $H_w(\boldsymbol{a}\&\boldsymbol{x}) \geq t$.

It's easy to see that the above predicate can be written as $\bigcup_{i=t}^{N} \mathcal{P}^{th}_{i=t}(\boldsymbol{a}, \boldsymbol{x})$. Hence, we can implement a FIBE scheme in a lazy way by repeating ciphertexts of an IPE scheme that supports the relations $\mathcal{P}^{th}_{\geq t}$ for $i = t, ..., N$.

$\mathsf{Fuzzy.Setup}(1^\lambda, 1^N)$: On input a security parameter λ, and identity size N, do:

1. $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^\lambda, q, n, m)$.
2. Chose a random matrix $\mathbf{B} \xleftarrow{\$} \mathbb{Z}_q^{n \times m}$ and a random vector $\boldsymbol{u} \xleftarrow{\$} \mathbb{Z}_q^n$.

Output $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \boldsymbol{u})$ and $\mathsf{msk} = \mathbf{T_A}$.

$\mathsf{Fuzzy.Extract}(\mathsf{pp}, \mathsf{msk}, id, t)$: On input public parameters pp, a master key msk, an identity $id = (a_1, ..., a_N) \in \{0,1\}^N$ and threshold $t \leq N$, do:

1. Set vector $\boldsymbol{v} = (a_1, ..., a_N, 1) \in \mathbb{Z}_q^\mu$.
2. Define the matrix

$$\mathbf{V}^{'} = \begin{bmatrix} a_1\mathbf{I}_n \\ a_2\mathbf{I}_n \\ \vdots \\ a_N\mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\mu n \times n}, \quad \mathbf{V} = \mathbf{G}^{-1}_{\mu n, \ell^{'}, m}(\mathbf{V}_i^{'} \cdot \mathbf{G}_{n,2,m}) \in [\ell^{'}]^{m \times m}. \tag{28}$$

3. Define the matrix $\mathbf{U} = \mathbf{BV} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{A}_{id} = [\mathbf{A}|\mathbf{U}]$.

4. Using the master secret key $\mathsf{msk} = (\mathbf{T_A}, \sigma)$, compute $\boldsymbol{r} \leftarrow \mathsf{SampleLeft}\,(\mathbf{A}, \mathbf{T_A}, \mathbf{U}, \boldsymbol{u}, \sigma)$. Then $\boldsymbol{r}$ is a vector in $\mathbb{Z}^{2m}$ satisfying $\mathbf{A}_{id} \cdot \boldsymbol{r} = \boldsymbol{u} \pmod q$.

Output the secret key $\mathsf{sk}_{id} = \boldsymbol{r}$.

$\mathsf{Fuzzy.Enc}(\mathsf{pp}, id^{'}, m)$: On input public parameters pp, an identity $id^{'} = (x_1, ..., x_N) \in \{0,1\}^N$, and a message $m \in \{0,1\}$, do:

1. Define a sequence of vectors $\boldsymbol{w}_i = (x_1, ..., x_N, -t-i+1) \in \mathbb{Z}_q^{\mu n\times n}, i = 1, ..., N-t+1$.
2. Choose a uniformly random $\boldsymbol{s} \xleftarrow{\$} \mathbb{Z}_q^n$.

3. Choose a noise vector $\boldsymbol{e}_0 \leftarrow D_{\mathbb{Z}_q^m,\alpha}$ and a noise term $e \leftarrow D_{\mathbb{Z}_q,\alpha}$.

4. Compute $\boldsymbol{c}_0 = \boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T$.

5. For all $i \in [N-t+1]$ Define the matrix

$$\mathbf{W}_i^{'} = \left[x_1\mathbf{I}_n\ x_2\mathbf{I}_n\ \ldots\ x_N\mathbf{I}_n\ -(t+i-1)\mathbf{I}_n\right] \in \mathbb{Z}_q^{n\times\mu n},\quad \mathbf{W}_i = \mathbf{W}_i^{'}\cdot\mathbf{G}_{\mu n,\ell',m} \in \mathbb{Z}_q^{n\times m}. \tag{29}$$

Pick a sequence of random matrices $\mathbf{R}_i \xleftarrow{\$} \{-1,1\}^{m\times m}, i = 1, ..., N-t+1$, define error vectors $\boldsymbol{e}_i^T = \boldsymbol{e}_0^T\mathbf{R}_i$. Set

$$\boldsymbol{c}_i = \boldsymbol{s}^T(\mathbf{B}+\mathbf{W}_i) + \boldsymbol{e}_i^T,\quad c = \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor\frac{q}{2}\rfloor m. \tag{30}$$

Output ciphertext $\mathsf{ct}_{id'} = (\boldsymbol{c}_0, \{\boldsymbol{c}_i\}, c)$.

$\mathsf{Fuzzy.Dec}(\mathsf{pp}, \mathsf{sk}_{id}, \mathsf{ct}_{id'})$: On input public parameters pp, a decryption key sk_{id}, and a ciphertext $\mathsf{ct}_{id'}$, do:

1. Let $H_w(id\&id^{'})$ denotes Hamming weight of the logical conjunction of id and $id^{'}$. If $H_w(id\&id^{'}) = k < t$, output $\perp$. Otherwise, we parse the $\mathsf{ct}_{id'} = (\boldsymbol{c}_0, \{\boldsymbol{c}_i\}, c)$, and compute the matrix $\mathbf{V}$ for id as above.

2. Compute $\tilde{\boldsymbol{c}}_{k-t+1} = \boldsymbol{c}_{k-t+1}\cdot\mathbf{V}$, let $\boldsymbol{c} = [\boldsymbol{c}_0|\tilde{\boldsymbol{c}}_{k-t+1}]$.

3. Compute $z = c - \boldsymbol{c}\cdot\boldsymbol{r} \pmod q$.

Output 0 if $|z| < q/4$ and 1 otherwise.

Correctness. We just consider the case $H_w(id\&id^{'}) = k \geq t$. $\tilde{\boldsymbol{c}}_{k-t+1} = \boldsymbol{c}_{k-t+1}\cdot\mathbf{V} = \boldsymbol{s}^T\mathbf{BV} + \boldsymbol{s}^T\mathbf{W}_{k-t+1}\mathbf{V} + \boldsymbol{e}_{k-t+1}^T\mathbf{V}$. It's easy to see that if $H_w(id\&id^{'}) = k$, $\mathbf{W}_{k-t+1}\mathbf{V} = \mathbf{0}$, so $\tilde{\boldsymbol{c}}_{k-t+1} = \boldsymbol{s}^T\mathbf{BV} + \boldsymbol{e}_{k-t+1}^T\mathbf{V}$. Then $\boldsymbol{c} = \boldsymbol{s}^T\mathbf{A}_{id} + [\boldsymbol{e}_0^T|\boldsymbol{e}_{k-t+1}^T\mathbf{V}]$, and $z = c - \boldsymbol{c}\cdot\boldsymbol{r} (\text{mod } q) = \lfloor q/2\rfloor m + \underbrace{e - [\boldsymbol{e}_0^T|\boldsymbol{e}_{k-t+1}^T\mathbf{V}]\boldsymbol{r}}_{\boldsymbol{e}} (mod\ q)$. If the error term $\boldsymbol{e}$

is small, our scheme is correct.

Security. We sketch the proof of the Selective-ID security of the FIBE scheme described above. We have the following theorem:

Theorem 5. *The FIBE scheme above is selectively secure provided that decision-*$\mathsf{LWE}_{n,q,\chi}$ *assumption holds.*

Proof(sketch). We propose a sequence of games where the first game is identical to the real security game from the definition. In the last game in the sequence the adversary has advantage zero. We show that a PPT adversary $\mathcal{A}$ cannot distinguish between the games which will prove that the adversary has negligible advantage in winning the security game. The LWE problem is used in proving that $\mathbf{Game\ 3}$ and $\mathbf{Game\ 4}$ are indistinguishable.

$\mathbf{Game\ 0}$. It's identical to the real game.

$\mathbf{Game\ 1}$. We slightly change the way that the challenger generates pp. Let $id^* = (x_1^*, ..., x_N^*)$ be the challenge identity. The challenger chooses a random matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a random vector $\boldsymbol{u} \in \mathbb{Z}^n$, and chooses $\mathbf{R}^* \in \{-1, 1\}^{m\times m}$. Define the matrix

$$\mathbf{W}^{'} = \left[x_1^*\mathbf{I}_n\ x_2^*\mathbf{I}_n\ \ldots\ x_N^*\mathbf{I}_n\ -t\mathbf{I}_n\right] \in \mathbb{Z}_q^{n\times\mu n}, \tag{31}$$

and set $\mathbf{B} = \mathbf{A}\mathbf{R}^* - \mathbf{W}^{'} \cdot \mathbf{G}_{\mu n,\ell',m} \in \mathbb{Z}_q^{n\times m}$. Outputs $\mathsf{pp} = (\mathbf{A}, \mathbf{B}, \boldsymbol{u})$, and sends pp to $\mathcal{A}$.

Game 2. We change the way that the challenger answer the key queries. If the adversary $\mathcal{A}$ submits the key query for $id = (a_1, ..., a_N)$, challenger first checks $H_w(id\&id^{'}) < t$, if so, define the matrix

$$\mathbf{V}^{'} = \begin{bmatrix} a_1\mathbf{I}_n \\ a_2\mathbf{I}_n \\ \vdots \\ a_N\mathbf{I}_n \\ \mathbf{I}_n \end{bmatrix} \in \mathbb{Z}_q^{\mu n\times n},\ \ \mathbf{V} = \mathbf{G}^{-1}_{\mu n,\ell',m}(\mathbf{V}^{'} \cdot \mathbf{G}_{n,2,m}). \tag{32}$$

Set $\mathbf{U} = \mathbf{B}\mathbf{V}, \mathbf{A}_{id} = [\mathbf{A}|\mathbf{U}] = [\mathbf{A}|\mathbf{A}\mathbf{R}^*\mathbf{V} - \mathbf{W}^{'}\mathbf{V}^{'}\mathbf{G}_{n,2,m}]$.

Let $\boldsymbol{r} \leftarrow \mathsf{SampleRight}\ (\mathbf{A}, \mathbf{R}^*\mathbf{V}, \mathbf{W}^{'}\mathbf{V}^{'}, \mathbf{T}_{\mathbf{G}_{n,2,m}}, \boldsymbol{u}, \sigma)$, so that $\mathbf{A}_{id} \cdot \boldsymbol{r} = \boldsymbol{u}$. Outputs $\mathsf{sk}_{id} = \boldsymbol{r}$, and sends it to $\mathcal{A}$. Otherwise, outputs $\perp$, and aborts the game.

$\mathbf{Game\ 3}$. Change the way to generate the challenge ciphertext. We use $\mathbf{R}^*$ as the matrix $\mathbf{R}_1$ used to generate $\boldsymbol{c}_1$, the remainder of the game is identical to $\mathbf{Game\ 2}$.

$\mathbf{Game\ 4}$. We just choose the challenge ciphertext as a random independent element from the ciphertext space. So the advantage of $\mathcal{A}$ is zero.

It easy to see in $\mathbf{Game\ 3}$, $\boldsymbol{c}_0 = \boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T$, $\boldsymbol{c}_1 = (\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T)\mathbf{R}^*$. $c = \boldsymbol{s}^T\boldsymbol{u} + e + \lfloor q/2\rfloor m$, $\boldsymbol{c}_i = (\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T)\mathbf{R}^* + \boldsymbol{s}^T[\mathbf{0}, ..., \mathbf{0}, -(i-1)\mathbf{I}_n] \cdot \mathbf{G}_{\mu n,\ell',m} + \boldsymbol{e}_i^T - \boldsymbol{e}_0^T\mathbf{R}^*, i = 2, ..., N - t + 1$. If $\boldsymbol{s}^T\mathbf{A} + \boldsymbol{e}_0^T$ and $\boldsymbol{s}^T\boldsymbol{u} + e$ are LWE instances, Then it simulates $\mathbf{Game\ 3}$, if they are random variables, it simulates $\mathbf{Game\ 4}$. □