Predicate Encryption for Multi-dimensional Range Queries from Lattices

Romain Gay(✉), Pierrick Méaux, and Hoeteck Wee

ENS, Paris, France
{rgay,meaux,wee}@di.ens.fr

Abstract. We construct a lattice-based predicate encryption scheme for multi-dimensional range and multi-dimensional subset queries. Our scheme is selectively secure and weakly attribute-hiding, and its security is based on the standard learning with errors (LWE) assumption. Multi-dimensional range and subset queries capture many interesting applications pertaining to searching on encrypted data. To the best of our knowledge, these are the first lattice-based predicate encryption schemes for functionalities beyond IBE and inner product.

1 Introduction

Predicate encryption [8,17,20] is a new paradigm for public-key encryption that supports search queries on encrypted data. In predicate encryption, ciphertexts are associated with descriptive values x in addition to a plaintext, secret keys are associated with a query predicate f, and a secret key decrypts the ciphertext to recover the plaintext if and only if $f(x) = 1$. The security requirement for predicate encryption enforces privacy of x and the plaintext even amidst multiple search queries, namely an adversary holding secret keys for different query predicates learns nothing about x and the plaintext if none of them is individually authorized to decrypt the ciphertext.

Multi-dimensional Range Queries. Following [8,20], we focus on predicate encryption for multi-dimensional range queries, as captured by the following examples:

– For network intrusion detection on logfiles, we would encrypt network flows labeled with a set of attributes from the network header, such as the source and destination addresses, port numbers, time-stamp, and protocol numbers. We could then issue auditors with restricted secret keys that can only decrypt the network flows that fall within a particular range of IP addresses and some specific time period.

R. Gay— Supported in part by ANR-14-CE28-0003 (Project EnBiD).
P. Méaux— INRIA and ENS. Supported in part by ANR-13-JS02-0003 (Project CLE).
H. Wee— ENS and CNRS. Supported in part by ANR-14-CE28-0003 (Project EnBiD), NSF Award CNS-1445424, ERC Project CryptoCloud (FP7/2007-2013 Grant Agreement no. 339563), the Alexander von Humboldt Foundation and a Google Faculty Research Award.

J. Katz (Ed.): PKC 2015, LNCS 9020, pp. 752–776, 2015.
DOI: 10.1007/978-3-662-46447-2_34

- For credit card fraud investigation, we would encrypt credit card transactions labeled with a set of attributes such as time, costs and zipcodes. We could then issue investigators with restricted secret keys that decrypt transactions over $1,000 which took place in the last month and originated from a particular range of zipcodes.
- For online dating, we would encrypt personal profiles labeled with dating preferences pertaining to age, height, weight and salary. Secret keys are associated with specific attributes and can only decrypt profiles for which the attributes match the dating preferences.

More generally, in multi-dimensional range queries, we are given a point $(z_1, \ldots, z_D) \in [T]^D$ and interval ranges $[x_1, y_1], \ldots, [x_D, y_D] \subseteq [T]$ and we want to know if $(x_1 \leq z_1 \leq y_1) \wedge \cdots \wedge (x_D \leq z_D \leq y_D)$. We also consider more general multi-dimensional subset queries where we are given subset $S_1, \ldots, S_D \subseteq [T]$ and we want to know if $(z_1 \in S_1) \wedge \cdots \wedge (z_D \in S_D)$. Note that in the first two examples, the search queries are associated with the keys, whereas in the third example, the search queries are associated with the ciphertext. We will refer to encryption schemes for the former as "key-policy" schemes, and schemes for the latter as "ciphertext-policy" schemes. In all of these examples, it is important that unauthorized parties do not learn the contents of the ciphertexts, nor of the meta-data associated with the ciphertexts, such as the network header or dating preferences. On the other hand, it is often okay to leak the meta-data to authorized parties. We stress that privacy of the meta-data is an additional security requirement provided by predicate encryption but not attribute-based encryption [14,15].

Prior Works. The first constructions of predicate encryption for multi-dimensional range queries were given in the works of Boneh and Waters [8] and Shi, et. al [20]; all of these constructions rely on bilinear groups and achieve parameters that are linear in the number of dimensions D. The latter work also presents a generic "brute force" construction based on any anonymous IBE, where the parameters grow exponentially in D.

1.1 Our Contributions

In this work, we construct a lattice-based predicate encryption scheme for multi-dimensional range queries, whose security is based on the standard learning with errors (LWE) assumption. Our scheme is selectively secure and weakly attribute-hiding, that is, we only guarantee privacy of the ciphertext attribute against collusions that are not authorized to decrypt the challenge ciphertext. The scheme is best suited for applications where the range T is very large but the number of dimensions D is a small constant, as is the case for the three applications outlined earlier and also the scenario considered in [20]. In addition, we extend our techniques to multi-dimensional subset queries, where we obtain improved efficiency over prior schemes [8]. We summarize our schemes in Table 1 and 2.

Our Approach. At a high level, our approach follows that of Shi et al. [20], who showed how to boost an anonymous IBE scheme into a predicate encryption scheme for multi-dimensional range queries. We show how to carry out a similar transformation over lattices, starting from the LWE-based anonymous IBE schemes in [1,4,9]. We highlight the main novelties in this work:

- First, we present a more modular and conceptually simpler approach for handling multi-dimensional range queries. We construct our scheme for the simpler AND-OR-EQ predicate (conjunction of disjunction of equalities), and present a combinatorial reduction from multi-dimensional range queries to this predicate. The simpler AND-OR-EQ predicate is symmetric, which immediately yields both key-policy and ciphertext-policy schemes for multi-dimensional range queries. We can only prove security for our lattice-based AND-OR-EQ predicate encryption under an "at most one" promise, which necessitates a more delicate reduction from multi-dimensional range queries to AND-OR-EQ where we decompose range queries into disjoint sub-intervals. Indeed, the same technical issue arises in the previous pairing-based schemes.
- To handle the inner disjunction of equality like $(X = a) \vee (X = b)$ for the key-policy AND-OR-EQ predicate where X is associated with the ciphertext and (a, b) with the key, our new ciphertext is an anonymous IBE ciphertext for the identity X, and the key comprises two IBE keys for a and b; decryption works by trying all possible IBE keys. Following [20], we will pad the plaintext with zeroes, so that we know which of the decryptions corresponds to the correct plaintext. To handle the outer conjunction, we rely on secret sharing, as with prior lattice-based fuzzy IBE [3].
- Correctness for the inner disjunction requires more care than that in the bilinear groups. Roughly speaking, we need to show that decrypting an IBE ciphertext for identity a with a key for identity $b \neq a$ yields a random-looking value. The straight-forward argument that relies on IBE security yields computational correctness. To achieve statistical correctness in the lattice-based setting, we rely on a simple but seemingly novel analysis of the output of lattice-based trapdoor sampling algorithms (c.f. Lemma 13).
- The intuition for security for the inner disjunction is as follows: if X is different from a and b, then both X and the plaintext remain hidden via security of the underlying IBE. On the other hand, if X is equal to one of a, b, then the decryptor does learn the exact value of X, which means that we cannot hope to achieve strong attribute-hiding using these techniques. To establish the weak attribute-hiding for the general AND-OR-EQ, we rely on techniques from lattice-based inner product encryption [4].

To the best of our knowledge, this is the first lattice-based predicate encryption scheme for functionalities beyond IBE and inner product [1,4,9,21], and we hope that it would inspire further research into lattice-based predicate encryption. We defer a more detailed overview of our construction to Section 4.

Table 1. Summary of existing predicate encryption schemes for multi-dimensional range queries: given a point $(z_1, \ldots, z_D) \in [T]^D$ and interval ranges $[x_1, y_1]$, $\ldots, [x_D, y_D] \subseteq [T]$, we want to know if $(x_1 \leq z_1 \leq y_1) \wedge \cdots \wedge (x_D \leq z_D \leq y_D)$. Here, D denotes the number of dimensions and T the number of points in each dimension. We omit the poly(n) multiplicative overhead, where n is the security parameter.

Reference	Size			Time		Attribute
	Public key	Ciphertext	Secret key	Encryption	Decryption	hiding
[8] (KP)	$O(D \cdot T)$	$O(D \cdot T)$	$O(D)$	$O(D \cdot T)$	$O(D)$	fully
[20] (KP, CP)	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O((\log T)^D)$	weakly
this paper (KP, CP)	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O(D \log T)$	$O((\log T)^D)$	weakly

Table 2. Summary of existing predicate encryption schemes for multi-dimensional subset queries: given a point $(z_1, \ldots, z_D) \in [T]^D$ and subsets $S_1, \ldots, S_D \subseteq [T]$, we want to know if $(z_1 \in S_1) \wedge \cdots \wedge (z_D \in S_D)$. Here, D denotes the number of dimension and T the size of the sets. We omit the poly(n) multiplicative overhead, where n is the security parameter. (KP) stands for key-policy and (CP) stands for ciphertext-policy.

Reference	Size			Time		Attribute
	Public key	Ciphertext	Secret key	Encryption	Decryption	hiding
[8]	$O(D \cdot T)$	$O(D \cdot T)$	$O(D \cdot T)$	$O(D \cdot T)$	$O(D \cdot T)$	fully
this paper (KP)	$O(D)$	$O(D)$	$O(D \cdot T)$	$O(D)$	$O(T^D)$	weakly
this paper (CP)	$O(D \cdot T)$	$O(D \cdot T)$	$O(D)$	$O(D \cdot T)$	$O(D)$	weakly

Organization. The rest of the paper is organized as follows. We recall the relevant background on lattices and the security model of predicate encryption in Section 2. We introduce the so-called AND-OR-EQ predicate, and show how to reduce multi-dimensional subset queries and multi-dimensional range queries to AND-OR-EQ in Section 3. We give our lattice-based predicate encryption scheme for AND-OR-EQ in Section 4, and we show that it gives an efficient construction for multi-dimensional range queries. Finally, we show in Section 5 how to improve the construction of Section 4 in order to obtain an efficient scheme for multi-dimensional subset queries.

2 Preliminaries

Notations. We denote by $s \leftarrow_{\mathrm{R}} S$ the fact that s is picked uniformly at random from a finite set S or from a distribution. By PPT, we denote a probabilistic polynomial-time algorithm. Throughout this paper, we use 1^n as the security parameter. For every vector $\mathbf{u} \in \mathbb{Z}_q^n$, we write $\mathbf{u} = (u_1, \ldots, u_n)$, and for any matrix $\mathbf{M} \in \mathbb{Z}_q^{n \times m}$, we write $M_{i,j}$ the (i, j) entry of $\mathbf{M}$. For any $x \in \mathbb{R}$, we denote by $\lfloor x \rfloor$ the largest integer less than or equal to x. For any $z \in [0, 1]$, we denote by $\lfloor z \rceil$ the closest integer to z.

Randomness Extraction. We use the following variant of the left-over hash lemma [11,16] from [1]:

Lemma 1 ([1], Lemma 13). *Let $m > (n+1)\log q + \omega(\log n)$, $q > 2$ be a prime number, $\mathbf{R} \leftarrow_{\mathrm{R}} \{-1,1\}^{m \times \ell} \mod q$, where $\ell = \ell(n)$ is polynomial in n. Let $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}, \mathbf{B} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times \ell}$; For all vectors $\mathbf{u} \in \mathbb{Z}^m$ the distribution $(\mathbf{A}, \mathbf{AR}, \mathbf{u}^\top \mathbf{R})$ is statistically close from the distribution $(\mathbf{A}, \mathbf{B}, \mathbf{u}^\top \mathbf{R})$.*

LWE Assumption. The decisional Learning With Error problem (dLWE) was introduced by Regev [19],

Definition 1 (dLWE). *For an integer $q = q(n) \geq 2$, an adversary $\mathcal{A}$ and an error distribution $\chi = \chi(n)$ over $\mathbb{Z}_q$, we define the following advantage function:*

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\chi}} := |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{z}_0) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{z}_1) = 1]|$$

where

$$\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}, \mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n, \mathbf{e} \leftarrow_{\mathrm{R}} \chi^m, \mathbf{z}_0 := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top \quad \textit{and} \quad \mathbf{z}_1 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^m$$

The $\mathsf{dLWE}_{n,m,q,\chi}$ assumption asserts that for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}_{\mathcal{A}}^{\mathsf{dLWE}_{n,m,q,\chi}}$ is a negligible function in n.

Throughout the paper, we denote by $\chi_{\max} < q$ the bound on the noise distribution χ.

2.1 Lattice Preliminaries

Lattices. For any matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and any vector $\mathbf{p} \in \mathbb{Z}_q^n$, we define the orthogonal q-ary lattice of $\mathbf{A}$: $\Lambda_q^\perp(\mathbf{A}) := \{\mathbf{u} \in \mathbb{Z}^m : \mathbf{Au} = \mathbf{0} \mod q\}$ and the shifted lattice: $\Lambda_q^{\mathbf{p}}(\mathbf{A}) := \{\mathbf{u} \in \mathbb{Z}^m : \mathbf{Au} = \mathbf{p} \mod q\}$. Similarly, for any matrix $\mathbf{P} \in \mathbb{Z}_q^{n \times \ell}$, we define: $\Lambda_q^{\mathbf{P}}(\mathbf{A}) := \{\mathbf{U} \in \mathbb{Z}^{m \times \ell} : \mathbf{AU} = \mathbf{P} \mod q\}$.

Matrix Norms. For any vector $\mathbf{x}$, we denote by $\|\mathbf{x}\|$ its ℓ_2 norm. For any matrix $\mathbf{R} \in \mathbb{Z}_q^{n \times \ell}$ we define the three following norms:

1. $\|\mathbf{R}\|$ denotes the maximum of the ℓ_2 norm over the columns of $\mathbf{R}$.
2. $\|\mathbf{R}\|_{\mathrm{GS}}$ denotes the Gram-Schmidt norm of $\mathbf{R}$ (see [7] for further details).
3. $\|\mathbf{R}\|_2$ denotes the operator norm of $\mathbf{R}$ defined as $\|\mathbf{R}\|_2 = \sup_{\|\mathbf{x}\|=1} \|\mathbf{Rx}\|$.

Note that for any matrix $\mathbf{S} \in \mathbb{Z}_q^{\ell \times m}$, and any vector $\mathbf{e} \in \mathbb{Z}_q^n$, we have $\|\mathbf{R} \cdot \mathbf{S}\| \leq \|\mathbf{R}\|_2 \cdot \|\mathbf{S}\|$, and $\|\mathbf{e}^\top \mathbf{F}\|_\infty \leq \|\mathbf{e}\| \cdot \|\mathbf{F}\|$.

Gaussian Distributions. For any positive parameter $\sigma \in \mathbb{R}_{>0}$, let $\rho_\sigma(\mathbf{x}) := \exp(-\pi\|\mathbf{x}\|^2/\sigma^2)$ be the Gaussian function on $\mathbb{R}^n$ of center $\mathbf{0}$ and parameter σ.

For any $n \in \mathbb{N}$ and any subset D of $\mathbb{Z}^n$, we define $\rho_\sigma(D) := \sum_{\mathbf{x}\in D} \rho_\sigma(\mathbf{x})$ the discrete integral of ρ_σ over D, and $\mathcal{D}_{D,\sigma}$ the discrete Gaussian distribution over D of parameter σ. That is, for all $\mathbf{y} \in D$, we have $\mathcal{D}_{D,\sigma}(\mathbf{y}) := \frac{\rho_\sigma(\mathbf{y})}{\rho_\sigma(D)}$.

Lemma 2 ([7], Lemma 2.5). *Let $n, m, \ell, q > 0$ be integers and $\sigma > 0$ be a Gaussian parameter. For all $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, $\mathbf{P} \in \mathbb{Z}_q^{n\times \ell}$, $\mathbf{U} \leftarrow_{\mathrm{R}} \mathcal{D}_{\Lambda_q^{\mathbf{P}}(\mathbf{A}),\sigma}$ and $\mathbf{R} \leftarrow_{\mathrm{R}} \{-1,1\}^{m\times m}$, with overwhelming probability in m,*

$$\|\mathbf{U}\| \le \sigma\sqrt{m}\ ,\ \|\mathbf{R}\|_2 \le 20\sqrt{m}$$

Trapdoor Generators. The following lemmas state properties of algorithms for generating short basis of lattices.

Lemma 3 ([5,6,18]). *Let $n, m, q > 0$ be integers with q prime and $m = \Theta(n \log q)$. There is a* PPT *algorithm* TrapGen *defined as follows:*

$\mathsf{TrapGen}(1^n, 1^m, q)$*:*
Inputs: *a security parameter n, an integer m such that $m = \Theta(n \log q)$, and a prime modulus q.*
Outputs: *a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and a basis $\mathbf{T_A} \in \mathbb{Z}_q^{m\times m}$ for $\Lambda_q^\perp(\mathbf{A})$ such that the distribution of $\mathbf{A}$ is $\mathrm{negl}(n)$-close to uniform and $\|\mathbf{T_A}\|_{\mathrm{GS}} = O(\sqrt{n \log q})$, with all but negligible probability in n.*

Lemma 4 ([18]). *Let $n, m, q > 0$ be integers with q prime and $m = \Theta(n \log q)$. There is a full-rank matrix $\mathbf{G}$ such that the lattice $\Lambda_q^\perp(\mathbf{G})$ has a publicly known basis $\mathbf{T_G} \in \mathbb{Z}^{m\times m}$ with $\|\mathbf{T_G}\|_{\mathrm{GS}} \le \sqrt{5}$.*

Lemma 5 ([13], Lemmas 5.1 and 5.2)

- *Let $m \ge 2n \log q$. With all but $\mathrm{negl}(n)$ probability, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m}$ is full rank (i.e. the subset-sums of the columns of $\mathbf{A}$ generate $\mathbb{Z}_q^n$).*
- *Assume $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is full-rank and $\sigma = \omega(\sqrt{\log n})$. Then, the following distributions are statistically close:*

$$\{(\mathbf{u}, \mathbf{Au}) : \mathbf{u} \leftarrow_{\mathrm{R}} \mathcal{D}_{\mathbb{Z}^m,\sigma}\} \quad \text{and} \quad \{(\mathbf{u}, \mathbf{p}) : \mathbf{p} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n, \mathbf{u} \leftarrow_{\mathrm{R}} \mathcal{D}_{\Lambda_q^{\mathbf{P}}(\mathbf{A}),\sigma}\}$$

2.2 Sampling Algorithms

Given a matrix $\mathbf{F} := [\mathbf{A}\|\mathbf{B}]$ and a matrix $\mathbf{P}$, we would like to sample random low-norm matrices $\mathbf{U}$ such that $\mathbf{FU} = \mathbf{P}$. Specifically, we want to sample $\mathbf{U}$ from the distribution $\mathcal{D}_{\Lambda_q^{\mathbf{P}}(\mathbf{F}),\sigma}$. The following lemma tells us we could do so given either (1) a low-norm basis $\mathbf{T_A}$ of $\Lambda_q^\perp(\mathbf{A})$ using RightSample, or (2) a low-norm matrix $\mathbf{R}$ and an invertible matrix $\mathbf{H}$ such that $\mathbf{B} = \mathbf{HG} + \mathbf{AR}$ using LeftSample. We will use (1) in the actual scheme, and (2) in the security proof.

Lemma 6 ([2,9,13,18] and [7], Lemma 2.8) *There exist* PPT *algorithms* RightSample *and* LeftSample *such that:*

RightSample$(\mathbf{A}, \mathbf{T_A}, \mathbf{B}, \mathbf{P}, \sigma)$*:*
Inputs: *full-rank matrices* $\mathbf{A}, \mathbf{B} \in \mathbb{Z}_q^{n\times m}$*, a basis* $\mathbf{T_A}$ *of* $\Lambda_q^{\perp}(\mathbf{A})$*, a matrix* $\mathbf{P} \in \mathbb{Z}_q^{n\times(\ell+n)}$ *and a Gaussian parameter* $\sigma = O(\|\mathbf{T_A}\|_{\mathrm{GS}})$.
Output: *a matrix* $\mathbf{U} \in \mathbb{Z}_q^{2m\times(\ell+n)}$ *whose distribution is statistically close to* $\mathcal{D}_{\Lambda_q^{\mathbf{P}}[\mathbf{A}\|\mathbf{B}],\sigma\cdot\omega(\sqrt{\log m})}$.

Remark 1. We can sample a short matrix $\mathbf{U} = \left(\frac{\mathbf{U}_1}{\mathbf{U}_2}\right) \in \mathbb{Z}_q^{2m\times(\ell+k)}$ in the same way that [2]. That is, we first sample the bottom part $\mathbf{U}_2 \in \mathbb{Z}_q^{m\times(\ell+k)}$ from $\mathcal{D}_{\mathbb{Z}_q^{m\times(\ell+k)},\sigma\cdot\omega(\sqrt{\log n})}$ and then, we sample the top part $\mathbf{U}_1 \in \mathbb{Z}_q^{m\times(\ell+k)}$ from a distribution statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{P}-\mathbf{BU}_2}(\mathbf{A}),\sigma\cdot\omega(\sqrt{\log m})}$, using $\mathbf{T_A}$.

LeftSample$(\mathbf{A}, \mathbf{R}, \mathbf{G}, \mathbf{H}, \mathbf{P}, \sigma)$ *:*
Inputs: *a full-rank matrix* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$*, a matrix* $\mathbf{R} \in \mathbb{Z}_q^{m\times m}$*, a full-rank matrix* $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ *as defined in Lemma 4, an invertible matrix* $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$*, a matrix* $\mathbf{P} \in \mathbb{Z}_q^{n\times(\ell+n)}$ *and a Gaussian parameter* $\sigma = O(\|\mathbf{R}\|_2)$.
Output: *a matrix* $\mathbf{U} \in \mathbb{Z}_q^{2m\times(\ell+n)}$ *whose distribution is statistically close to* $\mathcal{D}_{\Lambda_q^{\mathbf{P}}[\mathbf{A}\|\mathbf{HG}+\mathbf{AR}],\sigma\cdot\omega(\sqrt{\log m})}$.

2.3 Predicate Encryption

A predicate encryption scheme for a predicate $\mathsf{P}(\cdot,\cdot)$ consists of four algorithms (Setup, Enc, KeyGen, Dec):

$\mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M}) \to (\mathsf{mpk}, \mathsf{msk})$. The setup algorithm gets as input the security parameter n, the attribute universe $\mathcal{X}$, the predicate universe $\mathcal{Y}$, and the message space $\mathcal{M}$.

$\mathsf{Enc}(\mathsf{mpk}, x, m) \to \mathsf{ct}$. The encryption algorithm gets as input mpk, an attribute $x \in \mathcal{X}$ and a message $m \in \mathcal{M}$. It outputs a ciphertext ct.

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, y) \to \mathsf{sk}_y$. The key generation algorithm gets as input msk and a value $y \in \mathcal{Y}$. It outputs a secret key sk_y. Note that y is public given sk_y.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_y, \mathsf{ct}) \to m$. The decryption algorithm gets as input sk_y and a ciphertext ct. It outputs a message m.

Correctness. We require that for all $(x, y) \in \mathcal{X} \times \mathcal{Y}$ such that $\mathsf{P}(x, y) = 1$ and all $m \in \mathcal{M}$,

$$\Pr[\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x, m); \mathsf{Dec}(\mathsf{sk}_y, \mathsf{ct}) = m)] = 1 - \mathrm{negl}(n),$$

where the probability is taken over $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M})$ and the coins of Enc.

Security Model. For a stateful adversary $\mathcal{A}$, we define the advantage function

$$\mathsf{Adv}^{\mathrm{PE}}_{\mathcal{A}}(n) := \Pr\left[\beta = \beta' : \begin{array}{l} (x_0^*, x_1^*) \leftarrow \mathcal{A}(1^\lambda); \\ \beta \leftarrow_{\mathrm{R}} \{0,1\}; \\ (\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M}); \\ (m_0, m_1) \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{mpk}); \\ \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, x_\beta, m_\beta); \\ \beta' \leftarrow \mathcal{A}^{\mathsf{KeyGen}(\mathsf{msk},\cdot)}(\mathsf{ct}) \end{array}\right] - \frac{1}{2}$$

with the restriction that all queries y that $\mathcal{A}$ makes to $\mathsf{KeyGen}(\mathsf{msk}, \cdot)$ satisfies $\mathsf{P}(x_0^*, y) = \mathsf{P}(x_1^*, y) = 0$ (that is, sk_y does not decrypt ct). A predicate encryption scheme is *selectively secure and weakly attribute-hiding*[1] if for all PPT adversaries $\mathcal{A}$, the advantage $\mathsf{Adv}^{\mathrm{PE}}_{\mathcal{A}}(n)$ is a negligible function in n.

3 Reductions Amongst Predicates

3.1 AND-OR-EQ Predicate

In this section we state our general predicate, and exhibit the reductions from multi-dimensional subset queries and multi-dimensional range queries to the latter. This general predicate is symmetric (c.f. Lemma 11), which will allow us to obtain both ciphertext-policy and key-policy predicate encryption schemes in Section 4.

Disjunction of Equality Queries. Here, $\mathsf{P}_{\text{OR-EQ}} : \mathbb{Z}_q^\ell \times \mathbb{Z}_q^\ell \to \{0,1\}$, and $\mathsf{P}_{\text{OR-EQ}}(\mathbf{x}, \mathbf{y}) = 1$ iff $\bigvee_{i=1}^{\ell} (x_i = y_i)$.
We generalize the previous predicate to a multi-dimensional setting as follows $\mathsf{P}_{\text{AND-OR-EQ}} : \mathbb{Z}_q^{D\times\ell} \times \mathbb{Z}_q^{D\times\ell} \to \{0,1\}$, and:

$$\mathsf{P}_{\text{AND-OR-EQ}}(\mathbf{X}, \mathbf{Y}) = 1 \text{ iff } \bigwedge_{i=1}^{D} \bigvee_{j=1}^{\ell} (X_{i,j} = Y_{i,j})$$

We impose a so-called "at most one" promise on the input domains of the predicate for our predicate encryption scheme in Section 4. This technical property is required for our lattice-based instantiations (see Remark 3 in Section 4) and also implicitly used in prior pairing-based ones. We define

[1] In an adaptively secure scheme, the adversary specifies (x_0^*, x_1^*) after seeing mpk and making key queries. In a fully attribute-hiding scheme, the adversary is allowed key queries y for which $\mathsf{P}(x_0^*, y) = \mathsf{P}(x_1^*, y) = 1$, in which case the challenge messages m_0, m_1 must be equal.

the predicate $\mathsf{P}_{\text{AT MOST ONE}} : \mathbb{Z}_q^\ell \times \mathbb{Z}_q^\ell \to \{0,1\}$ and its multi-dimensional variant $\mathsf{P}_{\text{AND AT MOST ONE}} : \mathbb{Z}_q^{D\times\ell} \times \mathbb{Z}_q^{D\times\ell} \to \{0,1\}$ by:

$$\mathsf{P}_{\text{AT MOST ONE}}(\mathbf{x},\mathbf{y}) = 1 \text{ iff there exists at most one } j \in [\ell], x_j = y_j$$

$$\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X},\mathbf{Y}) = 1 \text{ iff } \forall i \in [D], \text{ there exists at most one } j \in [\ell] \text{ s.t. } X_{i,j} = Y_{i,j}$$

We require that the input domains $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{Z}_q^{D\times\ell}$ for $\mathsf{P}_{\text{AND-OR-EQ}}$ satisfy the "at most one" promise, namely for all $\mathbf{X} \in \mathcal{X}, \mathbf{Y} \in \mathcal{Y}$,

$$\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X},\mathbf{Y}) = 1$$

Indeed, the promise is satisfied by all of our reductions in this section.

3.2 Multi-dimensional Subset Queries

Predicate (ciphertext-policy). Here, $\mathsf{P}_{\text{CP-SUBSET}} : \{0,1\}^{D\times T} \times [T]^D \to \{0,1\}$ and

$$\mathsf{P}_{\text{CP-SUBSET}}(\mathbf{W},\mathbf{z}) = 1 \text{ iff } \bigwedge_{i=1}^{D} W_{i,z_i} = 1$$

For each dimension $i \in [D]$, the i'th row of $\mathbf{W}$ is the characteristic vector of a subset of $[T]$.

Reducing $\mathsf{P}_{\text{cp-subset}}$ to $\mathsf{P}_{\text{and-or-eq}}$. We map $(\mathbf{W},\mathbf{z}) \in \{0,1\}^{D\times T} \times [T]^D$ to $(\widetilde{\mathbf{W}},\widetilde{\mathbf{Z}}) \in \mathbb{Z}_q^{D\times T} \times \mathbb{Z}_q^{D\times T}$ where

- $\widetilde{\mathbf{W}}$ is the matrix $\mathbf{W}$ where zeros are replaced by -1, that is, for all $(i,j) \in [D] \times [T]$, $\widetilde{W}_{i,j} = 1$ if $W_{i,j} = 1$, and $\widetilde{W}_{i,j} = -1$ otherwise. (We need this modification in order to satisfy the "at most one" promise.)
- $\widetilde{\mathbf{Z}} \in \mathbb{Z}_q^{D\times T}$ denotes the matrix whose i'th row is $\mathbf{e}_{z_i} \in \mathbb{Z}_q^{1\times T}$, where $(\mathbf{e}_1,\ldots,\mathbf{e}_T)$ denotes the standard basis of $\mathbb{Z}^{1\times T}$.
  For instance, we map $((0,1,1),2)$ to $((-1,1,1),(0,1,0))$.

We can check that the following lemma holds:

Lemma 7 ($\mathsf{P}_{\text{cp-subset}}$ to $\mathsf{P}_{\text{and-or-eq}}$) *Let $(\mathbf{W},\mathbf{z}) \in \{0,1\}^{D\times T} \times [T]^D$, and $(\widetilde{\mathbf{W}},\widetilde{\mathbf{Z}}) \in \mathbb{Z}_q^{D\times T} \times \mathbb{Z}_q^{D\times T}$ defined as above.*

- $\mathsf{P}_{\text{CP-SUBSET}}(\mathbf{W},\mathbf{z}) = 1$ *iff* $\mathsf{P}_{\text{AND-OR-EQ}}(\widetilde{\mathbf{W}},\widetilde{\mathbf{Z}}) = 1$
- $\mathsf{P}_{\text{AND AT MOST ONE}}(\widetilde{\mathbf{W}},\widetilde{\mathbf{Z}}) = 1$

3.3 Multi-dimensional Range Queries

Predicate (key-policy). Here, $\mathsf{P}_{\text{KP-RANGE}} : [T]^D \times \mathcal{I}(T)^D \to \{0,1\}$ and $\mathsf{P}_{\text{KP-RANGE}}(\mathbf{z},\mathbf{I}) = 1$ iff $\bigwedge_{j=1}^{D} (z_j \in I_j)$, where $\mathcal{I}(T)$ denotes the set of all intervals of $[T]$.

We show how to reduce $\mathsf{P}_{\text{KP-RANGE}}$ to $\mathsf{P}_{\text{AND-OR-EQ}}$, which involves rewriting points and intervals as vectors.

Writing Points and Intervals as Vectors. For simplicity, we write t for $\lceil \log T \rceil$. In order to realize the "at most one" promise, we need to decompose intervals into disjoint sub-intervals where each sub-interval contains all the points with some fixed prefix, e.g. $[010, 110]$ can be written as $[010, 011] \cup [100, 101] \cup [110, 110]$. Indeed, any interval in $[T]$ can be partitioned into at most $2t$ disjoint sub-intervals with this property [10, Lemma 10.10].[2] In addition, we can ensure that there are exactly $2t$ sub-intervals by padding with empty intervals ε (using empty intervals ensures that no point ever lies in more than one of these $2t$ sub-intervals).

Lemma 8 (interval to vector [10]). *There is an efficient* PPT *algorithm* IntVec *that on input* $I \subseteq [T]$ *outputs* $(w_1, w_2, \ldots, w_{2t}) \in \left(\{0,1\}^* \cup \{\varepsilon\}\right)^{2t}$, *where* $t := \lceil \log T \rceil$, *with the following properties:*

- *for each* $i = 1, \ldots, t$, *we have* $w_{2i-1}, w_{2i} \in \{0,1\}^i \cup \{\varepsilon\}$;
- *for all* $z \in [T]$, *we have* $z \in I$ *iff one of* $w_1, \ldots, w_{2t}$ *is a prefix of* z;
- *for all* $z \in [T]$, *at most one of* $w_1, \ldots, w_{2t}$ *is a prefix of* z.

Here, ε *is not a prefix of any string.*

For instance, $\mathsf{IntVec}([010, 110]) = (\varepsilon, \varepsilon, 01, 10, 110, \varepsilon)$.

Remark 2 (Hashing bit strings into $\mathbb{Z}_q$*).* We map $\{0,1\}^t \cup \{\varepsilon\}$ where $t := \lceil \log T \rceil$ into $\mathbb{Z}_q$ in the straight-forward way, which requires that $q \geq T+1$. We can handle also larger T by using matrices over $\mathbb{Z}_q$ à la [1, Section 5].

Now we give the description of algorithm PtVec, used to map points to vectors.

PtVec: On input $z \in [T]$, output $(v_1, \ldots, v_{2t}) \in \mathbb{Z}_q^{2t}$, where $v_{2i-1} = v_{2i} :=$ i-bit prefix of z, $i = 1, \ldots, t$.
For instance, $\mathsf{PtVec}(011) = (0, 0, 01, 01, 011, 011)$.

Lemma 9. *For any point* $z \in [T]$ *and any interval* $I \subseteq [T]$,

- $z \in I$ *iff* $\mathsf{P}_{\text{OR-EQ}}(\mathsf{PtVec}(z), \mathsf{IntVec}(I)) = 1$
- $\mathsf{P}_{\text{AT MOST ONE}}(\mathsf{PtVec}(z), \mathsf{IntVec}(I)) = 1$.

Lemma 9 follows readily from Lemma 8.

Reducing $\mathsf{P}_{\text{kp-range}}$ to $\mathsf{P}_{\text{and-or-eq}}$. We map $(\mathbf{z}, \mathbf{I})$ to $(\mathsf{PtVec}^D(\mathbf{z}), \mathsf{IntVec}^D(\mathbf{I}))$ where

- $\mathsf{PtVec}^D(\mathbf{z}) \in \mathbb{Z}_q^{D \times 2t}$ denotes the matrix whose i'th row is $\mathsf{PtVec}(z_i)$
- $\mathsf{IntVec}^D(\mathbf{I}) \in \mathbb{Z}_q^{D \times 2t}$ denotes the matrix whose j'th row is $\mathsf{IntVec}(I_j)$

Lemma 10 ($\mathsf{P}_{\text{kp-range}}$ to $\mathsf{P}_{\text{and-or-eq}}$). *For all* $\mathbf{z} \in [T]^D$ *and* $\boldsymbol{I} \subseteq (\mathcal{I}[T])^D$,

- $\mathsf{P}_{\text{KP-RANGE}}(\mathbf{z}, \boldsymbol{I}) = 1$ *iff* $\mathsf{P}_{\text{AND-OR-EQ}}(\mathsf{PtVec}^D(\mathbf{z}), \mathsf{IntVec}^D(\boldsymbol{I})) = 1$
- $\mathsf{P}_{\text{AND AT MOST ONE}}(\mathsf{PtVec}^D(\mathbf{z}), \mathsf{IntVec}^D(\boldsymbol{I})) = 1$

Lemma 10 follows readily from Lemma 9, applied to each dimension $i \in [D]$.

[2] See http://en.wikipedia.org/wiki/Segment_tree for a visualization.

Predicate (ciphertext-policy). Here, $\mathsf{P}_{\text{CP-RANGE}} : \mathcal{I}(T)^D \times [T]^D \to \{0,1\}$ and

$$\mathsf{P}_{\text{CP-RANGE}}(\mathbf{I}, \mathbf{z}) = 1 \text{ iff } \bigwedge_{j=1}^{D} (z_j \in I_j)$$

The predicates $\mathsf{P}_{\text{OR-EQ}}$ and $\mathsf{P}_{\text{AT MOST ONE}}$ are symmetric:

Lemma 11 (Symmetry of $\mathsf{P}_{\text{or-eq}}$ and $\mathsf{P}_{\text{at most one}}$). *For all $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_q^\ell$, we have:*

- $\mathsf{P}_{\text{OR-EQ}}(\mathbf{x}, \mathbf{y}) = 1 \iff \mathsf{P}_{\text{OR-EQ}}(\mathbf{y}, \mathbf{x}) = 1$
- $\mathsf{P}_{\text{AT MOST ONE}}(\mathbf{x}, \mathbf{y}) = 1 \iff \mathsf{P}_{\text{AT MOST ONE}}(\mathbf{y}, \mathbf{x}) = 1$

Thanks to this symmetry, we can reduce $\mathsf{P}_{\text{CP-RANGE}}$ to $\mathsf{P}_{\text{AND-OR-EQ}}$ in the same way we did for $\mathsf{P}_{\text{KP-RANGE}}$.

Reducing $\mathsf{P}_{\text{cp-range}}$ to $\mathsf{P}_{\text{and-or-eq}}$. Following the previous reduction, we map $(\mathbf{I}, \mathbf{z})$ to $(\mathsf{IntVec}^D(\mathbf{I}), \mathsf{PtVec}^D(\mathbf{z}))$.

Lemma 12 ($\mathsf{P}_{\text{cp-range}}$ to $\mathsf{P}_{\text{and-or-eq}}$). *For all $\boldsymbol{I} \subseteq (\mathcal{I}[T])^D$ and $\mathbf{z} \in [T]^D$,*

- $\mathsf{P}_{\text{CP-RANGE}}(\boldsymbol{I}, \mathbf{z}) = 1$ *iff* $\mathsf{P}_{\text{AND-OR-EQ}}(\mathsf{IntVec}^D(\boldsymbol{I}), \mathsf{PtVec}^D(\mathbf{z})) = 1$
- $\mathsf{P}_{\text{AND AT MOST ONE}}(\mathsf{IntVec}^D(\boldsymbol{I}), \mathsf{PtVec}^D(\mathbf{z})) = 1$.

Lemma 12 follows from Lemma 10 and Lemma 11.

4 Predicate Encryption for AND-OR-EQ

Here we describe our predicate encryption scheme for the AND-OR-EQ predicate defined in Section 3.1, selectively secure and lattice-based. Recall that $\mathsf{P}_{\text{AND-OR-EQ}} : \mathbb{Z}_q^{D \times \ell} \times \mathbb{Z}_q^{D \times \ell} \to \{0,1\}$, and

$$\mathsf{P}_{\text{AND-OR-EQ}}(\mathbf{X}, \mathbf{Y}) = 1 \text{ iff } \bigwedge_{i=1}^{D} \bigvee_{j=1}^{\ell} (X_{i,j} = Y_{i,j})$$

The security of our scheme relies on the fact the ciphertext attributes and secret key predicates come from a restricted domain $\mathcal{X}, \mathcal{Y} \subseteq \mathbb{Z}_q^{D \times \ell}$ satisfying the "at most one" promise, namely for all $\mathbf{X} \in \mathcal{X}, \mathbf{Y} \in \mathcal{Y}$,

$$\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X}, \mathbf{Y}) = 1$$

Indeed, the promise is satisfied by all of our reductions in Section 3.1.

Overview. We begin with the special case $D = 1$. Given an attribute matrix $\mathbf{x} \in \mathbb{Z}_q^{1 \times \ell}$, the ciphertext is an LWE sample corresponding to a matrix of the form

$$\left[\mathbf{A} \| \mathbf{A}_1 + x_1 \mathbf{G} \| \cdots \| \mathbf{A}_\ell + x_\ell \mathbf{G}\right]$$

where $\mathbf{A}$, the $\mathbf{A}_i$'s and $\mathbf{G}$ are publicly known matrices, and the message is masked using an LWE sample corresponding to a public matrix $\mathbf{P}$. The secret key corresponding to $\mathbf{y} \in \mathbb{Z}_q^{1 \times \ell}$ is a collection of ℓ short matrices $\mathbf{U}_1, \ldots, \mathbf{U}_\ell$ such that for all $j \in [\ell]$,

$$\left[\mathbf{A} \| \mathbf{A}_j + y_j \mathbf{G}\right] \mathbf{U}_j = \mathbf{P}$$

To understand how decryption works, let us first suppose that there exists a unique j such that $x_j = y_j$ and that the decryptor knows j; then, the decryptor can use $\mathbf{U}_j$ to recover the plaintext. However, since the decryptor does not know $\mathbf{x}$, he will try to decrypt the ciphertext using each of $\mathbf{U}_1, \ldots, \mathbf{U}_\ell$. We will also need to pad the plaintext with redundant zeros so that the decryptor can identify the correct plaintext.

To establish security with respect to some selective challenge $\mathbf{x}^*$, we will need to simulate secret keys for all $\mathbf{y}$ such that $x_j^* \neq y_j$ for all $j \in [\ell]$. We can then simulate $\mathbf{U}_j$ using an "all-but-one" simulation (while puncturing at x_j^*) exactly as in prior IBE schemes in [1]. In order to establish weak attribute-hiding, we adopt a similar strategy to that for inner product encryption in [4]: roughly speaking, we will show that the challenge ciphertext is computational indistinguishable from an encryption of a random plaintext message under a random attribute $\mathbf{x}$.

Higher Dimensions. Given an attribute matrix $\mathbf{X} \in \mathbb{Z}_q^{D \times \ell}$, the ciphertext is an LWE sample corresponding to a matrix of the form

$$\left[\mathbf{A} \| \mathbf{A}_{1,1} + X_{1,1} \mathbf{G} \| \cdots \| \mathbf{A}_{1,\ell} + X_{1,\ell} \mathbf{G} \| \mathbf{A}_{2,1} + X_{2,1} \mathbf{G} \| \cdots \| \mathbf{A}_{D,\ell} + X_{D,\ell} \mathbf{G}\right]$$

The secret key corresponding to $\mathbf{Y} \in \mathbb{Z}_q^{D \times \ell}$ is a collection of $D \cdot \ell$ short matrices $\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,\ell}$ such that for $i \in [D]$, $j \in [\ell]$:

$$\left[\mathbf{A} \| \mathbf{A}_{i,j} + Y_{i,j} \mathbf{G}\right] \mathbf{U}_{i,j} = \mathbf{P}_i$$

where $\mathbf{P}_1, \ldots, \mathbf{P}_D$ is an additive secret-sharing of $\mathbf{P}$.

For correctness, observe that if there exist indices $(j_1, \ldots, j_D) \in [\ell]^D$ such that for all $i \in [D]$ $X_{i,j_i} = Y_{i,j_i}$ then the decryptor can use $\mathbf{U}_{1,j_1}, \cdots, \mathbf{U}_{D,j_D}$ to recover the plaintext. As with the case $D = 1$, the decryptor will need to enumerate over all $(j_1, \ldots, j_D) \in [\ell]^D$.

To simulate secret keys for $\mathbf{Y}$ with respect to some selective challenge $\mathbf{X}^*$, we first fix i^* such that $X_{i^*,j}^* \neq Y_{i^*,j}$ for all $j \in [\ell]$. Without loss of generality, suppose $i^* = 1$, that is, for all $j \in [\ell]$, $X_{1,j}^* \neq Y_{1,j}$.

Using the "at most one" promise on $\mathbf{X}$ and $\mathbf{Y}$, we know that for all $i \geq 2$, we have $X_{i,j}^* = Y_{i,j}$ for at most one $j \in [\ell]$, which we call j_i. That is, there exists a vector of indices $(j_2, \ldots, j_D) \in [\ell]^{D-1}$ such that for all $i \geq 2$ and all $j \neq j_i$, we have $X_{i,j}^* \neq Y_{i,j}$. We then proceed as follows:

- Sample random short matrices $\mathbf{U}_{2,j_2}, \ldots, \mathbf{U}_{D,j_D}$, which in turn determines the shares $\mathbf{P}_2, \ldots, \mathbf{P}_D$.

– Use an all-but-one simulation strategy to sample the short matrices $\mathbf{U}_{i,j}$, for all $i \geq 2$, and $j \neq j_i$.
– Define $\mathbf{P}_1 := \mathbf{P} - \sum_{i=2}^{D} \mathbf{P}_i$ and use an all-but-one simulation strategy to sample the remaining short matrices in the secret key.

Note that the construction relies crucially on the "at most one" promise on $\mathbf{X}$ and $\mathbf{Y}$. In fact, the scheme given in Section 4.1 is insecure if this promise is not fulfilled, that is, there is a PPT adversary that wins the game defined in Section 2.3 with non negligible advantage. The attack goes as follows. Suppose that the adversary holds a secret key for $\mathbf{Y} \in \mathbb{Z}_q^{D \times \ell}$ such that $\mathsf{P}_{\text{AND-OR-EQ}}(\mathbf{X}, \mathbf{Y}) = 0$ and $\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X}, \mathbf{Y}) = 0$. Since $\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X}, \mathbf{Y}) = 0$, for some dimension $i \in [D]$, there are at least two indices $j_1, j_2 \in [\ell]$ such that $X_{i,j_1} = Y_{i,j_1}$ and $X_{i,j_2} = Y_{i,j_2}$, and therefore, both short matrices $\mathbf{U}_{i,j_1}$ and $\mathbf{U}_{i,j_2}$ allow to recover the same LWE sample corresponding to the matrix $\mathbf{P}_i$, up to some error. When $X_{i,j_2} \neq Y_{i,j_2}$ however, $\mathbf{U}_{i,j_2}$ with the corresponding (i, j_2) component of the ciphertext only gives a uniformly random vector. Therefore, the fact that $\mathbf{U}_{i,j_1}$ and $\mathbf{U}_{i,j_2}$ both decrypts to (almost) the same LWE sample tells the adversary that $X_{i,j_1} = Y_{i,j_1}$ and $X_{i,j_2} = Y_{i,j_2}$ with high probability, contradicting the fact that the scheme is weakly attribute hiding. This induces an attack whose running time is polynomial in the security parameter.

4.1 Construction

Let $n \in \mathbb{N}$ be the security parameter. Let the attribute space $\mathcal{X}$ and predicate space $\mathcal{Y}$ be subsets of $\mathbb{Z}_q^{D \times \ell}$ satisfying the "at most one" promise. Let $q = q(n)$, $m = m(n, \ell, D)$ and $\chi_{\max} = \chi_{\max}(n, q, \ell, D)$ be positive integers. Let $\sigma = \sigma(n, q, \ell, D)$ be a Gaussian parameter.

$\mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M})$: On inputs the security parameter n, $\mathcal{X} \subseteq \mathbb{Z}_q^{D \times \ell}$, $\mathcal{Y} \subseteq \mathbb{Z}_q^{D \times \ell}$ and $\mathcal{M} := \{0,1\}^k$, do:

– Pick $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
– For all $i \in [D]$ and all $j \in [\ell]$, pick $\mathbf{A}_{i,j} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$.
– Pick $\mathbf{P} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times (k+n)}$.
– Compute $(\mathbf{G}, \mathbf{T}_\mathbf{G})$ as defined in Lemma 4.
– Output $\mathsf{mpk} := (\mathbf{P}, \mathbf{A}, \mathbf{A}_{1,1}, \ldots, \mathbf{A}_{D,\ell}, \mathbf{G}, \mathbf{T}_\mathbf{G})$ and $\mathsf{msk} := \mathbf{T}_\mathbf{A}$

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{X}, \mathbf{b})$: On input mpk, $\mathbf{X} \in \mathcal{X}$ and $\mathbf{b} \in \{0,1\}^k$, do:

– Pick $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$, and compute $\mathbf{c}_0 := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top$.
– For all $i \in [D]$ and all $j \in [\ell]$, do:
  • Pick $\mathbf{R}_{i,j} \leftarrow \{-1, 1\}^{m \times m}$.
  • Compute $\mathbf{c}_{i,j} := \mathbf{s}^\top \big(\mathbf{A}_{i,j} + X_{i,j} \mathbf{G} \big) + \mathbf{e}^\top \mathbf{R}_{i,j}$.
– Set $\mathbf{b}' := (\mathbf{b}, 0, \ldots, 0) \in \{0,1\}^{k+n}$, pick $\mathbf{e}' \leftarrow_{\mathrm{R}} \chi^{k+n}$, and compute $\mathbf{c}_f := \mathbf{s}^\top \mathbf{P} + \mathbf{e}'^\top + \mathbf{b}'^\top \cdot \lfloor q/2 \rfloor$.
– Output $\mathsf{ct} := (\mathbf{c}_0, \mathbf{c}_{1,1}, \ldots, \mathbf{c}_{D,\ell}, \mathbf{c}_f)$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{Y})$: On input the public parameters mpk, the master secret key msk, and a predicate matrix $\mathbf{Y} \in \mathcal{Y}$, do:

- Secret share $\mathbf{P}$ as $\{\mathbf{P}_i, i \in [D]\}$, such that $\sum_{i=1}^{D} \mathbf{P}_i = \mathbf{P}$.
- For all $i \in [D]$ and all $j \in [\ell]$:
  Sample a short matrix $\mathbf{U}_{i,j} \in \mathbb{Z}_q^{2m \times (k+n)}$ such that $\big[\mathbf{A} \| \mathbf{A}_{i,j} + Y_{i,j}\mathbf{G}\big]\mathbf{U}_{i,j} = \mathbf{P}_i$ using
  $\mathsf{RightSample}(\mathbf{A}, \mathbf{T}_{\mathbf{A}}, \mathbf{A}_{i,j} + Y_{i,j}\mathbf{G}, \mathbf{P}_i, \sigma)$ with $\sigma = O(\sqrt{n \log q})$.
- output the secret key $\mathsf{sk}_{\mathbf{Y}} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,\ell})$

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathbf{Y}}, \mathsf{ct})$: On input the public parameters mpk, a secret key $\mathsf{sk}_{\mathbf{Y}} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,\ell})$ for a predicate matrix $\mathbf{Y}$, and a ciphertext $\mathsf{ct} = (\mathbf{c}_0, \mathbf{c}_{1,1}, \ldots, \mathbf{c}_{D,\ell}, \mathbf{c}_f)$, do:

- For all $(j_1, \ldots, j_D) \in [\ell]^D$, compute $\mathbf{d} := \mathbf{c}_f - \sum_{i=1}^{D} \big[\mathbf{c}_0 \| \mathbf{c}_{i,j_i}\big]\mathbf{U}_{i,j_i} \mod q$.
- If $\left\lfloor \frac{\mathbf{d}}{q/2} \right\rceil \in \{0,1\}^k \times 0^n$, then output the first k bits of $\left\lfloor \frac{\mathbf{d}}{q/2} \right\rceil$. For any $z \in [0,1]$, we denote by $\lfloor z \rceil$ the closest integer to z.
- Otherwise, abort.

Running Time. The running times are:

- $O(D \cdot \ell \cdot \mathrm{poly}(n))$ for encryption;
- $O(D \cdot \ell \cdot \mathrm{poly}(n))$ for key generation;
- $O(\ell^D \cdot \mathrm{poly}(n))$ for decryption.

The above numbers take into account matrix multiplications and additions. When done naively, the above Dec algorithm takes $O(D \cdot \ell^D \cdot \mathrm{poly}(n))$ time. However, if one saves the intermediate results $\big[\mathbf{c}_0 \| \mathbf{c}_{i,j}\big]\mathbf{U}_{i,j}$ for all $(i,j) \in [D] \times [\ell]$, one can do it in $O(\ell^D \cdot \mathrm{poly}(n))$ time.

4.2 Correctness

Lemma 13. *Suppose that* $\chi_{\max}$ *is such that*

$$\chi_{\max} \leq q/4 \cdot \big(1 + D \cdot (1 + 20\sqrt{m}) \cdot s \cdot 2m\big)^{-1}$$

where $s = \sigma \cdot \omega(\sqrt{\log n})$. *Let* $(\mathbf{X}, \mathbf{Y}) \in \mathcal{X} \times \mathcal{Y}$ *such that* $\mathsf{P}_{\text{AND-OR-EQ}}(\mathbf{X}, \mathbf{Y}) = 1$.
Let $\mathsf{sk}_{\mathbf{Y}} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,\ell}) \leftarrow \mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{Y})$
and $\mathsf{ct} = (\mathbf{c}_0, \ldots, \mathbf{c}_f) \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mathbf{X}, \mathbf{b})$
With overwhelming probability in n, Dec *does not abort and outputs* $\mathbf{b}$.

Proof. Recall that Dec computes $\mathbf{d}$ for all $(j_1, \ldots, j_D) \in \big[\ell\big]^D$. We consider two cases:

Case 1: $\forall i$, $X_{i,j_i} = Y_{i,j_i}$. We show that with overwhelming probability in n, $\left\lfloor \frac{\mathbf{d}}{q/2} \right\rceil = (\mathbf{b}, 0, \ldots, 0) := \mathbf{b}'$.
For all $i \in [D]$,

$$\big[\mathbf{A} || \mathbf{A}_{i,j_i} + X_{i,j_i}\mathbf{G}\big]\mathbf{U}_{i,j_i} = \mathbf{P}_i \quad \text{thus} \quad \sum_{i=1}^{D} \big[\mathbf{A} || \mathbf{A}_{i,j_i} + X_{i,j_i}\mathbf{G}\big]\mathbf{U}_{i,j_i} = \mathbf{P}$$

This implies

$$\sum_{i=1}^{D}[\mathbf{c}_0\|\mathbf{c}_{i,j_i}]\mathbf{U}_{i,j_i} \approx \sum_{i=1}^{D}\mathbf{s}^\top\left[\mathbf{A}||\mathbf{A}_{i,j_i}+X_{i,j_i}\mathbf{G}\right]\mathbf{U}_{i,j_i} = \mathbf{s}^\top\mathbf{P} \approx \mathbf{c}_f - \mathbf{b}'^\top\cdot\lfloor q/2\rfloor$$

and thus $\mathbf{d} \approx \mathbf{b}'\cdot\lfloor q/2\rfloor$. To obtain $\left\lfloor\frac{\mathbf{d}}{q/2}\right\rceil = \mathbf{b}'$, it suffices to bound the error term and show that

$$\left\|\mathbf{e}'^\top - \sum_{i=1}^{D}(\mathbf{e}^\top||\mathbf{e}^\top\mathbf{R}_{i,j_i})\mathbf{U}_{i,j_i}\right\|_\infty < q/4.$$

We know that $\|\mathbf{e}'^\top\|_\infty \le \chi_{\max}$. In addition, for all $i \in [D]$,

$$\left\|(\mathbf{e}^\top||\mathbf{e}^\top\mathbf{R}_{i,j_i})\mathbf{U}_{i,j_i}\right\|_\infty \le \left\|(\mathbf{e}^\top||\mathbf{e}^\top\mathbf{R}_{i,j_i})\right\|\cdot\left\|\mathbf{U}_{i,j_i}\right\| \le \left(\|\mathbf{e}\|+\|\mathbf{R}_{i,j_i}\|_2\cdot\|\mathbf{e}\|\right)\cdot\left\|\mathbf{U}_{i,j_i}\right\|$$

By Lemma 6, $\|\mathbf{U}_{i,j_i}\| \le \sigma\cdot\omega(\sqrt{\log n})\cdot\sqrt{2m}$ and by Lemma 2, $\|\mathbf{R}_{i,j_i}\|_2 \le 20\sqrt{m}$. Combining these bounds, we obtain

$$\left\|\mathbf{e}'^\top - \sum_{i=1}^{D}(\mathbf{e}^\top||\mathbf{e}^\top\mathbf{R}_{i,j_i})\mathbf{U}_{i,j_i}\right\|_\infty \le \chi_{\max}(1+D(1+20\sqrt{m})\cdot\sigma\cdot\omega(\sqrt{\log m})\sqrt{2m})$$

We will set the parameters in Section 4.4 so that the quantity on the right is bounded by $q/4$.

Case 2: $\exists i^*, X_{i^*,j_{i^*}} \neq Y_{i^*,j_{i^*}}$**.** We show that the computed value $\mathbf{d}$ has a distribution which is statistically close to uniform, and therefore the probability that the last n bits of $\left\lfloor\frac{\mathbf{d}}{q/2}\right\rceil$ are all 0 is negligible in n.
By an analogous calculation to that in Case 1, we have:

$$\sum_{i=1}^{D}\left[\mathbf{A}||\mathbf{A}_{i,j_i}+X_{i,j_i}\mathbf{G}\right]\mathbf{U}_{i,j_i} = \mathbf{P} + \sum_{i=1}^{D}[\mathbf{0}||(X_{i,j_i}-Y_{i,j_i})\mathbf{G}]\mathbf{U}_{i,j_i}$$

We know that there exists some $i^* \in [D]$ such that $X_{i^*,j_{i^*}} \neq Y_{i^*,j_{i^*}}$, and we focus on

$$\left[\mathbf{0}^\top||\mathbf{s}^\top(X_{i^*,j_{i^*}}-Y_{i^*,j_{i^*}})\mathbf{G}\right]\mathbf{U}_{i^*,j_{i^*}}$$

By Remark 1, we know that the bottom part $\mathbf{U}_2 \in \mathbb{Z}_q^{m\times(k+n)}$ of $\mathbf{U}_{i^*,j_{i^*}}$ is sampled from $\mathcal{D}_{\mathbb{Z}_q^m,\sigma\cdot\omega(\sqrt{\log n})}$. Therefore, since $X_{i^*,j_{i^*}} - Y_{i^*,j_{i^*}} \neq 0$ and $\mathbf{G}$ is a full-rank matrix, by Lemma 5, we know that the distribution of $\mathbf{G}\mathbf{U}_2$ is indistinguishable from uniform over $Z_q^{n\times(k+n)}$ and then, the entire sum is indistinguishable from uniform over $\mathbb{Z}_q^{k+n}$. Therefore,

$$\mathbf{d} = \mathbf{c}_f - \sum_{i=1}^{D}[\mathbf{c}_0||\mathbf{c}_{i,j_i}]\mathbf{U}_{i,j_i} \mod q$$

is indistinguishable from uniform over $\mathbb{Z}_q^{k+n}$, and the probability that the last n bits of $\left\lfloor\frac{\mathbf{d}}{q/2}\right\rceil$ are all 0 is negligible in n.

4.3 Proof of Security

Lemma 14. *For any adversary $\mathcal{A}$ on the predicate encryption scheme, there exists an adversary $\mathcal{B}$ on the LWE assumption whose running time is roughly the same as that of $\mathcal{A}$ and such that*

$$\mathsf{Adv}^{\mathrm{PE}}_{\mathcal{A}}(n) \leq \mathsf{Adv}^{\mathsf{dLWE}_{n,m+n+k,q,\chi}}_{\mathcal{B}} + \mathrm{negl}(n)$$

The proof follows via a series of games, analogous to those in [4]. We first define several auxiliary algorithms for generating the simulated ciphertexts and secret keys, upon which we can describe the games.

Auxiliary Algorithms. We introduce the following auxiliary algorithms:

$\widetilde{\mathsf{Setup}}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M}, \mathbf{A}, \mathbf{P}, \mathbf{X}^*)$: on a security parameter n, an attribute space $\mathcal{X} \subseteq \mathbb{Z}_q^{D\times\ell}$, a predicate space $\mathcal{Y} \subseteq \mathbb{Z}_q^{D\times\ell}$ satisfying the "at most one promise", a message space $\mathcal{M} := \{0,1\}^k$, a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$, a matrix $\mathbf{P} \in \mathbb{Z}_q^{n\times(k+n)}$ and the challenge attribute matrix $\mathbf{X}^*$, do:

- Compute $(\mathbf{G}, \mathbf{T_G})$ as defined in Lemma 4.
- For all $i \in [D]$ and all $j \in [\ell]$, pick $\mathbf{R}_{i,j} \leftarrow \{-1,1\}^{m\times m}$ and set $\mathbf{A}_{i,j} := \mathbf{A}\mathbf{R}_{i,j} - X^*_{i,j}\mathbf{G}$.
- Output $\mathsf{mpk} := (\mathbf{P}, \mathbf{A}, \mathbf{A}_{1,1}, \ldots, \mathbf{A}_{D,\ell}, \mathbf{G}, \mathbf{T_G})$ and $\widetilde{\mathsf{msk}} := (\mathbf{X}^*, \mathbf{R}_{1,1}, \cdots, \mathbf{R}_{D,\ell}, \mathbf{T_A})$

$\widetilde{\mathsf{Enc}}(\mathsf{mpk}, \mathbf{b}; \widetilde{\mathsf{msk}}, \mathbf{d}_0, \mathbf{d}_f)$: On input the public parameters mpk, a message $\mathbf{b} \in \{0,1\}^k$, the master secret key $\widetilde{\mathsf{msk}}$, and the extra inputs $\mathbf{d}_0 \in \mathbb{Z}_q^m$, $\mathbf{d}_f \in \mathbb{Z}_q^{k+n}$ do:

- Set $\mathbf{c}_0 := \mathbf{d}_0^\top$,
- For all $i \in [D]$ and all $j \in [\ell]$, compute $\mathbf{c}_{i,j} := \mathbf{d}_0^\top \mathbf{R}_{i,j}$,
- Compute $\mathbf{b}' := (\mathbf{b}, 0, \ldots, 0) \in \{0,1\}^{k+n}$, and set $\mathbf{c}_f := \mathbf{d}_f^\top + \mathbf{b}'^\top \lfloor q/2 \rfloor$.
- Output $(\mathbf{c}_0, \ldots, \mathbf{c}_f)$.

$\widetilde{\mathsf{KeyGen}}(\mathsf{mpk}, \widetilde{\mathsf{msk}}, \mathbf{Y}, \mathbf{X}^*)$: On input the public parameters mpk, the master secret key $\widetilde{\mathsf{msk}}$, a predicate matrix $\mathbf{Y}$ and the challenge attribute matrix $\mathbf{X}^*$ do:

- If $\mathsf{P}(\mathbf{X}^*, \mathbf{Y}) = 1$, abort.
- Otherwise, since $\mathsf{P}(\mathbf{X}^*, \mathbf{Y}) = 0$, there must exist a $i^* \in [D]$ such that for all $j \in [\ell]$, $X^*_{i^*,j} \neq Y_{i^*,j}$. By the "at most one" property, for all $i \in [D]$, there is at most one $j_i \in [\ell]$ such that $X^*_{i,j_i} = Y_{i,j_i}$.
  We proceed in three steps :
  1. First, for all $i \in [D] \setminus \{i^*\}$ we sample: $\mathbf{U}_{i,j_i} \leftarrow_{\mathrm{R}} \mathcal{D}_{\mathbb{Z}_q^{2m\times(k+n)}, \sigma\cdot\omega(\sqrt{\log n})}$ with $\sigma = O(\sqrt{n \log q})$ and set

$$\mathbf{P}_i := \left[\mathbf{A} || \mathbf{A}\mathbf{R}_{i,j_i}\right] \mathbf{U}_{i,j_i} \in \mathbb{Z}_q^{n\times(k+n)}$$

2. Next, we set $\mathbf{P}_{i^*}$ to be: $\mathbf{P}_{i^*} := \mathbf{P} - \sum_{i \neq i^*} \mathbf{P}_i$
3. We sample the remaining matrices as follows:

$$\mathbf{U}_{i,j} \leftarrow \mathsf{LeftSample}(\mathbf{A}, \mathbf{R}_{i,j}, \mathbf{G}, Y_{i,j} - X^*_{i,j}, \mathbf{P}_i, \sigma).$$

This is possible because $Y_{i,j} - X^*_{i,j} \neq 0$, whenever $i = i^*$ or whenever $j \neq j_i$.

– Output $(\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,\ell})$.

Game Sequence. We present a series of games. We write $\mathsf{Adv}_{\mathrm{xxx}}$ to denote the advantage of $\mathcal{A}$ in $\mathsf{Game}_{\mathrm{xxx}}$.

– Game_0: is the real security game, as defined in Section 2.3.
– Game_1: same as Game_0, except that the challenger runs

$$\widetilde{\mathsf{Setup}}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M}; \mathbf{A}, \mathbf{P}, \mathbf{X}^*_\beta) \quad \text{and} \quad \widetilde{\mathsf{Enc}}(\mathsf{mpk}, \mathbf{b}_\beta; \widetilde{\mathsf{msk}}, \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top, \mathbf{s}^\top \mathbf{P} + \mathbf{e}'^\top)$$

with $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow_{\mathrm{R}} \mathsf{TrapGen}(1^n, 1^m), \mathbf{P} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times (k+n)}$, $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$, and $\mathbf{e}' \leftarrow_{\mathrm{R}} \chi^{k+n}$.
– Game_2: same as Game_1 except that the challenger runs $\widetilde{\mathsf{KeyGen}}$.
– Game_3: same as Game_2 except that the challenger runs

$$\widetilde{\mathsf{Enc}}(\mathsf{mpk}, \mathbf{b}_\beta; \widetilde{\mathsf{msk}}, \mathbf{d}_0, \mathbf{d}_f)$$

where $\mathbf{d}_0 \leftarrow_{\mathrm{R}} \mathbb{Z}_q^m$ and $\mathbf{d}_f \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{k+n}$.
– Game_4: is the same as Game_3 except that the challenger runs KeyGen.

We show in the following lemmas that each pair of games $(\mathsf{Game}_i, \mathsf{Game}_{i+1})$ are either statistically indistinguishable or computationally indistinguishable under the decision-LWE assumption. Finally, we show in Lemma 19 that no information is leaked about β is the last game Game_4.

Lemma 15 (Game_0 to Game_1). *For all $m > (n+1)\log q + \omega(\log n)$, we have $|\mathsf{Adv}_0 - \mathsf{Adv}_1| = \mathrm{negl}(n)$.*

Proof. From Game_0 to Game_1, we switch from $\mathsf{Setup}, \mathsf{Enc}$ to $\widetilde{\mathsf{Setup}}, \widetilde{\mathsf{Enc}}$.

Note that the only difference between Game_0 and Game_1 is that for all $i \in [D]$ and $j \in [\ell]$, the matrix $\mathbf{A}_{i,j}$ is set to be $\mathbf{A}_{i,j} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ in Game_0, whereas it is set to be $\mathbf{A}_{i,j} := \mathbf{A}\mathbf{R}_{i,j} - X^*_{i,j}\mathbf{G}$ in Game_1, where $\mathbf{R}_{i,j} \leftarrow_{\mathrm{R}} \{-1, 1\}^{m \times m}$. The matrix $\mathbf{A}_{i,j}$ only appears in the mpk and in the component $\mathbf{c}_{i,j} = \mathbf{s}^\top(\mathbf{A}_{i,j} + X^*_{i,j}\mathbf{G}) + \mathbf{e}^\top\mathbf{R}_{i,j}$ of the ciphertext.

Therefore it suffices to show that the distribution of $(\mathbf{A}, \mathbf{e}, \mathbf{A}_{i,j}, \mathbf{e}^\top\mathbf{R}_{i,j})$ in Game_0 and Game_1 are statistically close, that is,

$$(\mathbf{A}, \mathbf{e}, \mathbf{A}_{i,j}, \mathbf{e}^\top\mathbf{R}_{i,j}) \approx_s (\mathbf{A}, \mathbf{e}, \mathbf{A}\mathbf{R}_{i,j} - X^*_{i,j}\mathbf{G}, \mathbf{e}^\top\mathbf{R}_{i,j})$$

where $\mathbf{A}_{i,j} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m}$, $\mathbf{R}_{i,j} \leftarrow_{\mathrm{R}} \{-1,1\}^{m\times m}$, and $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$. Observe that

$$\begin{aligned}
&(\mathbf{A}, \mathbf{e}, \mathbf{A}_{i,j}, \mathbf{e}^\top \mathbf{R}_{i,j}) \\
&\equiv (\mathbf{A}, \mathbf{e}, \mathbf{A}_{i,j} - X^*_{i,j}\mathbf{G}, \mathbf{e}^\top \mathbf{R}_{i,j}) && \text{since } \mathbf{A}_{i,j} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n\times m} \\
&\approx_s (\mathbf{A}, \mathbf{e}, \mathbf{A}\mathbf{R}_{i,j} - X^*_{i,j}\mathbf{G}, \mathbf{e}^\top \mathbf{R}_{i,j}) && \text{by Lemma 1}
\end{aligned}$$

Lemma 16 (Game_1 to Game_2). $|\mathsf{Adv}_1 - \mathsf{Adv}_2| = \mathrm{negl}(n)$.

Proof. From Game_1 to Game_2, we switch from KeyGen to $\widetilde{\mathsf{KeyGen}}$. Therefore, it suffices to show that for any predicate $\mathbf{Y}$ such that $\mathsf{P}(\mathbf{X}^*, \mathbf{Y}) = 0$, the following distributions are statistically close:

$$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{Y}) \approx_s \widetilde{\mathsf{KeyGen}}(\mathsf{mpk}, \widetilde{\mathsf{msk}}, \mathbf{Y}, \mathbf{X}^*)$$

We write: $\mathbf{F}_{i,j} := (\mathbf{A} \| \mathbf{A}_{i,j} + Y_{i,j}\mathbf{G}) = \left(\mathbf{A} \| \mathbf{A}\mathbf{R}_{i,j} + (Y_{i,j} - X^*_{i,j})\mathbf{G}\right) \in \mathbb{Z}_q^{n\times 2m}$. Since $\mathsf{P}(\mathbf{X}^*, \mathbf{Y}) = 0$, we know that there must exist a $i^* \in [D]$ such that $Y_{i^*,j} \neq X^*_{i^*,j}$ for all $j \in [\ell]$. Because $\mathsf{P}_{\text{AND AT MOST ONE}}(\mathbf{X}^*, \mathbf{Y}) = 1$, we know that for all $i \in [D]$, there is at most one $j_i \in [\ell]$ such that $Y_{i,i_j} = X^*_{i,i_j}$. We proceed in three steps:

1. First, we argue that the joint distribution $\{\mathbf{P}_i, \mathbf{U}_{i,j_i} : i \in [D], i \neq i^*\}$ is statistically close in KeyGen and in $\widetilde{\mathsf{KeyGen}}$. This follows readily from Lemma 5.
2. Next, we argue that the joint distribution $\{\mathbf{P}_i : i \in [D]\}$ is statistically close in KeyGen and in $\widetilde{\mathsf{KeyGen}}$. This follows readily from secret sharing.
3. Finally, fix $\mathbf{P}_1, \ldots, \mathbf{P}_D$. We argue that the distribution of the remaining matrices is statistically close in KeyGen and in $\widetilde{\mathsf{KeyGen}}$. This follows from Lemma 6, which tells us that the output $\mathbf{U}_{i,j}$ of both $\mathsf{RightSample}$ in KeyGen and $\mathsf{LeftSample}$ in $\widetilde{\mathsf{KeyGen}}$, are statistically close to $\mathcal{D}_{\Lambda_q^{\mathbf{P}_i}(\mathbf{F}_{i,j}), \sigma\cdot\omega(\sqrt{\log n})}$.

   We can sample these $\mathbf{U}_{i,j}$ in $\widetilde{\mathsf{KeyGen}}$ applying $\mathsf{LeftSample}$ because $Y_{i,j} - X^*_{i,j} \neq 0$ whenever $i = i^*$ or whenever $j \neq j_i$.

Remark 3. Note that the "at most one" promise is crucial here, because when $Y_{i,j} - X^*_{i,j} = 0$, we cannot use $\mathsf{LeftSample}$ to sample a short matrix $\mathbf{U}_{i,j}$ such that $\left(\mathbf{A} \| \mathbf{A}\mathbf{R}_{i,j} + (Y_{i,j} - X^*_{i,j})\mathbf{G}\right) \mathbf{U}_{i,j} = \mathbf{P}_i$. If there exists "at most one" $j_i \in [\ell]$ such that $Y_{i,j_i} - X^*_{i,j_i} = 0$, we can sample a short matrix $\mathbf{U}_{i,j_i}$ from some discrete Gaussian distribution, and set $\mathbf{P}_i$ to be $\mathbf{P}_i := (\mathbf{A} \| \mathbf{A}\mathbf{R}_{i,j} + 0 \cdot \mathbf{G})\, \mathbf{U}_{i,j}$. Lemma 5 ensures that both $\mathbf{U}_{i,j}$ and $\mathbf{P}_i$ are correctly distributed. However, if there exist at least 2 indices j_i and $j_i' \in [\ell]$ such that $Y_{i,j_i} - X^*_{i,j_i} = Y_{i,j_i'} - X^*_{i,j_i'} = 0$, then there is more than one matrix that we cannot sample using $\mathsf{LeftSample}$, and the previous technique does not work anymore.

Lemma 17 (Game_2 to Game_3). *There exists an adversary $\mathcal{B}$ whose running time is roughly the same as that of $\mathcal{A}$ and such that*

$$|\mathsf{Adv}_2 - \mathsf{Adv}_3| \leq \mathsf{Adv}_{\mathcal{B}}^{\mathsf{dLWE}_{n,m+k+n,q,\chi}}$$

Note that only difference between Game_2 and Game_3 is that we switch the distribution of the inputs $(\mathbf{d}_0, \mathbf{d}_{D+1})$ to $\widetilde{\mathsf{Enc}}$ from LWE instances to random ones.

Proof. On input an LWE challenge

$$(\mathbf{A}, \mathbf{P}, \mathbf{d}_0, \mathbf{d}_{D+1})$$

where $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$ and $\mathbf{P} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times (k+n)}$ and $(\mathbf{d}_0, \mathbf{d}_{D+1})$ is either $(\mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top, \mathbf{s}^\top \mathbf{P} + \mathbf{e}'^\top)$ or random, $\mathcal{B}$ simulates $\mathcal{A}$ and proceeds as follows:

- runs $\widetilde{\mathsf{Setup}}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M}; \mathbf{A}, \mathbf{P}, \mathbf{X}^*_\beta)$ as in Game_2;
- answers $\mathcal{A}$'s private key queries by using $\widetilde{\mathsf{KeyGen}}$ as in Game_2;
- runs $\widetilde{\mathsf{Enc}}(\mathsf{mpk}, \mathbf{b}_\beta; \widetilde{\mathsf{msk}}, \mathbf{d}_0, \mathbf{d}_{D+1})$ to generate the challenge ciphertext;
- Finally, $\mathcal{A}$ guesses if it is interacting with a Game_2 or a Game_3 challenger. $\mathcal{B}$ outputs $\mathcal{A}$'s guess as the answer to the LWE challenge it is trying to solve.

When the LWE challenge is pseudorandom as in Definition 1, the adversary's view is as in Game_2. When the LWE challenge is random the adversary's view is as in Game_3. Therefore, $\mathcal{B}$'s advantage in solving LWE is the same as $\mathcal{A}$'s advantage in distinguishing Game_2 and Game_3.

Lemma 18 (Game_3 to Game_4). $|\mathsf{Adv}_3 - \mathsf{Adv}_4| = \mathrm{negl}(n)$

Proof. The differences between Game_3 and Game_4 are:

- In Game_3, $\mathbf{A} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$, and $\mathbf{T}_\mathbf{A} := \perp$, whereas in Game_4, $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow_{\mathrm{R}} \mathsf{TrapGen}(1^n, 1^m)$.
- In Game_3, the challenger answers the adversary's secret key using the $\widetilde{\mathsf{KeyGen}}$ algorithm, whereas he answers using the KeyGen algorithm in Game_4.

The proof is the same as the one of Lemma 16, by symmetry of the games.

Lemma 19 (Game_4). *We have* $|\mathsf{Adv}_4 - 1/2| = \mathrm{negl}(n)$

Proof. In Game_4, both the challenge ciphertext and the secret keys are independent of β. Moreover, by Lemma 1, we know that for all $i \in [D]$, for all $j \in [\ell]$ the two following distributions are statistically close

$$(\mathbf{A}, \mathbf{A}_{i,j}) \approx_s (\mathbf{A}, \mathbf{A}\mathbf{R}_{i,j} - X^*_{i,j}\mathbf{G})$$

where $\mathbf{A}_{i,j} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$, $\mathbf{R}_{i,j} \leftarrow_{\mathrm{R}} \{-1, 1\}^{m \times m}$, and $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$. Thus, the mpk does not leak any information on $\mathbf{X}^*_\beta$. Therefore, we get $|\mathsf{Adv}_4 - 1/2| = \mathrm{negl}(n)$.

4.4 Parameter Selection

- By Lemma 1, we need $m > (n+1)\log q + \omega(\log n)$
- By Lemma 3, and Lemma 4, we require $m = \Theta(n \log q)$
- By Lemma 5, we need $m \geq 2n \log q$

- By Lemma 6, we need $\sigma = O(\|\mathbf{T_A}\|_{\mathrm{GS}})$ and $\sigma = O(\|\mathbf{T_G}\|_{\mathrm{GS}} \cdot (1 + \|\mathbf{R}\|_2))$ where $\|\mathbf{T_A}\|_{\mathrm{GS}} = O(\sqrt{n \log q})$ and $\|\mathbf{R}\|_2 \leq 20\sqrt{m}$ with overwhelming probability in n, according to Lemma 3 and Lemma 5, respectively.
- For correctness of decryption, we require $\chi_{\max} \leq \frac{q}{4}\left(1 + D(1 + 20\sqrt{m}) \cdot s \cdot 2m\right)^{-1}$, where $s = \sigma \cdot \omega(\sqrt{\log n})$.

Consequently we take $m = \Theta(n \log q)$, $\sigma = O\left(\sqrt{n \log q}\right)$ and $\chi_{\max} = O\left(\frac{q}{D(n \log q)^{3/2} s}\right)$ where $s = \sigma \cdot \omega(\sqrt{\log n})$.

4.5 Putting Everything Together for Multi-dimensional Range Queries

When D denotes the number of dimensions and T the number of points in each, combining the preceding scheme with the reduction in Section 3.3, we get a scheme for $\mathsf{P}_{\text{CP-RANGE}}$ and $\mathsf{P}_{\text{KP-RANGE}}$ with ciphertexts and secret keys of sizes $O(D \log T)$ and running times $O(D \cdot \log T \cdot \mathrm{poly}(n))$ for encryption and key generation, $O((\log T)^D \cdot \mathrm{poly}(n))$ for decryption.

5 Shorter Ciphertexts and Secret Keys for Multi-dimensional Subset Queries

The predicate encryption scheme for $\mathsf{P}_{\text{AND-OR-EQ}}$ exhibited in Section 4 together with the reductions presented in Section 3 lead to efficient predicate encryption schemes for multi-dimensional subset queries.

Indeed, when D denotes the dimension and T denotes the size of the sets, we obtain a predicate encryption scheme for $\mathsf{P}_{\text{KP-SUBSET}}$ and $\mathsf{P}_{\text{CP-SUBSET}}$ with ciphertexts and secret keys of size $O(D \cdot T)$.

However, in this section we show how we can improve the size of the secret keys and ciphertexts in order to obtain:

- ciphertexts of size $O(D)$ for $\mathsf{P}_{\text{KP-SUBSET}}$
- secret keys of size $O(D)$ for $\mathsf{P}_{\text{CP-SUBSET}}$

5.1 Multi-dimensional Subset Queries, Ciphertext Policy

We can obtain a predicate encryption scheme for $\mathsf{P}_{\text{CP-SUBSET}}$ using the reduction to $\mathsf{P}_{\text{AND-OR-EQ}}$ defined in section 3.2, with secret keys of size $O(D \cdot T)$. We reduce the size of the secret keys size down to $O(D)$ using the fact that in each dimension, only one of the T matrices in the secret key is needed to decrypt.

On the one hand, for each dimension i, the reduction maps a point $z \in [T]$ into the vector $\mathbf{e}_z$. Therefore, for all $j \neq z$ the KeyGen algorithm generates a short matrix $\mathbf{U}_{i,j}$ which is a preimage of some target for the matrix $\left[\mathbf{A} \| \mathbf{A}_{i,j} + 0 \cdot \mathbf{G}\right]$.

On the other hand, a characteristic vector $\mathbf{w} \in \{0,1\}^T$, is mapped into a $-1, 1$ vector. Thus, for all j, the (i,j)'th component of the ciphertext is an LWE sample corresponding to the matrix $\left[\mathbf{A}_{i,j} + *\mathbf{G}\right], * \in \{-1, 1\}$.

Consequently, the matrices $\mathbf{U}_{i,j}$ for $j \neq z$ do not yield any useful information to decrypt the ciphertext. Therefore, we can remove them, and still satisfies correctness. Moreover, it is clear that removing parts of the secret key will not affect the security. Removing these matrices, we obtain a scheme whose secret keys have size $O(D)$, and whose decryption algorithm runs in time $O(D \cdot \text{poly}(n))$ (instead of $O(T^D \cdot \text{poly}(n))$).

Construction. Let $n \in \mathbb{N}$ be the security parameter and T, D positive integers. Let $q = q(n, T, D)$, $m = m(n, T, D)$ and $\chi_{\max} = \chi_{\max}(n, T, D)$ be positive integers, and $\sigma = \sigma(n, T, D)$ be a Gaussian parameter.

$\mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M})$: on a security parameter n, an attribute space $\mathcal{X} := \{0,1\}^{D \times T}$, a predicate space $\mathcal{Y} := \mathbb{Z}_q^D$, and a message space $\mathcal{M} := \{0,1\}^k$, do:

- Pick $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
- Pick $\mathbf{A}_{1,1}, \ldots, \mathbf{A}_{D,T} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$.
- Pick $\mathbf{P} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times (k+n)}$.
- Compute $(\mathbf{G}, \mathbf{T}_\mathbf{G})$ as defined in Lemma 4.
- Output: $\mathsf{mpk} := (\mathbf{P}, \mathbf{A}, \mathbf{A}_{1,1}, \ldots, \mathbf{A}_{D,T}, \mathbf{G}, \mathbf{T}_\mathbf{G})$ and $\mathsf{msk} := \mathbf{T}_\mathbf{A}$

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{X}, \mathbf{b})$: On input the public parameters mpk, an predicate matrix $\mathbf{X} \in \mathcal{X}$, and a message $\mathbf{b} \in \{0,1\}^k$, do:

- Pick $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$, and compute $\mathbf{c}_0 := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top$.
- for all $i \in [D]$ and all $j \in [T]$, do:
  - Pick $\mathbf{R}_{i,j} \leftarrow \{-1, 1\}^{n \times m}$.
  - Compute $\mathbf{c}_{i,j} := \mathbf{s}^\top \left(\mathbf{A}_{i,j} + X_{i,j} \mathbf{G} \right) + \mathbf{e}^\top \mathbf{R}_{i,j}$.
- Set $\mathbf{b}' := (\mathbf{b}, 0, \ldots, 0) \in \{0,1\}^{k+n}$, pick an $\mathbf{e}' \leftarrow_{\mathrm{R}} \chi^{k+n}$, and compute $\mathbf{c}_f := \mathbf{s}^\top \mathbf{P} + \mathbf{e}'^\top + \mathbf{b}'^\top \cdot \lfloor q/2 \rfloor$.
- Output: $\mathsf{ct} := (\mathbf{c}_0, \mathbf{c}_{1,1}, \ldots, \mathbf{c}_{D,T}, \mathbf{c}_f)$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{y})$: On input the public parameters mpk, the master secret key msk, and an attribute vector $\mathbf{y} \in \mathcal{Y}$, do:

- Secret share $\mathbf{P}$ as $\{\mathbf{P}_i, i \in [D]\}$, such that $\sum_{i=1}^{D} \mathbf{P}_i = \mathbf{P}$.
- For all $i \in [D]$:
  Sample a short matrix $\mathbf{U}_i \in \mathbb{Z}_q^{2m \times (k+n)}$ such that $\left[\mathbf{A} \| \mathbf{A}_{i,y_i} + \mathbf{G}\right] \mathbf{U}_i = \mathbf{P}_i$ using $\mathsf{RightSample}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{A}_{i,y_i} + \mathbf{G}, \mathbf{P}_i, \sigma)$ with $\sigma = O(\sqrt{n \log q})$.
- Output the secret key $\mathsf{sk}_\mathbf{Y} = (\mathbf{U}_1, \ldots, \mathbf{U}_D)$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_\mathbf{y}, \mathsf{ct})$: On input the public parameters mpk, a secret key $\mathsf{sk}_\mathbf{y} = (\mathbf{U}_1, \ldots, \mathbf{U}_D)$ for an attribute vector $\mathbf{y}$, and a ciphertext $\mathsf{ct} = (\mathbf{c}_0, \mathbf{c}_{1,1}, \ldots, \mathbf{c}_{D,T}, \mathbf{c}_f)$, do:

- Compute $\mathbf{d} := \mathbf{c}_f - \sum_{i=1}^{D} \left[\mathbf{c}_0 \| \mathbf{c}_{i,y_i}\right] \mathbf{U}_i \mod q$.
- Output the first k bits of $\left\lfloor \frac{\mathbf{d}}{q/2} \right\rceil$. For any $z \in [0,1]$, we denote by $\lfloor z \rceil$ the closest integer of z.

Correctness and Security. Correctness and security proofs follow readily from those of $\mathsf{P}_{\text{AND-OR-EQ}}$ in Section 4.

5.2 Multi-dimensional Subset Queries, Key-Policy

The following scheme is similar (however simpler) to the one presented in Section 4 for $\mathsf{P}_{\text{AND-OR-EQ}}$.

Overview. We begin with the special case $D = 1$. Given an attribute vector $x \in [T]^D$, the ciphertext is an LWE sample corresponding to the matrix $\left[\mathbf{A} \| \mathbf{A}_1 + x\mathbf{G}\right]$ and the message is masked using an LWE sample corresponding to a public matrix $\mathbf{P}$. The secret key corresponding to $\mathbf{Y} \in \{0,1\}^T$ is a collection of T short matrices $\mathbf{U}_1, \ldots, \mathbf{U}_T$ such that:

$$\begin{aligned} \left[\mathbf{A} \| \mathbf{A}_1 + j\mathbf{G}\right] \mathbf{U}_j = \mathbf{P} \quad & \text{if } \mathbf{Y}_j = 1 \\ \mathbf{U}_j = \perp \quad & \text{if } \mathbf{Y}_j = 0 \end{aligned}$$

For correctness, observe that if $\mathbf{Y}_x = 1$, then the decryptor can use $\mathbf{U}_x$ to recover the plaintext. However, since the decryptor does not know x, he will try to decrypt the ciphertext using each of $\mathbf{U}_1, \ldots, \mathbf{U}_T$. We will need to pad the plaintext with redundant zeroes so that the decryptor can identify the correct plaintext.

To establish security with respect to some selective challenge x^*, we will need to simulate the secret keys for all $\mathbf{Y}$ such that $\mathbf{Y}_{x^*} = 0$. Observe that for all $j = 1, \ldots, T$,

$$\mathbf{Y}_j = 1 \Longrightarrow j \neq x^*$$

We can then simulate $\mathbf{U}_j$ using an "all-but-one" simulation (while puncturing at x^*) exactly as in prior IBE schemes in [1]. In order to establish weak attribute-hiding, we adopt a similar strategy to that for inner product encryption in [4].

Higher Dimensions. Given an attribute vector $\mathbf{x} \in [T]^D$, the ciphertext is an LWE sample corresponding to the matrix $\left[\mathbf{A} \| \mathbf{A}_1 + x_1\mathbf{G} \| \cdots \| \mathbf{A}_D + x_D\mathbf{G}\right]$.

The secret key corresponding to $\mathbf{Y} \in \{0,1\}^{D \times T}$ is a collection of $D \cdot T$ short matrices $\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,T}$ such that for $j = 1, \ldots, T, i = 1, \ldots, D$:

$$\begin{aligned} \left[\mathbf{A} \| \mathbf{A}_i + j\mathbf{G}\right] \mathbf{U}_{i,j} = \mathbf{P}_i \quad & \text{if } \mathbf{Y}_{i,j} = 1 \\ \mathbf{U}_{i,j} = \perp \quad & \text{if } \mathbf{Y}_{i,j} = 0 \end{aligned}$$

where $\mathbf{P}_1, \ldots, \mathbf{P}_D$ is an additive secret-sharing of $\mathbf{P}$.

For correctness, observe that if $\mathbf{Y}_{1,x_1} = \mathbf{Y}_{2,x_2} = \ldots = \mathbf{Y}_{D,x_D} = 1$, then the decryptor can use $\mathbf{U}_{1,x_1}, \cdots, \mathbf{U}_{D,x_D}$ to recover the plaintext. As with the case $D = 1$, the decryptor will need to enumerate over all $\mathbf{x}' \in [T]^D$.

To simulate secret keys for $\mathbf{Y}$ with respect to some selective challenge $\mathbf{x}^*$, first we fix i^* such that $\mathbf{Y}_{k,x^*_{i^*}} = 0$. Without loss of generality, suppose $i^* = 1$, that is, $\mathbf{Y}_{1,x^*_1} = 0$. We then proceed as follows:

- Sample random short matrices $\mathbf{U}_{2,x_2^*}, \ldots, \mathbf{U}_{D,x_D^*}$, which in turn determines the shares $\mathbf{P}_2, \ldots, \mathbf{P}_D$.
- Define $\mathbf{P}_1 := \mathbf{P} - \sum_{i=2}^{D} \mathbf{P}_i$ and use an all-but-one simulation strategy to sample the remaining short matrices in the secret key.

Construction. Let $n \in \mathbb{N}$ be the security parameter and T, D be positive integers. Let $q = q(n)$, $m = m(n, T, D)$ and $\chi_{\max} = \chi_{\max}(n, q, T, D)$ be positive integers, and $\sigma = \sigma(n, q, T, D)$ be a Gaussian parameter.

$\mathsf{Setup}(1^n, \mathcal{X}, \mathcal{Y}, \mathcal{M})$: on n, $\mathcal{X} := \mathbb{Z}_q^D$, $\mathcal{Y} := \{0,1\}^{D \times T}$ and $\mathcal{M} := \{0,1\}^k$, do:

- Pick $(\mathbf{A}, \mathbf{T_A}) \leftarrow \mathsf{TrapGen}(1^n, 1^m, q)$.
- Pick $\mathbf{A}_1, \ldots, \mathbf{A}_D \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times m}$.
- Pick $\mathbf{P} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^{n \times (k+n)}$.
- Compute $(\mathbf{G}, \mathbf{T_G})$ as defined in Lemma 4.
- Output: $\mathsf{mpk} := (\mathbf{P}, \mathbf{A}, \mathbf{A}_1, \ldots, \mathbf{A}_D, \mathbf{G}, \mathbf{T_G})$ and $\mathsf{msk} := \mathbf{T_A}$

$\mathsf{Enc}(\mathsf{mpk}, \mathbf{x}, \mathbf{b})$: On input mpk, $\mathbf{x} \in \mathcal{X}$ and $\mathbf{b} \in \{0,1\}^k$:

- Pick $\mathbf{s} \leftarrow_{\mathrm{R}} \mathbb{Z}_q^n$, $\mathbf{e} \leftarrow_{\mathrm{R}} \chi^m$, and compute $\mathbf{c}_0 := \mathbf{s}^\top \mathbf{A} + \mathbf{e}^\top$.
- For all $i \in [D]$:
  - Pick $\mathbf{R}_i \leftarrow \{-1, 1\}^{m \times m}$.
  - Compute $\mathbf{c}_i := \mathbf{s}^\top \big(\mathbf{A}_i + x_i \mathbf{G} \big) + \mathbf{e}^\top \mathbf{R}_i$.
- Set $\mathbf{b}' := (\mathbf{b}, 0, \ldots, 0) \in \{0,1\}^{k+n}$, pick $\mathbf{e}' \leftarrow_{\mathrm{R}} \chi^{k+n}$, and compute $\mathbf{c}_f := \mathbf{s}^\top \mathbf{P} + \mathbf{e}'^\top + \mathbf{b}'^\top \cdot \lfloor q/2 \rfloor$.
- Output: $\mathsf{ct} := (\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_D, \mathbf{c}_f)$

$\mathsf{KeyGen}(\mathsf{mpk}, \mathsf{msk}, \mathbf{Y})$: On input mpk, msk, and $\mathbf{Y} \in \mathcal{Y}$, do:

- Secret share $\mathbf{P}$ as $\{\mathbf{P}_i, i \in [D]\}$, such that $\sum_{i=1}^{D} \mathbf{P}_i = \mathbf{P}$.
- For all $i \in [D]$ and $j \in [T]$:
  - If $Y_{i,j} = 1$ then sample a short matrix $\mathbf{U}_{i,j} \in \mathbb{Z}_q^{2m \times (k+n)}$ such that
$$\big[\mathbf{A} \| \mathbf{A}_i + j\mathbf{G}\big] \mathbf{U}_{i,j} = \mathbf{P}_i$$
    using $\mathsf{RightSample}(\mathbf{A}, \mathbf{T_A}, \mathbf{A}_i + j\mathbf{G}, \mathbf{P}_i, \sigma)$ with $\sigma = O(\sqrt{n \log q})$.
  - Otherwise set $\mathbf{U}_{i,j} := \perp$.
- Output the secret key $\mathsf{sk}_{\mathbf{Y}} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,T})$.

$\mathsf{Dec}(\mathsf{mpk}, \mathsf{sk}_{\mathbf{Y}}, \mathsf{ct})$: On input the public parameters mpk, a secret key $\mathsf{sk}_{\mathbf{Y}} = (\mathbf{U}_{1,1}, \ldots, \mathbf{U}_{D,T})$ for a predicate matrix $\mathbf{Y}$, and a ciphertext $\mathsf{ct} = (\mathbf{c}_0, \mathbf{c}_1, \ldots, \mathbf{c}_D, \mathbf{c}_f)$, do:

- For all $\mathbf{x}' = (x'_1, \ldots, x'_D) \in [T]^D$, compute $\mathbf{d}_{\mathbf{x}'} := \mathbf{c}_f - \sum_{i=1}^{D} \big[\mathbf{c}_0 \| \mathbf{c}_i\big] \mathbf{U}_{i,x'_i} \mod q$.

– If $\left\lfloor \frac{\mathbf{d}_{\mathbf{x}'}}{q/2} \right\rceil \in \{0,1\}^k \times 0^n$ for exactly one vector $\mathbf{x}' \in [T]^D$, then output the first k bits of $\left\lfloor \frac{\mathbf{d}_{\mathbf{x}'}}{q/2} \right\rceil$. For any $z \in [0,1]$, we denote by $\lfloor z \rceil$ the closest integer of z.
– Otherwise, abort.

We defer the proofs of correctness and security to the full version [12].

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient Lattice (H)IBE in the Standard Model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010)
2. Agrawal, S., Boneh, D., Boyen, X.: Lattice Basis Delegation in Fixed Dimension and Shorter-Ciphertext Hierarchical IBE. In: Rabin, T. (ed.) CRYPTO 2010. LNCS, vol. 6223, pp. 98–115. Springer, Heidelberg (2010)
3. Agrawal, S., Boyen, X., Vaikuntanathan, V., Voulgaris, P., Wee, H.: Functional Encryption for Threshold Functions (or, Fuzzy IBE) from Lattices. In: Public Key Cryptography, pp. 280–297 (2012)
4. Agrawal, S., Freeman, D.M., Vaikuntanathan, V.: Functional Encryption for Inner Product Predicates from Learning with Errors. In: Lee, D.H., Wang, X. (eds.) ASIACRYPT 2011. LNCS, vol. 7073, pp. 21–40. Springer, Heidelberg (2011)
5. Ajtai, M.: Generating Hard Instances of Lattice Problems (Extended Abstract). In: STOC, pp. 99–108 (1996)
6. Alwen, J., Peikert, C.: Generating Shorter Bases for Hard Random Lattices. In: STACS, pp. 75–86 (2009)
7. Boneh, D., Gentry, C., Gorbunov, S., Halevi, S., Nikolaenko, V., Segev, G., Vaikuntanathan, V., Vinayagamurthy, D.: Fully Key-Homomorphic Encryption, Arithmetic Circuit ABE and Compact Garbled Circuits. In: Nguyen, P.Q., Oswald, E. (eds.) EUROCRYPT 2014. LNCS, vol. 8441, pp. 533–556. Springer, Heidelberg (2014)
8. Boneh, D., Waters, B.: Conjunctive, Subset, and Range Queries on Encrypted Data. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 535–554. Springer, Heidelberg (2007)
9. Cash, D., Hofheinz, D., Kiltz, E., Peikert, C.: Bonsai Trees, or How to Delegate a Lattice Basis. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 523–552. Springer, Heidelberg (2010)
10. De Berg, M., Van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry. Springer, Heidelberg (2000)
11. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. SIAM J. Comput. **38**(1), 97–139 (2008)
12. Gay, R., Méaux, P., Wee, H.: Predicate Encryption for Multi-Dimensional Range Queries from Lattices. Cryptology ePrint Archive, Report 2014/965 (2014), http://eprint.iacr.org/
13. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: STOC, pp. 197–206 (2008)
14. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: STOC, pp. 545–554 (2013), also, Cryptology ePrint Archive, Report 2013/337

15. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: ACM Conference on Computer and Communications Security, pp. 89–98 (2006)
16. Impagliazzo, R., Levin, L.A., Luby, M.: Pseudo-random generation from one-way functions. In: STOC 1989 Proceedings of the Twenty-First Annual ACM Symposium on Theory of Computing, pp. 12–24. ACM, New York (1989)
17. Katz, J., Sahai, A., Waters, B.: Predicate Encryption Supporting Disjunctions, Polynomial Equations, and Inner Products. In: Smart, N.P. (ed.) EUROCRYPT 2008. LNCS, vol. 4965, pp. 146–162. Springer, Heidelberg (2008)
18. Micciancio, D., Peikert, C.: Trapdoors for Lattices: Simpler, Tighter, Faster, Smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)
19. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: STOC. pp. 84–93 (2005)
20. Shi, E., Bethencourt, J., Chan, H.T.H., Song, D.X., Perrig, A.: Multi-Dimensional Range Query over Encrypted Data. In: IEEE Symposium on Security and Privacy, pp. 350–364 (2007)
21. Xagawa, K.: Improved (Hierarchical) Inner-Product Encryption from Lattices. In: Public Key Cryptography, pp. 235–252 (2013)