Batched Threshold Predicate Encryption from Succinct LWE

Shayeef Murshid[1][0009-0008-7042-7911], Ramprasad Sarkar[2][0000-0002-9907-3319], and Mriganka Mandal[1][0000-0001-8691-4185]

[1] Cryptology and Security Research Unit
R C Bose Centre for Cryptology and Security
Indian Statistical Institute Kolkata, Kolkata, India-700108
{shayeef_r,mriganka}@isical.ac.in
[2] Department of Mathematics, Indian Institute of Information Technology Lucknow, Lucknow 226002, Uttar Pradesh, India
rpsarkar@iiitl.ac.in

Abstract. Predicate encryption extends attribute-based encryption by providing fine-grained access control while hiding the attribute associated with a ciphertext beyond what the secret keys held by an adversary necessarily reveal. Batched identity-based encryption enables authorization with respect to a designated collection of identities. We introduce batched threshold predicate encryption (BT-PE), which combines batch-scoped authorization, expressive predicate-based access control, and threshold decryption. We give a lattice-based construction supporting polynomial-size arithmetic-circuit predicates, compact batch digests, and compact per-party key shares. Decryption is possible only when the encrypted attribute belongs to the authorized batch, satisfies the required predicate, and a sufficiently large set of parties participates. We prove correctness and query-selective one-sided security under the ℓ-succinct LWE assumption, together with standard assumptions on digital signatures and compute-and-compare obfuscation.

Keywords: Predicate Encryption · Batched Encryption · Threshold Cryptography · Lattice-Based Cryptography

1 Introduction

Attribute-based encryption (ABE) and predicate encryption (PE) provide fine-grained cryptographic access control by associating ciphertexts with attributes and secret keys with access policies or predicates, so that decryption is possible only when the encrypted attribute satisfies the predicate encoded in the key. Beginning with the fuzzy identity-based encryption framework of Sahai and Waters [24] and the key-policy formulation of Goyal *et al.* [18], a long line of work has developed increasingly expressive constructions supporting rich access structures and general circuits [4, 7, 14, 28]. Predicate encryption further strengthens this paradigm by hiding information about the encrypted attribute itself beyond

what is necessarily revealed through the secret keys held by an adversary. Early formulations were given by Katz *et al.* [21] and were later extended toward stronger forms of predicate and attribute privacy [27]. This privacy makes PE particularly appealing for applications such as private query evaluation, oblivious access control, and anonymous credential systems where revealing the underlying attributes may itself compromise sensitive information.

Lattice-based constructions offer a particularly attractive route to expressive ABE and PE in the post-quantum setting through variants of the learning with errors (LWE) assumption [4,14,15]. Subsequent works also showed how one-sided secure PE for general functionality can be obtained from ABE using compute-and-compare obfuscation [17,32]. Despite this progress, traditional constructions typically provide persistent authorization. Once a user receives a key for a predicate f, that key can be applied to every ciphertext whose attribute satisfies f. In applications involving auditing, selective data release, time-dependent authorization, or dynamically formed collections of encrypted records, it is natural to require access to be simultaneously predicate dependent and batch scoped. Such authorization should apply only to a designated collection of ciphertexts rather than indefinitely to all ciphertexts satisfying the same predicate.

The recent notion of batched identity-based encryption (BIBE), introduced by Agarwal *et al.* [2], provides the latter form of control at the level of identities. A public digest succinctly represents a selected batch of identities, and the authority derives compact decryption material associated with this digest. A defining efficiency goal is that the decryption key should not grow linearly with the number of identities contained in the batch. Batched encryption has subsequently motivated work on batch and threshold-batch functionality, particularly in applications such as encrypted mempools and blockchain systems [1, 2, 5, 13]. In the lattice setting, Xiong *et al.* [33] showed how polynomial batch encoding, matrix commitments, and lattice preimage sampling can be combined to obtain batched identity-based encryption in the standard model from the succinct LWE assumption.

Batched IBE, nevertheless, expresses authorization only through membership: whether a ciphertext can be decrypted is determined by whether its identity belongs to the designated batch. Many natural applications require a stronger form of authorization. A ciphertext should belong to a specific dynamically selected batch, its hidden attribute should satisfy an expressive predicate, and the predicate itself should not destroy the attribute-privacy guarantee. Combining these properties with lattice-based batching is nontrivial. The batch-membership test, predicate evaluation, and attribute-hiding argument must all be compatible with the same lattice ciphertext and with succinct batch-dependent key material. This leads to our first motivating question.

> *Can predicate encryption be combined with succinct batch authorization from lattices while supporting expressive access policies?*

Threshold cryptography [9, 22, 26] provides a natural framework for distributing trust in cryptographic systems. In a T-out-of-N threshold encryption scheme,

any coalition of at least T parties can jointly decrypt, while any group of fewer than T parties learns no information about the protected plaintext. Classical ABE and PE, in contrast, typically assign decryption capability to a single key holder for each access policy, thereby concentrating trust in one party. This limitation is undesirable in settings such as private auctions, medical-record access, decentralized systems, and privacy-preserving authorization, where sensitive data should be released only through the cooperation of several authorized parties. Recent work on attribute-based threshold encryption [8] combines threshold decryption with policy-based access control, while related threshold techniques have also been explored for batched identity-based encryption [2, 5, 13]. This motivates the following question.

> *Can lattice-based predicate encryption simultaneously support batch encryption, T-out-of-N threshold decryption, expressive access policies, and succinct per-party key shares?*

1.1 Our Contributions

We introduce *batched threshold predicate encryption* (BT-PE), which combines predicate-based access control, batch-scoped authorization, and threshold decryption. In our setting, a ciphertext is associated with an attribute and a batch label, while a key instance is bound to a designated batch, a predicate, and the corresponding label. Decryption succeeds only when the encrypted attribute belongs to the authorized batch, satisfies the predicate, carries the correct label, and at least T parties from the same key instance participate. Our construction achieves the following properties.

- **Batch-scoped authorization.** A key is associated with a designated batch of attributes and can be used only for ciphertexts whose hidden attribute belongs to that batch.
- **Expressive predicates.** The construction supports polynomial-size arithmetic circuit predicates, allowing fine-grained access policies over encrypted attributes.
- **Threshold decryption.** The decryption capability is distributed among N parties. Any coalition of at least T parties from the same key instance can decrypt, while any collection of at most $T-1$ shares reveals no information about the underlying decryption capability.
- **Compact key shares and batch digests.** Each party holds a single key share of size $\mathsf{poly}(\lambda, |f_\lambda|, \log N, \log B)$, and a batch of B attributes is represented by a digest $\mathsf{dig}_S \in \mathbb{Z}_q^{\ell \times m}$ whose size grows only polylogarithmically with B and is independent of N and T.

Our construction is based on lattice techniques and protects the message through secure compute-and-compare obfuscation [16, 17, 32]. The threshold layer uses Shamir secret sharing [26], while digital signatures authenticate both the long-term key shares and their use in partial decryption. We formulate a query-selective one-sided security notion for BT-PE. The adversary may obtain at

most $T-1$ shares of any single key instance that authorizes both challenge attributes and may otherwise query key instances that authorize neither challenge attribute. Queries that authorize exactly one challenge attribute are not allowed. We prove security under the $2m^2$-succinct LWE assumption together with the security properties of Shamir secret sharing, post-quantum digital signatures, and compute-and-compare obfuscation. The result is summarized below.

Theorem 1 (Informal). *Assume the $2m^2$-succinct LWE assumption, secure compute-and-compare obfuscation, post-quantum secure digital signatures, and the stated properties of Shamir secret sharing. Then there exists a batched threshold predicate encryption scheme for polynomial-size arithmetic-circuit predicates with T-out-of-N threshold decryption that is query-selectively one-sided secure. Each secret-key share has size polynomial in λ, $|f_\lambda|$, $\log N$, and $\log B$, and each batch digest has size polynomial in λ and $\log B$.*

1.2 Related Work

Attribute-Based and Predicate Encryption. Attribute-based encryption (ABE) originates from the fuzzy identity-based encryption framework of Sahai and Waters [24] and the key-policy construction of Goyal *et al.* [18]. Subsequent works developed increasingly expressive access structures and general-circuit policies [4, 7, 14, 28]. Lattice-based ABE for general circuits was obtained from LWE by Gorbunov *et al.* [14], while Boneh *et al.* [4] developed key-homomorphic techniques that became fundamental in later lattice-based constructions. More recent work has obtained improved and nearly optimal parameters from succinct LWE [30, 31]. Predicate encryption (PE) strengthens policy-based encryption by protecting information about the encrypted attribute. Early formulations were given by Katz *et al.* [21] and strengthened toward predicate privacy by Shen *et al.* [27]. General-circuit lattice-based PE was studied by Gorbunov *et al.* [15], while compute-and-compare obfuscation has provided a useful route from one-sided secure ABE to expressive PE [17, 32].

Batched Encryption and Batched Access Control. Batched identity-based encryption was introduced by Agarwal *et al.* [2], initiating the study of encryption mechanisms that authorize decryption with respect to designated collections of identities. Subsequent works developed threshold and batch variants in both pairing-based and lattice-based settings [1, 5, 6, 13]. Sarkar *et al.* [25] extended this direction to batched attribute-based encryption and introduced both B-ABE and threshold batched ABE constructions from pairings. In the lattice setting, Xiong *et al.* [33] gave a batched IBE construction in the standard model from succinct LWE. Their polynomial-digest and matrix-commitment techniques provide tools for representing a batch within lattice-based encryption. Boneh *et al.* [5] observed that key-policy ABE directly yields batch decryption, gave a lattice-based batch decryption scheme with shorter parameters, and constructed threshold batch decryption from trilinear maps. These works primarily focus on identity-based or attribute-based authorization, whereas our setting combines batch-scoped authorization with expressive predicate evaluation.

Threshold Encryption and Policy-Based Threshold Access. Threshold cryptography distributes decryption authority among several parties so that sufficiently many participants are required to recover the plaintext [9,22,26]. Recent work has combined threshold decryption with richer forms of access control. Attribute-based threshold encryption was introduced in [8], where decryption requires a threshold number of parties whose policies authorize the ciphertext attribute. Threshold techniques have also been developed for batched identity-based encryption [2, 5, 13]. Our work lies at the intersection of these directions. We introduce batched threshold predicate encryption, where authorization is simultaneously restricted to a designated batch, governed by an expressive predicate, and distributed among multiple parties through threshold decryption. In particular, our construction separates the threshold mechanism from the underlying predicate-encryption functionality so that compact per-party shares can be used without requiring a separate encryption key for every authorized party.

2 Technical Overview

Notations. Bold uppercase letters (e.g., $\mathbf{A}, \mathbf{B}$) denote matrices, bold lowercase letters (e.g., $\mathbf{u}, \mathbf{v}$) denote vectors, and non-bold letters denote scalars. We write $\mathbf{J}$ for the all-ones matrix, whose dimensions are always clear from the context, and $\mathbf{I}_d$ for the $d \times d$ identity matrix. For a finite set S, we write $s \leftarrow S$ for uniform sampling from S, and $|S|$ for its cardinality. We use $\approx_s$ and $\approx_c$ for statistical and computational indistinguishability, respectively, and $\Delta(X, Y)$ for statistical distance. For $x \in \{0,1\}^n$, $x[a]$ denotes its a-th coordinate. Throughout, q denotes a prime lattice modulus and $\mathbb{Z}_q^* := \mathbb{Z}_q \setminus \{0\}$. For a vector $\mathbf{x}$, $\|\mathbf{x}\|_2$ denotes its Euclidean norm. For a matrix $\mathbf{R}$, $\|\mathbf{R}\|_\infty$ denotes its maximum-entry norm, $\|\mathbf{R}\|_2$ its spectral norm, and $\widetilde{\mathbf{R}}$ its Gram-Schmidt orthogonalization. Whenever norms are applied to elements of $\mathbb{Z}_q$, we identify them with their canonical representatives in $(-q/2, q/2]$. The notation $\lfloor \cdot \rceil$ denotes component-wise rounding to the nearest integer. All logarithms are base two. We use PPT and QPT for probabilistic polynomial-time and quantum polynomial-time algorithms, respectively. The cryptographic and lattice preliminaries used in this work are provided in Appendix 3.

This section gives a high-level description of our lattice-based construction of one-sided batched threshold predicate encryption ($\mathsf{BT\text{-}PE}$). The construction combines succinct batch authorization, predicate-based access control, threshold decryption, and one-sided attribute hiding in a post-quantum setting. The main idea is to generate a single short lattice decryption matrix,share this matrix among the decryption parties, and reconstruct it only inside a protected decryption circuit.

2.1 Towards Batched Threshold Predicate Encryption

A $\mathsf{BT\text{-}PE}$ scheme associates a ciphertext with a hidden attribute $x \in \{0,1\}^k$ and a batch label tg, while authorization is specified by a batch S, a predicate f_λ, and a threshold T. A batch S of size B is represented by a digest $\mathsf{dig} \leftarrow$

$\mathsf{Digest}(\mathsf{mpk}, S)$. Given this digest, a predicate f_λ, and a label tg, key generation produces N secret-key shares belonging to one common key instance. Any set of at least T parties can jointly decrypt provided that the encrypted attribute belongs to the authorized batch, satisfies the predicate, and carries the correct label. The algorithms comprising the BT-PE syntax are as follows.

- **Setup**. On input the security parameter, batch size B, threshold T, committee size N, attribute length k, and message length h, outputs a master public key mpk and master secret key msk.
- **Enc**. On input mpk, a message $\mu \in \{0,1\}^h$, an attribute $x \in \{0,1\}^k$, and a batch label tg, outputs a ciphertext ct.
- **Digest**. Deterministically compresses a batch $S \subseteq \{0,1\}^k$ of size B into a succinct digest dig.
- **KeyGen**. On input msk, a digest dig, a predicate f_λ, and a batch label tg, outputs secret-key shares $\{\mathsf{sk}_i\}_{i\in[N]}$.
- **PDec**. Given a secret-key share, the corresponding batch S, and a ciphertext, outputs a partial decryption share d_i when the batch, policy, and label checks succeed, and outputs $\perp$ otherwise.
- **Combine**. Given at least T valid partial decryption shares, reconstructs the plaintext μ.

Correctness requires that an authorized coalition recovers the message except with negligible probability whenever $x \in S$, $f_\lambda(x) = 0$, the labels agree, and at least T distinct parties from the same key instance participate. Our security notion additionally hides which of two challenge attributes was encrypted. In the query-selective one-sided security game, the adversary fixes x_0^*, x_1^* and tg^* in advance. It may obtain shares for at most $T-1$ predeclared parties of each key instance that authorizes both challenge attributes, and may otherwise query key instances that authorize neither. Queries authorizing exactly one challenge attribute are inadmissible.

Importantly, the same restriction applies to both secret-key queries and partial-decryption queries, irrespective of the ciphertext supplied to PDec. This is necessary because a partial-decryption token contains an authenticated share. Although the token is bound to a particular ciphertext core, the underlying share remains reusable information and the adversary must never obtain T shares of a jointly challenge-compatible key instance.

2.2 Designing BT-PE from Lattices

Our construction combines a lattice authorization mechanism with information-theoretic threshold sharing and two authentication layers. The lattice component uses polynomial batch encoding, matrix commitments, key-homomorphic evaluation, Gaussian trapdoor sampling, and succinct LWE. Threshold functionality is obtained by Shamir-sharing one short lattice preimage matrix among the N parties. A root signature certifies each long-term Shamir share, while an independently generated per-party signing key binds each use of that share to a

specific ciphertext core. Finally, compute-and-compare obfuscation releases the plaintext only when the protected circuit reconstructs the correct random lock.

The lattice authorization matrix. The central algebraic object is $\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}} = [\mathbf{A} \mid \mathsf{dig} \mid \mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 \mid \mathbf{B}_{f_\lambda}]$, where $\mathbf{B}_{f_\lambda} \leftarrow \mathsf{Eval}_{\mathsf{pk}}(f_\lambda, \mathbf{B}_1, \ldots, \mathbf{B}_k)$. The four blocks respectively encode the base lattice instance, batch authorization, label consistency, and predicate satisfaction. Instead of generating a separate lattice preimage for every threshold share, key generation samples a single short matrix $\mathbf{R} \in \mathbb{Z}^{4m \times \ell_{\mathsf{lock}}}$ satisfying

$$\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}}\mathbf{R} = \mathbf{D} \pmod q, \tag{1}$$

for the fixed public target $\mathbf{D} \in \mathbb{Z}_q^{\ell \times \ell_{\mathsf{lock}}}$. This matrix is the complete lattice decryption key for the corresponding batch, predicate, and label.

Succinct batch authorization. Attributes are injectively embedded into $\mathbb{Z}_q$ using $\iota : \{0,1\}^k \to \mathbb{Z}_q$. For a batch S, define $P_{\iota(S)}(X) = \prod_{a \in \iota(S)}(X - a)$ and let $\gamma \leftarrow \mathsf{GenCoeff}(\iota(S))$ encode its coefficients. The identity $\mathsf{ID}(x)\gamma^\top = P_{\iota(S)}(\iota(x))$ provides an algebraic membership test: it vanishes exactly when $x \in S$. The batch is represented in the key-generation matrix only through $\mathsf{dig}_S = -\mathbf{QV}(\gamma^\top \otimes \mathbf{I}_m)$. Consequently, the complete batch is not stored explicitly in the lattice portion of a secret-key share. We stress that a key instance is therefore bound to the digest dig and not to the batch S itself, which is exactly what keeps a share independent of B. Our one-sided security proof does not require the digest to be binding, because the challenge lock is removed from the ciphertext by a statistical argument (Lemma 21) that holds even against an adversary possessing a complete key instance for a colliding batch.

Threshold sharing of the lattice key. After sampling $\mathbf{R}$, key generation first encodes it over a Shamir field $\mathbb{F}_{\mathsf{SS}}$ and computes $(\mathsf{sh}_1, \ldots, \mathsf{sh}_N) \leftarrow \mathsf{SS.Share}_{T,N}(\mathsf{Encode}_{\mathsf{SS}}(\mathbf{R}))$. Each party therefore stores one Shamir share of the same lattice key. Any T valid shares reconstruct $\mathsf{Encode}_{\mathsf{SS}}(\mathbf{R})$ exactly, and hence, recover $\mathbf{R}$ inside the protected decryption circuit. In contrast, every set of at most $T-1$ shares is uniformly distributed and information-theoretically independent of $\mathbf{R}$. This property is central both to threshold correctness and to the simulation of jointly challenge-compatible queries in the security proof. This approach also avoids the usual lattice-noise problem associated with large secret-sharing reconstruction coefficients: Shamir interpolation is performed entirely over $\mathbb{F}_{\mathsf{SS}}$, and the exact matrix $\mathbf{R}$ is reconstructed *before* any noisy lattice decryption is performed.

Authenticating long-term and ciphertext-core-specific shares. Setup generates a root signature key pair. For each party i, key generation independently generates a token-signing pair $(\mathsf{tvk}_i, \mathsf{tsk}_i)$ and uses the root key to certify $(i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i)$. The identifier kid is common to all N shares produced by one invocation of KeyGen. When party i performs partial decryption, it signs a token containing $(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}})$ together with its batch, key-instance data, and Shamir share. Thus, the long-term root certificate authenticates the share itself, whereas the second signature authenticates the use of this share for one partic-

ular ciphertext core. We therefore refer to the resulting partial-decryption token as ciphertext-core-specific or encryption-instance-specific.

Encrypting a random comparison value. Encryption does not encode the message μ directly in the lattice ciphertext. Instead, it samples a uniform lock $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$ and encrypts ς in the lattice component. The inner ciphertext contains the base sample $\mathbf{c}_{\mathsf{in}} = \mathbf{sA} + \mathbf{e}_0$, attribute components $\{\mathbf{c}_a\}_{a\in[k]}$, a batch component $\mathbf{c}_{\mathsf{bat}}$, a label component $\mathbf{c}_{\mathsf{tg}}$, and

$$\mathbf{c}_{\mathsf{out}} = \mathbf{sD} + \mathbf{e}_{\mathsf{out}} + \left\lfloor \frac{q}{2} \right\rfloor \varsigma. \tag{2}$$

Encryption also samples a fresh $\mathsf{cid} \leftarrow \{0,1\}^{2\lambda}$. The public ciphertext is $\mathsf{ct} = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$. The decryption circuit underlying $\widetilde{Q}$ is hardwired with x, tg, cid, and $\mathsf{ct}_{\mathsf{inner}}$, and accepts only partial-decryption tokens authenticated for exactly this ciphertext core. The circuit also receives $(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}})$ as part of its input, so that the per-token signatures, which sign the ciphertext core, can be verified against precisely the values for which the circuit was built.

Compute-and-compare protection. The circuit $\mathcal{D}$ is used as the computation in a compute-and-compare program with comparison value ς and payload μ. The circuit itself only attempts to reconstruct a candidate lock ς'. If $\varsigma' = \varsigma$, the compute-and-compare program releases μ; otherwise, it outputs $\perp$. This separates the two cryptographic roles. The lattice layer protects the random lock ς, while the compute-and-compare layer uses this lock to protect the actual message μ.

Threshold decryption inside the protected circuit. Given T partial decryption tokens, $\mathcal{D}$ first verifies that their party indices are distinct and that they refer to the same batch, predicate, label, digest, and key-instance identifier. It verifies both the root certificate on every Shamir share and the ciphertext-core-specific token signature. It additionally checks $x \in S$ and $f_\lambda(x) = 0$. The circuit then reconstructs $\mathbf{R}$ from the T Shamir shares and verifies $\mathbf{F}_{\mathsf{dig}_S, f_\lambda, \mathsf{tg}}\mathbf{R} = \mathbf{D} \bmod q$. The polynomial batch identity and matrix commitment transform the batch component into $\mathbf{s}\,\mathsf{dig}_S$ plus bounded error, while key-homomorphic evaluation gives $\mathbf{sB}_{f_\lambda}$ plus bounded evaluation noise. Consequently, the circuit obtains $\overline{\mathbf{c}} = \mathbf{sF}_{\mathsf{dig}_S, f_\lambda, \mathsf{tg}} + \mathbf{e}_{\mathsf{ct}}$. Using Eq. (1), a single multiplication by the reconstructed matrix gives $\overline{\mathbf{c}}\mathbf{R} = \mathbf{sD} + \mathbf{e}_{\mathsf{ct}}\mathbf{R}$. Subtracting this from Eq. (2) leaves $\lfloor q/2 \rfloor \varsigma$ plus bounded error, from which the circuit recovers ς.

2.3 Correctness and Efficiency Guarantees

Correctness follows from three ingredients. First, perfect correctness of the signature scheme guarantees that all honestly generated root certificates and ciphertext-core-specific token signatures verify. Second, Shamir correctness ensures that any T honest shares reconstruct the exact matrix $\mathbf{R}$. Third, the lattice identities ensure that this reconstructed matrix decrypts the hidden lock with bounded error. For an authorized batch, the relation $\mathsf{ID}(x)\gamma^\top = 0$ together with the matrix-commitment identity gives $\mathbf{c}_{\mathsf{dig}} = \mathbf{s}\,\mathsf{dig}_S + \mathbf{e}_{\mathsf{dig}}$. Similarly, when $f_\lambda(x) = 0$, key-

homomorphic evaluation yields $\mathbf{c}_{f_\lambda} = \mathbf{s}\mathbf{B}_{f_\lambda} + \mathbf{e}_{f_\lambda}$. Combining the four ciphertext blocks, therefore, gives $\overline{\mathbf{c}} = \mathbf{s}\mathbf{F}_{\mathsf{dig}_S, f_\lambda, \mathsf{tg}} + \mathbf{e}_{\mathsf{ct}}$.

Since Shamir reconstruction returns the exact matrix $\mathbf{R}$, there is only one lattice error term, $\mathbf{e}_{\mathsf{ct}}\mathbf{R}$. In particular, threshold interpolation does not amplify lattice noise. Writing $E := c_{\mathsf{G}}\chi'\sqrt{\ell_{\mathsf{lock}}} + 2\sigma\sqrt{m\ell_{\mathsf{lock}}}\Delta_{\mathsf{ct}}$, our parameter condition $q > 4E + 2$ guarantees $E < (q-2)/4$. The extra additive 2 accounts for the fact that, for odd prime q, the bit 1 is encoded at $\lfloor q/2 \rfloor = (q-1)/2$. Under this bound, nearest-integer decoding recovers every coordinate of ς correctly, and perfect correctness of compute-and-compare obfuscation then releases μ.

Each party stores a single Shamir share of a $4m \times \ell_{\mathsf{lock}}$ matrix, together with fixed-size authentication information. The number of share coordinates is independent of N and T, while their representation length grows only through $\log Q_{\mathsf{SS}}$, and hence, polylogarithmically with N. The batch is represented in key generation through the single digest matrix dig_S rather than by explicitly storing all B attributes. Since the modulus q grows only polynomially with B, both the digest and every share are polylogarithmic in the batch size.

2.4 Achieving One-Sided Security

We next summarize the proof of Theorem 3. Let x_0^*, x_1^* and tg^* be the selectively chosen challenge attributes and label. For each fixed challenge bit b, the proof gradually removes the dependence of the challenge view on x_b^*, μ_b, and the random lock. The proof begins with a normalization step so that all subsequent leftover-hash, succinct LWE, and output-masking arguments apply to precisely the distributions assumed in their statements.

Normalizing the setup distribution. The real construction generates $(\mathbf{A}, \mathbf{T}_{\mathbf{A}})$ using TrapGen and samples the public matrix-commitment preimage $\mathbf{T}$ using the resulting trapdoor. The first hybrid replaces $\mathbf{A}$ by an exactly uniform matrix and replaces $\mathbf{T}$ by its canonical Gaussian coset distribution. Similarly, every honestly generated lattice key is viewed as a canonical Gaussian preimage conditioned on the same norm bound. The statistical closeness of TrapGen to uniform and of SamplePre to canonical Gaussian sampling justifies this normalization. From this hybrid onward, $\mathbf{A}$ is exactly uniform and $\mathbf{T}$ has exactly the distribution appearing in the succinct LWE assumption.

Programming the challenge instance. For the fixed attribute x_b^*, the proof samples short programming matrices and sets $\mathbf{Q} = \mathbf{A}\mathbf{U}_{\mathsf{bat}} - \mathbf{C}_b^*$, $\mathbf{L}_0 = \mathbf{A}\mathbf{U}_{\mathsf{tg}} - \mathsf{tg}^*\mathbf{L}_1$, $\mathbf{D} = \mathbf{A}\mathbf{U}_{\mathbf{D}}$, and $\mathbf{B}_a = \mathbf{A}\mathbf{S}_a^* - x_{b,a}^*\mathbf{G}$. Since $\mathbf{A}$ is now exactly uniform, the matrix leftover-hash lemma applies directly. For the binary matrices $\mathbf{U}_{\mathsf{bat}}$, $\mathbf{U}_{\mathsf{tg}}$, and $\mathbf{U}_{\mathbf{D}}$, the proof uses the standard conversion $\widetilde{\mathbf{U}} = 2\mathbf{U} - \mathbf{J} \in \{\pm 1\}^{m \times d}$, which is available because q is odd. The matrix leftover-hash lemma is then applied jointly to these matrices and to the concatenation of the attribute matrices $\mathbf{S}_1^*, \ldots, \mathbf{S}_k^*$. This yields public parameters statistically close to the honest distribution while preserving the correlated challenge-ciphertext terms.

Programming jointly compatible threshold shares. For a jointly compatible key instance, the adversary can obtain shares for at most $T - 1$ declared

parties. Shamir privacy states that these shares are jointly uniform and perfectly independent of the underlying lattice matrix $\mathbf{R}$. The proof therefore samples the exposed shares directly without generating $\mathbf{R}$. The conditional-completion property shows that these prescribed shares can be extended to a full honest sharing of any chosen $\mathbf{R}$, while explainability recovers sharing randomness with exactly the honest distribution. Since admissibility prevents the adversary from ever obtaining an undeclared compatible share, neither the underlying $\mathbf{R}$ nor the unexposed shares need actually be generated in the simulation. Token-signing key pairs, root certificates, and ciphertext-core-specific partial-decryption signatures are generated honestly from these programmed shares. They therefore constitute ordinary randomized post-processing of the perfectly simulated Shamir view.

Alternative trapdoors for incompatible queries. Every other admissible key query is incompatible with both challenge attributes. Relative to the fixed x_b^*, at least one of batch membership, label consistency, or predicate satisfaction must fail.
If $x_b^* \notin S$, the batch polynomial evaluates to a nonzero scalar and the programmed digest has the form $\mathsf{dig}_S = \mathbf{A}\mathbf{R}_S + \beta_S\mathbf{G}$ with $\beta_S \in \mathbb{Z}_q^*$. The public gadget trapdoor can therefore be used to derive a trapdoor for the batch prefix. If $\mathsf{tg} \neq \mathsf{tg}^*$, then $\mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 = \mathbf{A}\mathbf{U}_{\mathsf{tg}} + (\mathsf{tg} - \mathsf{tg}^*)\mathbf{L}_1$, and the trapdoor for $\mathbf{L}_1$ supplies the required alternative trapdoor. Finally, when $f_\lambda(x_b^*) = 1$, $\mathsf{Eval}_{\mathsf{sim}}$ gives $\mathbf{B}_{f_\lambda} = \mathbf{A}\mathbf{S}_{f_\lambda} - \mathbf{G}$, so the public gadget trapdoor again applies. In all three cases the Gaussian width σ is chosen to dominate the Gram-Schmidt norm τ_{sim} of the resulting alternative trapdoors, which is what makes these simulated keys statistically close to canonical Gaussian preimages. In the latter two cases, the proof applies $\mathsf{ExtendLeft}$ to the entire preceding prefix of $\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}}$, with suitable zero padding, so that the derived trapdoor respects the actual block order $[\mathbf{A} \mid \mathsf{dig} \mid \mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 \mid \mathbf{B}_f]$. Canonical Gaussian sampling makes all such keys statistically indistinguishable from honest key generation.

Replacing the challenge LWE sample. After the preceding hybrids, every admissible key and partial-decryption query can be answered without the master trapdoor $\mathbf{T}_\mathbf{A}$. The challenge component $\mathbf{c}_{\mathsf{in}}^* = \mathbf{s}\mathbf{A}+\mathbf{e}_0$ can therefore be embedded directly into the $2m^2$-succinct LWE challenge.
Because setup has already been normalized, the succinct LWE challenge provides exactly the required distribution: uniform $\mathbf{A}$, public $\mathbf{W}$, and a canonical Gaussian matrix-commitment preimage $\mathbf{T}$. The assumption then replaces $\mathbf{c}_{\mathsf{in}}^*$ by an independent uniform row while preserving the entire public commitment state.

Masking the challenge ciphertext and inserting the zero lock. Once $\mathbf{c}_{\mathsf{in}}^*$ is uniform, output masking can be applied directly because its hypothesis also requires uniform $\mathbf{A}$. The batch and tag components are first replaced by independently uniform vectors. Separately, since $\mathbf{D} = \mathbf{A}\mathbf{U}_\mathbf{D}$, the rerandomized output term is statistically close to uniform as well. Translation by $\lfloor q/2 \rfloor \varsigma$ preserves uniformity, so the lattice ciphertext may be changed from an encryption of the random lock ς to an encryption of the all-zero string. This separation is useful in the proof as the batch and tag components are removed first, and only then the lock is removed from $\mathbf{c}_{\mathsf{out}}^*$. At the end of these hybrids, every value

in the ciphertext core and every value hardwired into the decryption circuit is independent of the uniformly sampled comparison value ς.

Simulating the compute-and-compare object. The final computational transition is formulated as an explicit reduction to compute-and-compare security. The reduction constructs the zero-lock ciphertext core and the corresponding circuit before sampling the independent comparison value. It then receives either the real compute-and-compare obfuscation or the simulator output and passes this candidate object to $\mathcal{A}$.

After the challenge, the adversary may continue to issue adaptive key and partial-decryption queries. These queries need not be regarded as static auxiliary information: the reduction answers them itself. In the zero-lock hybrid, all oracle response mechanisms are independent of ς. Compatible shares use the programmed Shamir state, incompatible shares use the alternative trapdoors, and ciphertext-core-specific signatures depend only on the ciphertext core supplied in the query. Hence, the complete adaptive continuation can be simulated without knowledge of the comparison value, establishing the compute-and-compare transition.

Removing the challenge bit. After replacing the compute-and-compare object by its simulator, the proof compares the experiments for $b = 0$ and $b = 1$. The remaining dependence on x_b^* in the programmed batch matrix is removed by another matrix leftover-hash argument, again converting the binary programming matrix via $\widetilde{\mathbf{U}}_{\mathsf{bat}} = 2\mathbf{U}_{\mathsf{bat}} - \mathbf{J}$. For the attribute blocks, the proof applies the matrix leftover-hash lemma jointly to the concatenated sign matrix $[\mathbf{S}_1^* \mid \cdots \mid \mathbf{S}_k^*]$, conditioned on the already uniform value $\mathbf{c}_{\mathsf{in}}^*$. This simultaneously removes all dependence on the coordinates of x_b^*. The label programming and target matrix are independent of b, while the batch and tag ciphertext components are already uniform and the output component contains the all-zero lock.

Finally, adaptive oracle transcripts are compared inductively. Admissibility is defined jointly with respect to x_0^* and x_1^* and is therefore independent of b. Compatible queries return uniformly programmed Shamir shares, whereas incompatible queries return canonical Gaussian lattice keys. The authentication layers apply the same response algorithms to these corresponding values. Thus, every successive query preserves statistical indistinguishability, and the final experiments for $b = 0$ and $b = 1$ are statistically indistinguishable. This establishes query-selective one-sided security.

3 Preliminaries

This section recalls the cryptographic and lattice-theoretic tools used in our construction. In particular, we use compute-and-compare obfuscation, post-quantum digital signatures, Shamir threshold secret sharing, standard lattice trapdoor algorithms, key-homomorphic evaluation, matrix commitments, and the succinct LWE assumption.

3.1 Compute-and-Compare Obfuscation

We recall the notion of compute-and-compare obfuscation [16, 17, 32], also referred to as lockable obfuscation. A compute-and-compare program $\mathsf{CC}[C, \varsigma, \mu]$ evaluates a circuit C on an input z and outputs the message μ if $C(z) = \varsigma$, and outputs $\perp$ otherwise. Here, ς is called the *lock* or *comparison value*.

Syntax. Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of polynomial-size circuits, let $\mathcal{M} = \{\mathcal{M}_\lambda\}_{\lambda\in\mathbb{N}}$ be a sequence of message spaces, and let $\ell_{\mathsf{lock}} = \ell_{\mathsf{lock}}(\lambda)$ denote the lock length. A compute-and-compare obfuscator for $\mathcal{C}$ and $\mathcal{M}$ is a pair of efficient algorithms $\prod_{\mathsf{CC}} = (\mathsf{CC.Obf}, \mathsf{CC.Eval})$ with the following syntax.

- $\widetilde{C} \leftarrow \mathsf{CC.Obf}(1^\lambda, C, \mu, \varsigma)$. On input the security parameter 1^λ, a circuit $C \in \mathcal{C}_\lambda$, a message $\mu \in \mathcal{M}_\lambda$, and a lock $\varsigma \in \{0,1\}^{\ell_{\mathsf{lock}}}$, the algorithm outputs an obfuscated circuit $\widetilde{C}$.
- $\mu' \vee \perp \leftarrow \mathsf{CC.Eval}(\widetilde{C}, z)$. On input an obfuscated circuit $\widetilde{C}$ and an input z for C, the algorithm outputs either a message $\mu' \in \mathcal{M}_\lambda$ or the failure symbol $\perp$.

We require that $\prod_{\mathsf{CC}} = (\mathsf{CC.Obf}, \mathsf{CC.Eval})$ satisfy the following properties.

- **Perfect correctness.** For every $\lambda \in \mathbb{N}$, every $C \in \mathcal{C}_\lambda$, every $\mu \in \mathcal{M}_\lambda$, every $\varsigma \in \{0,1\}^{\ell_{\mathsf{lock}}}$, and every valid input z,

$$\mathsf{CC.Eval}\big(\mathsf{CC.Obf}(1^\lambda, C, \mu, \varsigma), z\big) = \begin{cases} \mu, & C(z) = \varsigma, \\ \perp, & C(z) \neq \varsigma. \end{cases}$$

- **Security.** There exists an efficient simulator $\mathsf{Sim}_{\mathsf{CC}}$ such that, for every QPT adversary and every efficiently samplable joint distribution (C, μ, aux) that is independent of a uniformly sampled lock $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$, we have

$$\big(\mathsf{aux}, \mathsf{CC.Obf}(1^\lambda, C, \mu, \varsigma)\big) \approx_c \big(\mathsf{aux}, \mathsf{Sim}_{\mathsf{CC}}(1^\lambda, 1^{|C|}, 1^{|\mu|})\big).$$

Perfectly correct compute-and-compare obfuscators for polynomial-size general circuits can be obtained from the learning with errors (LWE) assumption [16, 17, 32]. We use a post-quantum secure instantiation and choose $\ell_{\mathsf{lock}} \geq \lambda$ so that the uniformly sampled lock has sufficient entropy for the security guarantee above.

3.2 Post-Quantum Digital Signatures

We recall the standard notion of digital signatures. In our construction, digital signatures are used to authenticate threshold key shares and partial-decryption tokens. In particular, they allow the decryption circuit to verify that the supplied shares and tokens were generated by the appropriate authority or party.

Syntax. A digital signature scheme is a tuple of efficient algorithms $\prod_{\mathsf{SIG}} = (\mathsf{SIG.Gen}, \mathsf{SIG.Sign}, \mathsf{SIG.Ver})$ with the following syntax.

– $(\mathsf{vk}, \mathsf{ssk}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$. On input the security parameter 1^λ, the key-generation algorithm outputs a signing key ssk and a verification key vk.
– $\sigma \leftarrow \mathsf{SIG.Sign}(\mathsf{ssk}, M)$. On input a signing key ssk and a message M, the signing algorithm outputs a signature σ.
– $b \leftarrow \mathsf{SIG.Ver}(\mathsf{vk}, M, \sigma)$. On input a verification key vk, a message M, and a signature σ, the deterministic verification algorithm outputs $b \in \{0, 1\}$.

We require $\prod_{\mathsf{SIG}}$ to satisfy the following properties.

- **Perfect correctness.** For every $\lambda \in \mathbb{N}$, every $(\mathsf{vk}, \mathsf{ssk}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$, and every message M, $\mathsf{SIG.Ver}\big(\mathsf{vk}, M, \mathsf{SIG.Sign}(\mathsf{ssk}, M)\big) = 1$.
- **Post-quantum EUF-CMA security.** For every QPT adversary $\mathcal{A}$ with classical oracle access to $\mathsf{SIG.Sign}(\mathsf{ssk}, \cdot)$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\Pr\left[\begin{array}{l|l} \mathsf{SIG.Ver}(\mathsf{vk}, M^*, \sigma^*) = 1 & (\mathsf{vk}, \mathsf{ssk}) \leftarrow \mathsf{SIG.Gen}(1^\lambda), \\ \wedge\ M^* \notin \mathcal{Q} & (M^*, \sigma^*) \leftarrow \mathcal{A}^{\mathsf{SIG.Sign}(\mathsf{ssk}, \cdot)}(\mathsf{vk}) \end{array}\right] \leq \mathsf{negl}(\lambda),$$

  where $\mathcal{Q}$ denotes the set of messages submitted by $\mathcal{A}$ to its signing oracle.

Digital signature schemes satisfying existential unforgeability under chosen-message attacks can be constructed from one-way functions [12]. For our construction, we use a perfectly correct post-quantum secure instantiation, which may be instantiated from standard post-quantum assumptions such as lattice-based assumptions.

3.3 Shamir Threshold Secret Sharing

We recall Shamir threshold secret sharing [26], which provides an information-theoretic mechanism for distributing a secret among multiple parties. In a T-out-of-N sharing, any set of at least T parties can reconstruct the secret, while every set of at most $T - 1$ parties obtains no information about it. Since this privacy is information-theoretic and does not rely on any computational hardness assumption, it remains secure even against quantum adversaries. We apply the scheme coordinate-wise to matrix-valued secrets.

Syntax. Let Q_{SS} be a prime, let $\mathbb{F}_{\mathsf{SS}} := \mathbb{F}_{Q_{\mathsf{SS}}}$, and let $1 \leq T \leq N < Q_{\mathsf{SS}}$. The Shamir threshold secret-sharing scheme consists of the algorithms $\mathsf{SS.Share}_{T,N}$ and $\mathsf{SS.Reconst}_{T,N}$ described below.

– $(\mathsf{sh}_1, \ldots, \mathsf{sh}_N) \leftarrow \mathsf{SS.Share}_{T,N}(\mathbf{M})$. On input a matrix $\mathbf{M} \in \mathbb{F}_{\mathsf{SS}}^{a \times b}$, the sharing algorithm independently chooses, for every coordinate $(u, v) \in [a] \times [b]$, a uniformly random polynomial

$$p_{u,v}(X) = M_{u,v} + r_{u,v,1}X + \cdots + r_{u,v,T-1}X^{T-1} \in \mathbb{F}_{\mathsf{SS}}[X]$$

  of degree at most $T - 1$. Party $i \in [N]$ receives the matrix share $\mathsf{sh}_i \in \mathbb{F}_{\mathsf{SS}}^{a \times b}$ defined by $\mathsf{sh}_i[u, v] := p_{u,v}(i)$.

– $\mathbf{M} \leftarrow \mathsf{SS.Reconst}_{T,N}(\{(i, \mathsf{sh}_i)\}_{i \in U})$. On input shares from a set $U \subseteq [N]$ with $|U| \geq T$, the reconstruction algorithm interpolates the corresponding polynomials coordinate-wise and recovers the shared matrix $\mathbf{M}$.

We use the following standard properties of Shamir secret sharing.

- **Perfect correctness.** For every $\mathbf{M} \in \mathbb{F}_{\mathsf{SS}}^{a \times b}$ and every $U \subseteq [N]$ with $|U| \geq T$, $\mathsf{SS.Reconst}_{T,N}(\{(i, \mathsf{sh}_i)\}_{i \in U}) = \mathbf{M}$ with probability one.
- **Perfect privacy of unauthorized shares.** For every set $C \subseteq [N]$ with $|C| \leq T - 1$, the joint distribution $\{\mathsf{sh}_i\}_{i \in C}$ is uniform over $(\mathbb{F}_{\mathsf{SS}}^{a \times b})^{|C|}$ and is statistically independent of the shared secret $\mathbf{M}$. Consequently, any coalition of fewer than T parties learns no information about $\mathbf{M}$, even with unbounded or quantum computation.
- **Efficient conditional completion.** There exists an efficient algorithm $\mathsf{SS.Complete}_{T,N}$ such that, for every $C \subseteq [N]$ with $|C| \leq T - 1$, every secret $\mathbf{M}$, and every admissible collection $\{(i, \mathsf{sh}_i)\}_{i \in C}$, the algorithm generates the remaining shares according to the distribution of an honest Shamir sharing of $\mathbf{M}$ conditioned on the prescribed shares.
- **Explainability.** There exists an efficient algorithm $\mathsf{SS.Explain}_{T,N}$ that, given the secret $\mathbf{M}$ and a complete valid sharing, recovers the underlying sharing polynomials and hence, the randomness defining the corresponding invocation of $\mathsf{SS.Share}_{T,N}$.
- **Compact shares.** Each party receives a single matrix $\mathsf{sh}_i \in \mathbb{F}_{\mathsf{SS}}^{a \times b}$. Hence, the number of field elements in an individual share is independent of the total number of parties N and the threshold T.

In our construction, Shamir sharing is applied to an encoding of a short matrix $\mathbf{R} \in \mathbb{Z}^{4m \times \ell_{\mathsf{lock}}}$. We choose the prime $Q_{\mathsf{SS}} > N$ large enough so that every admissible entry of $\mathbf{R}$ has a unique centered representation in $\mathbb{F}_{\mathsf{SS}}$. We denote the coordinatewise embedding and its inverse by $\mathsf{Encode}_{\mathsf{SS}}$ and $\mathsf{Decode}_{\mathsf{SS}}$, respectively. The precise choice of Q_{SS} is given together with the construction parameters.

3.4 Lattice Preliminaries

For a matrix $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m}$, define $\Lambda_q^{\perp}(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \bmod q\}$. For $\mathbf{u} \in \mathbb{Z}_q^{\ell}$, define the affine coset $\Lambda_q^{\mathbf{u}}(\mathbf{A}) := \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{u} \bmod q\}$. For a target matrix $\mathbf{U} = (\mathbf{u}_1 | \cdots | \mathbf{u}_d) \in \mathbb{Z}_q^{\ell \times d}$, we write $D_\sigma(\Lambda_q^{\mathbf{U}}(\mathbf{A}))$ for the distribution on matrices whose j-th column is sampled independently from $D_\sigma(\Lambda_q^{\mathbf{u}_j}(\mathbf{A}))$. We write $\mathcal{D}_{\mathbb{Z},\sigma}$ for the discrete Gaussian distribution over $\mathbb{Z}$ with parameter $\sigma > 0$, and $\mathcal{D}_{\mathbb{Z}^d,\sigma}$ for its d-fold product distribution. The quantities $\chi, \chi' > 0$ used later denote Gaussian widths. Thus, for example, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^{1 \times m},\chi}$ means that the coordinates of $\mathbf{e}$ are independent discrete Gaussians of parameter χ.

Definition 1 (Learning with Errors [23]). *Fix integers ℓ, m, a prime q, and a noise distribution $\mathcal{D}$ over $\mathbb{Z}$. The $(\ell, m, q, \mathcal{D})$-LWE problem is to distinguish the*

distributions $(\mathbf{A}, \mathbf{sA} + \mathbf{e})$ *and* $(\mathbf{A}, \mathbf{u})$*, where* $\mathbf{A} \leftarrow \mathbb{Z}_q^{\ell \times m}$*,* $\mathbf{s} \leftarrow \mathbb{Z}_q^{1 \times \ell}$*,* $\mathbf{e} \leftarrow \mathcal{D}^m$*, and* $\mathbf{u} \leftarrow \mathbb{Z}_q^{1 \times m}$ *are independently sampled. We use* $m = \Theta(\ell \log q)$ *throughout.*

Gadget Matrices. Let $\kappa := \lceil \log q \rceil$ and define $\mathbf{g}' := (1, 2, \ldots, 2^{\kappa-1}) \in \mathbb{Z}_q^{1 \times \kappa}$. For every positive integer d, define the unpadded gadget matrix $\mathbf{G}_d := \mathbf{I}_d \otimes \mathbf{g}' \in \mathbb{Z}_q^{d \times d\kappa}$. For the lattice row dimension ℓ, we use the padded gadget matrix $\mathbf{G} := [\mathbf{G}_\ell \mid \mathbf{0}] \in \mathbb{Z}_q^{\ell \times m}$ whenever $m > \ell\kappa$. The lattice $\Lambda_q^\perp(\mathbf{G})$ admits a publicly known short basis $\mathbf{T}_\mathbf{G}$. For every d, let $\mathbf{G}_d^{-1}$ denote the deterministic binary decomposition map satisfying $\mathbf{G}_d \mathbf{G}_d^{-1}(\mathbf{U}) = \mathbf{U}$ for every conformable matrix $\mathbf{U}$.

Batch Digest Coefficients. The algorithm $\mathsf{GenCoeff}$ compresses a batch $S \subseteq \mathbb{Z}_q$ of size B by forming the polynomial $P_S(X) = \prod_{a \in S}(X - a) = \sum_{i=0}^{B} y_i X^i$ and gadget-decomposing its coefficient vector to obtain $\gamma := \left(\mathbf{G}_{B+1}^{-1}((\mathbf{y}')^\top)\right)^\top \in \{0,1\}^{1 \times \kappa(B+1)}$, where $\mathbf{y}' := (y_0, \ldots, y_B) \in \mathbb{Z}_q^{1 \times (B+1)}$ and $\mathbf{G}_{B+1} := \mathbf{I}_{B+1} \otimes \mathbf{g}' \in \mathbb{Z}_q^{(B+1) \times \kappa(B+1)}$. For $\mathsf{ID}(a) = (1, a, \ldots, a^B) \otimes \mathbf{g}'$, gadget decomposition gives $\mathsf{ID}(a)\gamma^\top = P_S(a)$, so $\mathsf{ID}(a)\gamma^\top = 0$ when $a \in S$, while for $a \notin S$,it lies in $\mathbb{Z}_q^*$.

3.5 Lattice Trapdoors

Lemma 1 (Lattice Trapdoor Algorithms [4]). *Let* λ *be a security parameter and let* ℓ, m, q *be lattice parameters. There exist efficient algorithms with the following properties:*

- $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^\ell, 1^m, q)$*. For* $m = \Theta(\ell \log q)$*, the algorithm outputs a full-rank matrix* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m}$ *that is statistically close to uniform, together with a basis* $\mathbf{T}_\mathbf{A} \in \mathbb{Z}^{m \times m}$ *of* $\Lambda_q^\perp(\mathbf{A})$ *satisfying* $\|\widetilde{\mathbf{T}}_\mathbf{A}\| = O(\sqrt{\ell \log q})$*, except with negligible probability.*
- $\mathbf{X} \leftarrow \mathsf{SamplePre}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{U}, \sigma)$*. Given* $\mathbf{U} \in \mathbb{Z}_q^{\ell \times d}$ *and* $\sigma \geq \|\widetilde{\mathbf{T}}_\mathbf{A}\| \omega(\sqrt{\log m})$*, the algorithm outputs* $\mathbf{X} \in \mathbb{Z}^{m \times d}$ *distributed statistically close to* $D_\sigma(\Lambda_q^\mathbf{U}(\mathbf{A}))$*.*
- $\mathbf{X} \leftarrow \mathsf{SampleRight}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B}, \mathbf{U}, \sigma)$*. For* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m_1}$*,* $\mathbf{B} \in \mathbb{Z}_q^{\ell \times m_2}$*, and* $\mathbf{U} \in \mathbb{Z}_q^{\ell \times d}$*, the algorithm outputs* $\mathbf{X} \in \mathbb{Z}^{(m_1+m_2) \times d}$ *distributed statistically close to* $D_\sigma(\Lambda_q^\mathbf{U}([\mathbf{A} \mid \mathbf{B}]))$ *for a sufficiently large Gaussian parameter* σ*.*
- $\mathbf{T}_{[\mathbf{A}|\mathbf{B}]} \leftarrow \mathsf{ExtendRight}(\mathbf{A}, \mathbf{T}_\mathbf{A}, \mathbf{B})$*. Given* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m_1}$*, a trapdoor for* $\mathbf{A}$*, and an arbitrary conformable matrix* $\mathbf{B} \in \mathbb{Z}_q^{\ell \times m_2}$*, the algorithm outputs a trapdoor for* $[\mathbf{A} \mid \mathbf{B}]$*.*
- $\mathbf{T}_{[\mathbf{A}|\mathbf{AR}+\mathbf{B}]} \leftarrow \mathsf{ExtendLeft}(\mathbf{A}, \mathbf{B}, \mathbf{T}_\mathbf{B}, \mathbf{R})$*. Let* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m_1}$*,* $\mathbf{B} \in \mathbb{Z}_q^{\ell \times m_2}$*, and* $\mathbf{R} \in \mathbb{Z}^{m_1 \times m_2}$*, and let* $\mathbf{T}_\mathbf{B}$ *be a short basis of* $\Lambda_q^\perp(\mathbf{B})$*. Then the algorithm outputs a trapdoor for* $[\mathbf{A} \mid \mathbf{AR} + \mathbf{B}]$ *satisfying* $\|\widetilde{\mathbf{T}}_{[\mathbf{A}|\mathbf{AR}+\mathbf{B}]}\| \leq \|\widetilde{\mathbf{T}}_\mathbf{B}\|(1 + \|\mathbf{R}\|_2)$ *up to the standard trapdoor-sampling factors.*

3.6 Auxiliary Lattice Lemmas

Lemma 2 (Gaussian Tail Bound). *Let* $\sigma > 0$ *and let* $\mathbf{x} \leftarrow \mathcal{D}_{\mathbb{Z}^d,\sigma}$*. For every* $\lambda \in \mathbb{N}$*,* $\Pr\left[\|\mathbf{x}\| > \sigma\sqrt{d\lambda}\right] \leq d \cdot 2^{-\lambda}$*. In particular, for* $d = \mathsf{poly}(\lambda)$*, the failure*

probability is negligible. We occasionally write $c_{\mathsf{G}} := \sqrt{\lambda}$ *and use the shorthand* $\|\mathbf{x}\| \leq c_{\mathsf{G}}\sigma\sqrt{d}$.

Lemma 3 (Gaussian Matrix Norm Bound [4]). *Let* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times M}$ *and* $\mathbf{U} \in \mathbb{Z}_q^{\ell \times d}$. *If* $\mathbf{R} \leftarrow D_\sigma(\Lambda_q^{\mathbf{U}}(\mathbf{A}))$, *then, except with negligible probability,* $\|\mathbf{R}\|_2 \leq \sigma\sqrt{Md}$. *Moreover, if* $\mathbf{S} \leftarrow \{\pm 1\}^{m \times m}$, *then* $\|\mathbf{S}\|_2 \leq 20\sqrt{m}$ *except with negligible probability.*

We now recall two versions of the leftover hash lemma [3, 10, 19].

Lemma 4 (Leftover Hash Lemma). *Let* λ *be a security parameter, and let* ℓ, m, q *be lattice parameters. Let* $\mathcal{P}$ *be a distribution over* $\mathbb{Z}^m$ *satisfying* $H_\infty(\mathcal{P}) \geq 2\ell \log q$. *Then,* $(\mathbf{A}, \mathbf{A}\mathbf{r} \bmod q) \approx_s (\mathbf{A}, \mathbf{u})$, *where* $\mathbf{A} \leftarrow \mathbb{Z}_q^{\ell \times m}$, $\mathbf{r} \leftarrow \mathcal{P}$, *and* $\mathbf{u} \leftarrow \mathbb{Z}_q^{\ell}$.

Lemma 5 (Matrix Leftover Hash Lemma). *Suppose* $m > (\ell + 1)\log_2 q + \omega(\log \ell)$ *and* $q > 2$ *is prime. Let* $d = \mathsf{poly}(\lambda)$, $\mathbf{S} \leftarrow \{\pm 1\}^{m \times d}$, $\mathbf{A} \leftarrow \mathbb{Z}_q^{\ell \times m}$, *and* $\mathbf{B} \leftarrow \mathbb{Z}_q^{\ell \times d}$ *be independently uniform. Then, for every fixed row vector* $\mathbf{e} \in \mathbb{Z}_q^{1 \times m}$,

$$(\mathbf{A}, \mathbf{AS}, \mathbf{eS}) \approx_s (\mathbf{A}, \mathbf{B}, \mathbf{eS}).$$

Lemma 6 (Noise Rerandomization [20]). *Let* $\chi > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log d})\}$. *There exists an efficient algorithm* ReRand *such that, for* $\mathbf{v} \in \mathbb{Z}_q^{1 \times m}$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^{1 \times m}, \chi}$, $\mathbf{R} \in \mathbb{Z}^{m \times d}$ *and* $\rho > \|\mathbf{R}\|_2$,

$$\big(\mathbf{R}, \mathbf{x} := \mathbf{v} + \mathbf{e}, \mathsf{ReRand}(\mathbf{R}, \mathbf{x}, \rho)\big) \approx_s \big(\mathbf{R}, \mathbf{x}, \mathbf{vR} + \mathbf{e}'\big),$$

where $\mathbf{e}' \leftarrow \mathcal{D}_{\mathbb{Z}^{1 \times d}, 2\chi\rho}$. *In particular, taking* $\rho = \chi'/(2\chi)$ *gives output noise of Gaussian width* χ', *provided* $\chi'/(2\chi) > \|\mathbf{R}\|_2$.

Lemma 7 (Smudging Lemma). *Let* $B_1, B_2 \in \mathbb{N}$ *and let* $e \in [-B_1, B_1]$ *be fixed. Let* E_1, E_2 *be independently and uniformly distributed over* $[-B_2, B_2]$. *Then,* $\Delta(E_1 + e, E_2) < B_1/B_2$.

Lemma 8 (Gaussian Samples [11]). *Let* ℓ, m, q, σ *be lattice parameters satisfying* $\sigma \geq \log m$, $m \geq 2\ell \log q$, *and let* q *be prime. Then, for all but a* $q^{-\ell}$ *fraction of matrices* $\mathbf{A} \in \mathbb{Z}_q^{\ell \times m}$, *the following holds,*

$$\{(\mathbf{x}, \mathbf{Ax}) : \mathbf{x} \leftarrow D_{\mathbb{Z}^m, \sigma}\} \approx_s \left\{(\mathbf{x}, \mathbf{y}) : \mathbf{y} \leftarrow \mathbb{Z}_q^{\ell},\ \mathbf{x} \leftarrow D_\sigma(\Lambda_q^{\mathbf{y}}(\mathbf{A}))\right\}.$$

Corollary 1 (Output Masking). *Let* $\mathbf{A} \leftarrow \mathbb{Z}_q^{\ell \times m}$, $\mathbf{U} \leftarrow \{0,1\}^{m \times h}$, *and set* $\mathbf{D} := \mathbf{AU}$. *Let* $\mathbf{c} \leftarrow \mathbb{Z}_q^{1 \times m}$ *be uniform and define* $\mathbf{z} := \mathsf{ReRand}\left(\mathbf{U}, \mathbf{c}, \frac{\chi'}{2\chi}\right)$. *Then, it follows that* $(\mathbf{A}, \mathbf{D}, \mathbf{c}, \mathbf{z}) \approx_s (\mathbf{A}, \mathbf{D}, \mathbf{c}, \mathbf{u})$, *where* $\mathbf{u} \leftarrow \mathbb{Z}_q^{1 \times h}$ *is independent uniform.*

3.7 Key-Homomorphic Evaluation

Lemma 9 (Key-Homomorphic Evaluation [4]). *Let $\mathcal{F}_{\mathsf{ckt}} = \{\mathcal{F}_{\mathsf{ckt},\lambda}\}_\lambda$ be a family of polynomial-size arithmetic circuits over $\mathbb{Z}_q$. Let $d = d(\lambda)$ and $r = r(\lambda)$ be upper bounds on circuit depth and gate fan-in, respectively, and let $p = p(\lambda)$ denote an upper bound on at least one input to every multiplication gate. There exist deterministic polynomial-time algorithms $\mathsf{Eval}_{\mathsf{pk}}$, $\mathsf{Eval}_{\mathsf{ct}}$, and $\mathsf{Eval}_{\mathsf{sim}}$ with the following properties:*

- $\mathbf{B}_f \leftarrow \mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{B}_1, \ldots, \mathbf{B}_z)$ *outputs* $\mathbf{B}_f \in \mathbb{Z}_q^{\ell \times m}$.
- *For* $\mathbf{s} \in \mathbb{Z}_q^{1\times \ell}$ *and* $\delta > 0$, *define* $E_{\mathbf{s},\delta}(x, \mathbf{B}) := \{\mathbf{s}(x\mathbf{G} + \mathbf{B}) + \mathbf{e} : \|\mathbf{e}\| \le \delta\}$. *If* $\mathbf{c}_i \in E_{\mathbf{s},\delta}(x_i, \mathbf{B}_i)$ *for all* $i \in [z]$, *then* $\mathbf{c}_f \leftarrow \mathsf{Eval}_{\mathsf{ct}}(f, ((x_i, \mathbf{B}_i, \mathbf{c}_i))_{i=1}^z)$ *belongs to* $E_{\mathbf{s},\Delta}(f(x), \mathbf{B}_f)$, *where* $\mathbf{B}_f = \mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{B}_1, \ldots, \mathbf{B}_z)$ *and* $\Delta < \delta\, \alpha_{\mathcal{F}}(\ell)$, *with* $\alpha_{\mathcal{F}}(\ell) = O((p^{r-1}m)^d\sqrt{m})$.
- *Given* $\mathbf{B}_i = \mathbf{A}\mathbf{S}_i - x_i^*\mathbf{G}$, *the simulation algorithm* $\mathbf{S}_f \leftarrow \mathsf{Eval}_{\mathsf{sim}}(f, ((x_i^*, \mathbf{S}_i))_{i=1}^z, \mathbf{A})$ *satisfies* $\mathbf{A}\mathbf{S}_f - f(x^*)\mathbf{G} = \mathbf{B}_f$, *where* $\mathbf{B}_f = \mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{A}\mathbf{S}_1 - x_1^*\mathbf{G}, \ldots, \mathbf{A}\mathbf{S}_z - x_z^*\mathbf{G})$. *If* $\mathbf{S}_1, \ldots, \mathbf{S}_z \leftarrow \{\pm 1\}^{m\times m}$, *then* $\|\mathbf{S}_f\|_2 < \alpha_{\mathcal{F}}(\ell)$ *except with negligible probability.*

3.8 Matrix Commitments

Lemma 10 (Matrix Commitment [30, 31]). *Let λ be a security parameter, and let ℓ, m, q be lattice parameters. Let $\mathsf{pp} := (\mathbf{A}, \mathbf{W}, \mathbf{T})$, where $\mathbf{A} \in \mathbb{Z}_q^{\ell\times m}$, $\mathbf{W} \in \mathbb{Z}_q^{2m^2\ell\times m}$, and $\mathbf{T} \in \mathbb{Z}^{(2m^2+1)m\times 2m^3}$ satisfy $[\mathbf{I}_{2m^2} \otimes \mathbf{A} \mid \mathbf{W}]\mathbf{T} = \mathbf{I}_{2m^2} \otimes \mathbf{G}$. There exists deterministic efficient algorithms $(\mathsf{Com}, \mathsf{Ver}, \mathsf{Open})$ with the following properties.*

- $\mathbf{C}_{\mathbf{x}} \leftarrow \mathsf{Com}(\mathsf{pp}, \mathbf{x})$, *on input* $\mathbf{x} \in \mathbb{Z}_q^d$, *outputs* $\mathbf{C}_{\mathbf{x}} \in \mathbb{Z}_q^{\ell\times m}$.
- $\mathbf{V} \leftarrow \mathsf{Ver}(\mathsf{pp}, 1^d)$ *outputs* $\mathbf{V} \in \mathbb{Z}_q^{m\times dm}$.
- $\mathbf{Z}_{\mathbf{x}} \leftarrow \mathsf{Open}(\mathsf{pp}, \mathbf{x})$ *outputs* $\mathbf{Z}_{\mathbf{x}} \in \mathbb{Z}_q^{m\times dm}$.

For every $d \in \mathbb{N}$ and every $\mathbf{x} \in \mathbb{Z}_q^d$, these matrices satisfy $\mathbf{C}_{\mathbf{x}}\mathbf{V} = \mathbf{x} \otimes \mathbf{G} - \mathbf{A}\mathbf{Z}_{\mathbf{x}}$. Moreover, $\|\mathbf{V}\|_\infty \le O(\|\mathbf{T}\|_\infty m^4 \log q)$ and $\|\mathbf{Z}_{\mathbf{x}}\|_\infty \le O(\|\mathbf{T}\|_\infty \log d\, m^7 \log q)$.

3.9 Hardness Assumptions

ℓ-Succinct LWE Assumption [30, 31] Let ℓ, m, n, q, χ be lattice and noise parameters with $m \ge 2n \log q$. Sample $\mathbf{A} \leftarrow \mathbb{Z}_q^{n\times m}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^{1\times n}$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times m},\chi}$, $\mathbf{u} \leftarrow \mathbb{Z}_q^{1\times m}$, and $\mathbf{W} \leftarrow \mathbb{Z}_q^{n\ell\times m}$. Let $\mathbf{T}$ be a short preimage satisfying $[\mathbf{I}_\ell \otimes \mathbf{A} \mid \mathbf{W}]\mathbf{T} = \mathbf{I}_\ell \otimes \mathbf{G}$. The ℓ-succinct LWE assumption states that $(\mathbf{A}, \mathbf{s}\mathbf{A} + \mathbf{e}, \mathbf{W}, \mathbf{T}) \approx_c (\mathbf{A}, \mathbf{u}, \mathbf{W}, \mathbf{T})$.

4 Batched Threshold Predicate Encryption

This section defines the notion of batched threshold predicate encryption (BTPE) with one-sided attribute-hiding security.

4.1 System Model of BT-PE

Let λ be a security parameter. A batched threshold predicate encryption scheme is a tuple of efficient algorithms $\prod_{\mathsf{BT\text{-}PE}} = (\mathsf{Setup}, \mathsf{Enc}, \mathsf{Digest}, \mathsf{KeyGen}, \mathsf{PDec}, \mathsf{Combine})$ with message space $\mathcal{M}$, attribute space $\{0,1\}^k$, predicate family $\mathcal{F} = \{\mathcal{F}_\lambda\}_{\lambda\in\mathbb{N}}$ where each $f_\lambda \in \mathcal{F}_\lambda$ is a function $f_\lambda : \{0,1\}^k \to \{0,1\}$, and batch-label space Γ. We adopt the convention that an attribute x satisfies f_λ if $f_\lambda(x) = 0$. We assume throughout that the master public key mpk is implicitly available to every algorithm. Throughout we require $1 \le T \le N$.

$\mathsf{Setup}(1^\lambda, 1^B, 1^T, 1^N, 1^k, 1^h) \to (\mathsf{mpk}, \mathsf{msk})$. On input the security parameter λ, a batch-size bound B, a threshold T,the number N of decryption parties,the attribute length k, and the message length h, the algorithm outputs a master public key mpk and a master secret key msk.To lighten notation, we omit 1^k and 1^h from the input of Setup whenever they are clear from the context.

$\mathsf{Enc}(\mathsf{mpk}, \mu, x, \mathsf{tg}) \to \mathsf{ct}$. On input mpk, a message $\mu \in \mathcal{M}$, an attribute $x \in \{0,1\}^k$, and a batch label $\mathsf{tg} \in \Gamma$, the encryption algorithm outputs a ciphertext ct.

$\mathsf{Digest}(\mathsf{mpk}, S) \to \mathsf{dig}$. On input mpk and a batch $S \subseteq \{0,1\}^k$ with $|S| = B$, the deterministic digest algorithm outputs a digest dig.

$\mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}, f_\lambda, \mathsf{tg}) \to \{\mathsf{sk}_i\}_{i\in[N]}$. On input msk, a digest dig, a predicate $f_\lambda \in \mathcal{F}_\lambda$, and a batch label $\mathsf{tg} \in \Gamma$, the algorithm outputs secret-key shares $\mathsf{sk}_1, \ldots, \mathsf{sk}_N$ jointly bound to $(\mathsf{dig}, f_\lambda, \mathsf{tg})$.

$\mathsf{PDec}(\mathsf{mpk}, \mathsf{sk}_i, S, \mathsf{ct}) \to d_i$ or $\perp$. On input mpk, a secret-key share sk_i, a batch S, and a ciphertext ct, the partial-decryption algorithm outputs a ciphertext-specific partial-decryption token d_i, or $\perp$ on failure.

$\mathsf{Combine}(\mathsf{mpk}, \mathsf{ct}, \{d_i\}_{i\in U}) \to \mu$ or $\perp$. On input mpk, a ciphertext ct, and a collection of partial-decryption tokens $\{d_i\}_{i\in U}$ for $U \subseteq [N]$, the algorithm outputs a message $\mu \in \mathcal{M}$, or $\perp$ on failure.

4.2 Correctness and Efficiency of BT-PE

We require the scheme to satisfy the following correctness and efficiency properties.

1. **Decryption correctness.** For every $\lambda, B, T, N, k \in \mathbb{N}$, every $\mu \in \mathcal{M}$, $\mathsf{tg} \in \Gamma$, every batch $S \subseteq \{0,1\}^k$ with $|S| = B$, every $x \in S$, every $f_\lambda \in \mathcal{F}_\lambda$ satisfying $f_\lambda(x) = 0$, and every $U \subseteq [N]$ with $|U| \ge T$,

$$\Pr\left[\begin{array}{l|l} \mathsf{Combine}\big(\mathsf{mpk}, & (\mathsf{mpk},\mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^B, 1^T, 1^N), \\ \mathsf{ct}, \{d_i\}_{i\in U}\big) = \mu & \mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mu, x, \mathsf{tg}), \\ & \mathsf{dig} \leftarrow \mathsf{Digest}(\mathsf{mpk}, S), \\ & \{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}, f_\lambda, \mathsf{tg}), \\ & d_i \leftarrow \mathsf{PDec}(\mathsf{mpk}, \mathsf{sk}_i, S, \mathsf{ct}) \quad \forall i \in U \end{array}\right] \ge 1 - \mathsf{negl}(\lambda).$$

   Moreover, if $x \notin S$ or $f_\lambda(x) \neq 0$, then the decryption procedure outputs $\perp$ except with negligible probability.

2. **Succinctness.** There exists a universal polynomial $\mathsf{poly}(\cdot)$ such that, for all admissible parameters, the sizes of every secret-key share sk_i and the digest dig are bounded by $\mathsf{poly}(\lambda, |f_\lambda|, \log N, \log B)$.

4.3 Query-Selective One-Sided Security of BT-PE

For a security parameter λ and a bit $b \in \{0,1\}$, define the query-selective one-sided security experiment $\mathsf{Exp}^{\mathsf{bt\text{-}pe}}_{\mathcal{A}}(\lambda, b)$ between a challenger and an adversary $\mathcal{A}$ as follows. At the beginning of the experiment, $\mathcal{A}(1^\lambda)$ outputs a batch-size bound B, a threshold T, a committee size N, an attribute length k, a message space $\mathcal{M}$, two challenge attributes $x_0^*, x_1^* \in \{0,1\}^k$, and a challenge batch label $\mathsf{tg}^* \in \Gamma$.

For a tuple $(S, f_\lambda, \mathsf{tg})$, define

$$\mathsf{Comp}_b(S, f_\lambda, \mathsf{tg}) := \big[\mathsf{tg} = \mathsf{tg}^* \wedge x_b^* \in S \wedge f_\lambda(x_b^*) = 0\big], \qquad b \in \{0,1\}.$$

We call a tuple *jointly challenge compatible* if $\mathsf{Comp}_0 = \mathsf{Comp}_1 = 1$, and *challenge incompatible* if $\mathsf{Comp}_0 = \mathsf{Comp}_1 = 0$. A tuple satisfying exactly one of $\mathsf{Comp}_0, \mathsf{Comp}_1$ is inadmissible.

- **Setup:** The challenger first computes $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^B, 1^T, 1^N)$ and gives mpk to $\mathcal{A}$. It initializes an empty dictionary Keys indexed by triples $(S, f_\lambda, \mathsf{tg})$.
- **Query declaration:** Before making any oracle queries, $\mathcal{A}$ declares a set $\mathcal{Q}^* \subseteq [N] \times \{S \subseteq \{0,1\}^k : |S| = B\} \times \mathcal{F}_\lambda$ subject to the following two requirements. First, every $(i, S, f) \in \mathcal{Q}^*$ satisfies $x_0^*, x_1^* \in S$ and $f(x_0^*) = f(x_1^*) = 0$. Second, for every pair (S, f), $\big|\{i \in [N] : (i, S, f) \in \mathcal{Q}^*\}\big| \le T - 1$, so that at most $T-1$ parties are declared for any single key instance. These are the only jointly challenge-compatible tuples for which $\mathcal{A}$ may obtain either a secret-key share or a partial-decryption token, irrespective of the ciphertext supplied to $\mathcal{O}_{\mathsf{PDec}}$.
- $\mathcal{O}_{\mathsf{KeyGen}}(i, S, f_\lambda, \mathsf{tg})$**:** On input a party $i \in [N]$, a batch $S \subseteq \{0,1\}^k$ with $|S| = B$, a predicate $f_\lambda \in \mathcal{F}_\lambda$, and a label $\mathsf{tg} \in \Gamma$, proceed as follows.
  - If $\mathsf{Comp}_0(S, f_\lambda, \mathsf{tg}) \neq \mathsf{Comp}_1(S, f_\lambda, \mathsf{tg})$, return $\perp$.
  - If $\mathsf{Comp}_0(S, f_\lambda, \mathsf{tg}) = \mathsf{Comp}_1(S, f_\lambda, \mathsf{tg}) = 1$, require $(i, S, f_\lambda) \in \mathcal{Q}^*$; otherwise, return $\perp$.
  - If $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})] = \emptyset$, compute $\mathsf{dig} \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$ and $\{\mathsf{sk}_j\}_{j\in[N]} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}, f_\lambda, \mathsf{tg})$, and store $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})] := \{\mathsf{sk}_j\}_{j\in[N]}$.
  - Return the i-th share in $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})]$.
- $\mathcal{O}_{\mathsf{PDec}}(i, S, f_\lambda, \mathsf{tg}, \mathsf{ct})$**:** On input a party $i \in [N]$, a batch $S \subseteq \{0,1\}^k$ with $|S| = B$, a predicate $f_\lambda \in \mathcal{F}_\lambda$, a label $\mathsf{tg} \in \Gamma$, and a ciphertext ct, proceed as follows.
  - If $\mathsf{Comp}_0(S, f_\lambda, \mathsf{tg}) \neq \mathsf{Comp}_1(S, f_\lambda, \mathsf{tg})$, return $\perp$.
  - If $\mathsf{Comp}_0(S, f_\lambda, \mathsf{tg}) = \mathsf{Comp}_1(S, f_\lambda, \mathsf{tg}) = 1$, require $(i, S, f_\lambda) \in \mathcal{Q}^*$; otherwise return $\perp$. This restriction applies to every partial-decryption query, independently of whether ct is the challenge ciphertext.

- If $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})] = \emptyset$, compute $\mathsf{dig} \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$ and $\{\mathsf{sk}_j\}_{j\in[N]} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}, f_\lambda, \mathsf{tg})$, and store $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})] := \{\mathsf{sk}_j\}_{j\in[N]}$.
- Let sk_i be the i-th share in $\mathsf{Keys}[(S, f_\lambda, \mathsf{tg})]$ and return $\mathsf{PDec}(\mathsf{mpk}, \mathsf{sk}_i, S, \mathsf{ct})$.

- **Challenge phase:** This phase occurs at most once. The adversary submits two messages $\mu_0, \mu_1 \in \mathcal{M}$. The challenger computes $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{mpk}, \mu_b, x_b^*, \mathsf{tg}^*)$ and returns ct^* to $\mathcal{A}$. The adversary may continue to access the above oracles subject to the same restrictions.
- **Output phase:** The adversary finally outputs a bit $b' \in \{0, 1\}$, which is taken as the output of the experiment.

We say that $\prod_{\mathsf{BT\text{-}PE}}$ is *query-selectively one-sided secure* if, for every admissible QPT adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that

$$\mathsf{Adv}_{\mathcal{A}}^{\mathsf{bt\text{-}pe}}(\lambda) := \left| \Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathsf{bt\text{-}pe}}(\lambda, 0) = 1] - \Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathsf{bt\text{-}pe}}(\lambda, 1) = 1] \right| \leq \mathsf{negl}(\lambda).$$

5 Lattice-Based Construction of BT-PE

In this section, we give our lattice-based construction of one-sided batched threshold predicate encryption. Let λ be the security parameter, B be the batch size, N be the number of decryption parties, and $1 \leq T \leq N$ be the reconstruction threshold. Let k be the attribute length, h be the message length, and $\ell_{\mathsf{lock}} \geq \lambda$ be the lock length. We define the following building blocks.

- Let $\ell, m, q, \chi, \chi', \sigma_0, \sigma$ be the lattice and Gaussian parameters, where q is an odd prime with $q \geq 2^k$, and let $\kappa := \lceil \log q \rceil$. We further require $m = \Theta(\ell \log q)$ and $m > (\ell+1)\log_2 q + \omega(\log \ell)$, the latter being the hypothesis of the matrix leftover-hash lemma (Lemma 5), which is invoked in the security proof.
- Let $\mathcal{X} := \{0,1\}^k$ be the attribute space, $\mathcal{M} := \{0,1\}^h$ be the message space, and $\Gamma := \mathbb{Z}_q$ be the batch-label space.
- Define the canonical embedding $\iota(x) := \sum_{a=1}^{k} x_a 2^{a-1} \in \mathbb{Z}_q$ for every $x = (x_1, \ldots, x_k) \in \{0,1\}^k$. Since $q \geq 2^k$, the map ι is injective on $\{0,1\}^k$.
- We use the Shamir secret-sharing scheme of Appendix 3.3, the digital signature scheme of Appendix 3.2, and the compute-and-compare obfuscator of Appendix 3.1.
- Define $B_{\mathbf{R}} := 2\sigma\sqrt{m\ell_{\mathsf{lock}}}$. Choose the prime Q_{SS} such that $Q_{\mathsf{SS}} > N$ and $Q_{\mathsf{SS}} > 2\lceil B_{\mathbf{R}} \rceil + 1$. Thus, every admissible short matrix $\mathbf{R} \in \mathbb{Z}^{4m \times \ell_{\mathsf{lock}}}$ admits an injective centered encoding over $\mathbb{F}_{\mathsf{SS}}$.
- Choose the Gaussian widths so that every invocation of $\mathsf{SamplePre}$ satisfies Lemma 1. Let τ_{setup} and τ_{key} be deterministic polynomial upper bounds on the Gram-Schmidt norms of the setup and key-generation trapdoors, respectively. We require

  $$\sigma_0 \geq \tau_{\mathsf{setup}}\omega(\sqrt{\log((2m^2+1)m)}) \quad \text{and} \quad \sigma \geq \max\{\tau_{\mathsf{key}}, \tau_{\mathsf{sim}}\}\,\omega(\sqrt{\log(4m)}),$$

  where τ_{sim} is the bound on the alternative trapdoors of the security proof defined below.

- For the matrix-commitment trapdoor, define $M_{\mathbf{T}} := (2m^2+1)m$, $d_{\mathbf{T}} := 2m^3$, and $T_{\max} := \sigma_0\sqrt{M_{\mathbf{T}} d_{\mathbf{T}}}$. By Lemma 3, $\|\mathbf{T}\|_\infty \le \|\mathbf{T}\|_2 \le T_{\max}$ except with negligible probability.
- Choose the noise widths so that $\chi > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log \ell_{\mathsf{lock}}})\}$ and $\chi'/(2\chi) > \sqrt{m \max\{m, \ell_{\mathsf{lock}}\}}$. Since $\|\mathbf{U}\|_2 \le \|\mathbf{U}\|_F \le \sqrt{md}$ for every $\mathbf{U} \in \{0,1\}^{m\times d}$, this guarantees $\chi'/(2\chi) > \|\mathbf{U}\|_2$ for the binary programming matrices $\mathbf{U}_{\mathsf{bat}}, \mathbf{U}_{\mathsf{tg}} \in \{0,1\}^{m\times m}$ and $\mathbf{U}_{\mathbf{D}} \in \{0,1\}^{m\times \ell_{\mathsf{lock}}}$ of the security proof, as required by Lemma 6 in the proof of Lemma 15.
- Let $d_B := \kappa(B+1)$, let $V_{\max}$ and $Z_{\max}$ be the deterministic bounds on $\|\mathbf{V}\|_\infty$ and $\|\mathbf{Z}_x\|_\infty$ given by Lemma 10 for $\|\mathbf{T}\|_\infty \le T_{\max}$, and let $\alpha_{\mathcal{F}}$ be as in Lemma 9. Let $\tau_{\mathbf{L}}$ be a deterministic polynomial upper bound on $\|\widetilde{\mathbf{T}}_{\mathbf{L}_1}\|$ for $(\mathbf{L}_1, \mathbf{T}_{\mathbf{L}_1}) \leftarrow \mathsf{TrapGen}(1^\ell, 1^m, q)$. Define $\rho_{\mathsf{bat}} := m\, d_B\, (mV_{\max} + Z_{\max})$, $\rho_{\mathsf{tg}} := m$, and

$$\tau_{\mathsf{sim}} := \max\big\{\|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1+\rho_{\mathsf{bat}}),\ \tau_{\mathbf{L}}(1+\rho_{\mathsf{tg}}),\ \|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1+\alpha_{\mathcal{F}})\big\}.$$

  Here ρ_{bat} bounds $\|\mathbf{R}_S\|_2$ in Lemma 16, because

$$\begin{aligned}\|\mathbf{R}_S\|_2 &\le (\|\mathbf{U}_{\mathsf{bat}}\|_2\|\mathbf{V}\|_2 + \|\mathbf{Z}_b^*\|_2)\|\mathbf{Y}_S\|_2 \\ &\le (m\cdot m\sqrt{d_B}\,V_{\max} + m\sqrt{d_B}\,Z_{\max})\sqrt{d_B} = \rho_{\mathsf{bat}}.\end{aligned}$$

  Similarly, ρ_{tg} bounds $\|(\mathbf{U}_{\mathsf{tg}}; \mathbf{0})\|_2$ in Lemma 17, and $\alpha_{\mathcal{F}}$ bounds $\|\mathbf{S}_f\|_2$ in Lemma 18.

The algorithms of BT-PE are as follows.

$\mathsf{Setup}(1^\lambda, 1^B, 1^T, 1^N, 1^k, 1^h)$:

- Compute $(\mathbf{A}, \mathbf{T}_{\mathbf{A}}) \leftarrow \mathsf{TrapGen}(1^\ell, 1^m, q)$.
- Sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{2m^2\ell\times m}$ and compute $\mathbf{T}' \leftarrow \mathsf{ExtendRight}\big(\mathbf{I}_{2m^2}\otimes\mathbf{A},\ \mathbf{I}_{2m^2}\otimes \mathbf{T}_{\mathbf{A}},\ \mathbf{W}\big)$.
- Sample $\mathbf{T} \leftarrow \mathsf{SamplePre}\big([\mathbf{I}_{2m^2}\otimes\mathbf{A} \mid \mathbf{W}],\ \mathbf{T}',\ \mathbf{I}_{2m^2}\otimes\mathbf{G},\ \sigma_0\big)$.
- Set $\mathsf{pp}_{\mathsf{com}} := (\mathbf{A}, \mathbf{W}, \mathbf{T})$.
- Sample independently $\mathbf{Q}, \mathbf{L}_0, \mathbf{L}_1, \mathbf{B}_1, \ldots, \mathbf{B}_k \leftarrow \mathbb{Z}_q^{\ell\times m}$ and $\mathbf{D} \leftarrow \mathbb{Z}_q^{\ell\times \ell_{\mathsf{lock}}}$.
- Generate a root certification key pair $(\mathsf{vk}_{\mathsf{root}}, \mathsf{ssk}_{\mathsf{root}}) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$.
- Output

$$\begin{aligned}\mathsf{mpk} &:= (\mathsf{pp}_{\mathsf{com}}, \mathbf{Q}, \mathbf{L}_0, \mathbf{L}_1, \mathbf{B}_1, \ldots, \mathbf{B}_k, \mathbf{D}, \mathsf{vk}_{\mathsf{root}}, B, T, N, Q_{\mathsf{SS}}),\\ \mathsf{msk} &:= (\mathbf{T}_{\mathbf{A}}, \mathsf{ssk}_{\mathsf{root}}).\end{aligned}$$

$\mathsf{Digest}(\mathsf{mpk}, S)$:

- For $S = \{x_1, \ldots, x_B\} \subseteq \{0,1\}^k$, compute $\gamma \leftarrow \mathsf{GenCoeff}(\iota(S))$ and $\mathbf{V} \leftarrow \mathsf{Ver}(\mathsf{pp}_{\mathsf{com}}, 1^{\kappa(B+1)})$.
- Set $\mathbf{Y}_S := \gamma^\top \otimes \mathbf{I}_m$ and output $\mathsf{dig}_S := -\mathbf{QVY}_S \in \mathbb{Z}_q^{\ell\times m}$.

Decryption Circuit $\mathcal{D}$

Hardwired: mpk, x, tg, cid^*, and $\mathsf{ct}^*_{\mathsf{inner}}$.

Input: cid, $\mathsf{ct}_{\mathsf{inner}}$, and $\{d_i\}_{i\in U_0}$, where $|U_0| = T$.

1. If $\mathsf{cid} \neq \mathsf{cid}^*$ or $\mathsf{ct}_{\mathsf{inner}} \neq \mathsf{ct}^*_{\mathsf{inner}}$, output $0^{\ell_{\mathsf{lock}}}$.
2. Parse $d_i = (i, S_i, \mathsf{kid}_i, f_i, \mathsf{tg}_i, \mathsf{dig}_i, \mathsf{sh}_i, \mathsf{tvk}_i, \mathsf{cert}_i, \tau_i)$ for every $i \in U_0$. If the indices are not pairwise distinct, output $0^{\ell_{\mathsf{lock}}}$.
3. If the batches $\{S_i\}_{i\in U_0}$ are not identical, output $0^{\ell_{\mathsf{lock}}}$. Let S denote their common value. If $|S| \neq B$ or $x \notin S$, output $0^{\ell_{\mathsf{lock}}}$.
4. Compute $\mathsf{dig}_S \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$. Choose any $i_0 \in U_0$ and set $\mathsf{kid} := \mathsf{kid}_{i_0}$ and $f := f_{i_0}$. For every $i \in U_0$, verify $\mathsf{kid}_i = \mathsf{kid}$, $f_i = f$, $\mathsf{tg}_i = \mathsf{tg}$, and $\mathsf{dig}_i = \mathsf{dig}_S$. If any check fails, output $0^{\ell_{\mathsf{lock}}}$.
5. For every $i \in U_0$, set $\mathsf{auth}_i := (i, \mathsf{kid}, f, \mathsf{tg}, \mathsf{dig}_S, \mathsf{sh}_i, \mathsf{tvk}_i)$ and verify $\mathsf{SIG.Ver}(\mathsf{vk}_{\mathsf{root}}, \mathsf{auth}_i, \mathsf{cert}_i) = 1$. If any verification fails, output $0^{\ell_{\mathsf{lock}}}$.
6. For every $i \in U_0$, set $\mathsf{tok}_i := (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, i, S, \mathsf{kid}, f, \mathsf{tg}, \mathsf{dig}_S, \mathsf{sh}_i)$ and verify $\mathsf{SIG.Ver}(\mathsf{tvk}_i, \mathsf{tok}_i, \tau_i) = 1$. If any verification fails, output $0^{\ell_{\mathsf{lock}}}$.
7. If $f(x) \neq 0$, output $0^{\ell_{\mathsf{lock}}}$.
8. Reconstruct $\widehat{\mathbf{R}} \leftarrow \mathsf{SS.Reconst}_{T,N}(\{(i, \mathsf{sh}_i)\}_{i\in U_0})$ and compute $\mathbf{R} \leftarrow \mathsf{Decode}_{\mathsf{SS}}(\widehat{\mathbf{R}})$. If decoding fails or $\|\mathbf{R}\|_2 > B_{\mathbf{R}}$, output $0^{\ell_{\mathsf{lock}}}$.
9. Compute $\mathbf{B}_f \leftarrow \mathsf{Eval}_{\mathsf{pk}}(f, \mathbf{B}_1, \ldots, \mathbf{B}_k)$ and set $\mathbf{F} := [\mathbf{A} \mid \mathsf{dig}_S \mid \mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 \mid \mathbf{B}_f]$. If $\mathbf{FR} \neq \mathbf{D} \bmod q$, output $0^{\ell_{\mathsf{lock}}}$.
10. Parse $\mathsf{ct}_{\mathsf{inner}} = (\mathbf{c}_{\mathsf{in}}, \mathbf{c}_1, \ldots, \mathbf{c}_k, \mathbf{c}_{\mathsf{bat}}, \mathbf{c}_{\mathsf{tg}}, \mathbf{c}_{\mathsf{out}})$. Compute $\gamma \leftarrow \mathsf{GenCoeff}(\iota(S))$, $\mathbf{Y}_S := \gamma^\top \otimes \mathbf{I}_m$, $\mathbf{V} \leftarrow \mathsf{Ver}(\mathsf{pp}_{\mathsf{com}}, 1^{\kappa(B+1)})$, and $\mathbf{Z}_x \leftarrow \mathsf{Open}(\mathsf{pp}_{\mathsf{com}}, \mathsf{ID}(x))$. Set $\mathbf{M}_x := \left(\begin{smallmatrix} -\mathbf{Z}_x \\ -\mathbf{V} \end{smallmatrix}\right)\mathbf{Y}_S$ and $\mathbf{c}_{\mathsf{dig}} := [\mathbf{c}_{\mathsf{in}} \mid \mathbf{c}_{\mathsf{bat}}]\mathbf{M}_x$.
11. Compute $\mathbf{c}_f \leftarrow \mathsf{Eval}_{\mathsf{ct}}(f, \{(x_a, \mathbf{B}_a, \mathbf{c}_a)\}_{a=1}^k)$ and set $\overline{\mathbf{c}} := [\mathbf{c}_{\mathsf{in}} \mid \mathbf{c}_{\mathsf{dig}} \mid \mathbf{c}_{\mathsf{tg}} \mid \mathbf{c}_f]$.
12. Compute $\mathbf{v} := \mathbf{c}_{\mathsf{out}} - \overline{\mathbf{c}}\mathbf{R} \bmod q$ and $\varsigma' := \lfloor (2/q)\mathbf{v} \rceil \bmod 2 \in \{0,1\}^{\ell_{\mathsf{lock}}}$.
13. Output ς'.

Fig. 1: Decryption circuit $\mathcal{D}$ for $\prod_{\mathsf{BT\text{-}PE}}$.

$\mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}, f_\lambda, \mathsf{tg})$:

- Compute $\mathbf{B}_{f_\lambda} \leftarrow \mathsf{Eval}_{\mathsf{pk}}(f_\lambda, \mathbf{B}_1, \ldots, \mathbf{B}_k)$.
- Define $\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}} := [\mathbf{A} \mid \mathsf{dig} \mid \mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 \mid \mathbf{B}_{f_\lambda}] \in \mathbb{Z}_q^{\ell\times 4m}$.
- Using $\mathbf{T}_{\mathbf{A}}$ and repeated applications of $\mathsf{ExtendRight}$, derive a trapdoor $\mathbf{T}_{\mathsf{dig},f_\lambda,\mathsf{tg}}$ for $\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}}$.
- Sample $\mathbf{R} \leftarrow \mathsf{SamplePre}\big(\mathbf{F}_{\mathsf{dig},f_\lambda,\mathsf{tg}}, \mathbf{T}_{\mathsf{dig},f_\lambda,\mathsf{tg}}, \mathbf{D}, \sigma\big) \in \mathbb{Z}^{4m\times\ell_{\mathsf{lock}}}$ and $\mathsf{kid} \leftarrow \{0,1\}^\lambda$.
- If $\|\mathbf{R}\|_2 > B_{\mathbf{R}}$, repeat the sampling step.

- Compute $\widehat{\mathbf{R}} := \mathsf{Encode}_{\mathsf{SS}}(\mathbf{R})$ and $(\mathsf{sh}_1, \ldots, \mathsf{sh}_N) \leftarrow \mathsf{SS.Share}_{T,N}(\widehat{\mathbf{R}})$.
- For every $i \in [N]$, generate $(\mathsf{tvk}_i, \mathsf{tsk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$. Set $\mathsf{auth}_i := (i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i)$ and compute $\mathsf{cert}_i \leftarrow \mathsf{SIG.Sign}(\mathsf{ssk}_{\mathsf{root}}, \mathsf{auth}_i)$.
- Output $\{\mathsf{sk}_i\}_{i\in[N]}$, where $\mathsf{sk}_i := (i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i, \mathsf{tsk}_i, \mathsf{cert}_i)$.

$\mathsf{Enc}(\mathsf{mpk}, \mu, x, \mathsf{tg})$:

- Parse $x = (x_1, \ldots, x_k)$. Sample $\mathsf{cid} \leftarrow \{0,1\}^{2\lambda}$, $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$, $\mathbf{s} \leftarrow \mathbb{Z}_q^{1\times\ell}$, $\mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times m},\chi}$, and $\mathbf{S}_1, \ldots, \mathbf{S}_k \leftarrow \{\pm 1\}^{m\times m}$.
- Set $\mathbf{c}_{\mathsf{in}} := \mathbf{s}\mathbf{A} + \mathbf{e}_0$ and, for every $a \in [k]$, $\mathbf{c}_a := \mathbf{s}(x_a\mathbf{G} + \mathbf{B}_a) + \mathbf{e}_0\mathbf{S}_a$.
- Define $\mathsf{ID}(x) := (1, \iota(x), \ldots, \iota(x)^B) \otimes \mathbf{g}'$ and compute $\mathbf{C}_{\mathsf{ID}} \leftarrow \mathsf{Com}(\mathsf{pp}_{\mathsf{com}}, \mathsf{ID}(x))$.
- Sample $\mathbf{e}_{\mathsf{bat}}, \mathbf{e}_{\mathsf{tg}} \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times m},\chi'}$ and $\mathbf{e}_{\mathsf{out}} \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times \ell_{\mathsf{lock}}},\chi'}$. Set

$$\begin{aligned}
\mathbf{c}_{\mathsf{bat}} &:= \mathbf{s}(\mathbf{Q} + \mathbf{C}_{\mathsf{ID}}) + \mathbf{e}_{\mathsf{bat}},\\
\mathbf{c}_{\mathsf{tg}} &:= \mathbf{s}(\mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1) + \mathbf{e}_{\mathsf{tg}},\\
\mathbf{c}_{\mathsf{out}} &:= \mathbf{s}\mathbf{D} + \mathbf{e}_{\mathsf{out}} + \left\lfloor \frac{q}{2} \right\rfloor \varsigma.
\end{aligned}$$

- Set $\mathsf{ct}_{\mathsf{inner}} := (\mathbf{c}_{\mathsf{in}}, \mathbf{c}_1, \ldots, \mathbf{c}_k, \mathbf{c}_{\mathsf{bat}}, \mathbf{c}_{\mathsf{tg}}, \mathbf{c}_{\mathsf{out}})$.
- Compute $\widetilde{Q} \leftarrow \mathsf{CC.Obf}(1^\lambda, \mathcal{D}, \mu, \varsigma)$, where $\mathcal{D}$ is the decryption circuit of Figure 1, with mpk, x, tg, cid, and $\mathsf{ct}_{\mathsf{inner}}$ hardwired. Let $s_{\mathcal{F}}$ denote an upper bound on the size of the predicates in $\mathcal{F}_\lambda$. Since $\mathcal{D}$ reads the predicate f from the partial-decryption tokens, it evaluates f using a universal circuit for predicates of size at most $s_{\mathcal{F}}$; hence, the size of $\mathcal{D}$ depends only on $s_{\mathcal{F}}$ and not on the particular predicate.
- Output $\mathsf{ct} := (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$.

$\mathsf{PDec}(\mathsf{mpk}, \mathsf{sk}_i, S, \mathsf{ct})$:

- Parse $\mathsf{sk}_i = (i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i, \mathsf{tsk}_i, \mathsf{cert}_i)$ and $\mathsf{ct} = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$. If parsing fails, output $\perp$.
- Compute $\mathsf{dig}_S \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$. If $\mathsf{dig}_S \neq \mathsf{dig}$, output $\perp$.
- Set $\mathsf{auth}_i := (i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i)$. If $\mathsf{SIG.Ver}(\mathsf{vk}_{\mathsf{root}}, \mathsf{auth}_i, \mathsf{cert}_i) \neq 1$, output $\perp$.
- Check that $\mathsf{cid} \in \{0,1\}^{2\lambda}$ and that $\mathsf{ct}_{\mathsf{inner}}$ has the prescribed syntax and dimensions. If not, output $\perp$.
- Define $\mathsf{tok}_i := (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, i, S, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i)$ and compute $\tau_i \leftarrow \mathsf{SIG.Sign}(\mathsf{tsk}_i, \mathsf{tok}_i)$.
- Output $d_i := (i, S, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i, \mathsf{tvk}_i, \mathsf{cert}_i, \tau_i)$.

$\mathsf{Combine}(\mathsf{mpk}, \mathsf{ct}, \{d_i\}_{i\in U})$:

- If $|U| < T$, output $\perp$.
- Parse $\mathsf{ct} = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$. If parsing fails, output $\perp$.
- Choose any subset $U_0 \subseteq U$ of size exactly T. If the corresponding party indices are not pairwise distinct, output $\perp$.
- Output $\mathsf{CC.Eval}\left(\widetilde{Q}, \left(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \{d_i\}_{i\in U_0}\right)\right) \in \mathcal{M} \cup \{\perp\}$.

6 Analysis of BT-PE Construction

Theorem 2 (Correctness). *Assume that $B, N, k, h, \ell_{\mathsf{lock}}$ and the size of every predicate circuit $f_\lambda \in \mathcal{F}_\lambda$ are polynomially bounded in λ. Assume further that $q \geq 2^k$, Gaussian parameters σ_0 and σ satisfy the* SamplePre *conditions specified above, and*

$$q > 4\left(c_{\mathsf{G}}\chi'\sqrt{\ell_{\mathsf{lock}}} + 2\sigma\sqrt{m\ell_{\mathsf{lock}}}\,\Delta_{\mathsf{ct}}\right) + 2, \tag{3}$$

where Δ_{ct} is the deterministic bound defined below. Then, the construction of Section 5 satisfies the decryption correctness property of Section 4.2.

Proof. Fix $\lambda, B, T, N, k, h \in \mathbb{N}$, a message $\mu \in \{0,1\}^h$, a batch label $\mathsf{tg} \in \Gamma$, a batch $S \subseteq \{0,1\}^k$ with $|S| = B$, an attribute $x \in S$, a predicate $f_\lambda \in \mathcal{F}_\lambda$ satisfying $f_\lambda(x) = 0$, and a set $U \subseteq [N]$ with $|U| \geq T$. Let $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}(1^\lambda, 1^B, 1^T, 1^N)$, $\mathsf{dig}_S \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$, and $\{\mathsf{sk}_i\}_{i\in[N]} \leftarrow \mathsf{KeyGen}(\mathsf{msk}, \mathsf{dig}_S, f_\lambda, \mathsf{tg})$. Encryption samples $\mathsf{cid} \leftarrow \{0,1\}^{2\lambda}$ and a uniform lock $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$, constructs $\mathsf{ct}_{\mathsf{inner}}$, computes $\widetilde{Q} \leftarrow \mathsf{CC.Obf}(1^\lambda, \mathcal{D}, \mu, \varsigma)$, and outputs $\mathsf{ct} = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$. For every $i \in U$, let $d_i \leftarrow \mathsf{PDec}(\mathsf{mpk}, \mathsf{sk}_i, S, \mathsf{ct})$.

All secret-key shares produced in the same invocation of KeyGen contain the common tuple $(\mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}_S)$. For every $i \in [N]$, key generation also produces a token-verification key tvk_i and a root certificate cert_i on $\mathsf{auth}_i = (i, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}_S, \mathsf{sh}_i, \mathsf{tvk}_i)$. By perfect correctness of the digital signature scheme, all root-certificate checks performed by PDec and $\mathcal{D}$ succeed.

Moreover, PDec signs the ciphertext-specific message $\mathsf{tok}_i = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, i, S, \mathsf{kid}, f_\lambda, \mathsf{tg}, \mathsf{dig}_S, \mathsf{sh}_i)$ using tsk_i. Therefore, the token signature τ_i verifies under tvk_i inside $\mathcal{D}$. Since the circuit is hardwired with the same $(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}})$ generated during encryption, its initial ciphertext-binding check also succeeds. The remaining consistency checks $|S| = B$, $x \in S$, $\mathsf{dig}_i = \mathsf{dig}_S$, $\mathsf{tg}_i = \mathsf{tg}$, and $f_\lambda(x) = 0$ succeed as well.

The algorithm Combine chooses a subset $U_0 \subseteq U$ with $|U_0| = T$. By perfect correctness of Shamir sharing (Appendix 3.3), the circuit reconstructs

$$\widehat{\mathbf{R}} = \mathsf{SS.Reconst}_{T,N}\big(\{(i, \mathsf{sh}_i)\}_{i\in U_0}\big) = \mathsf{Encode}_{\mathsf{SS}}(\mathbf{R}).$$

Since $\|\mathbf{R}\|_2 \leq B_{\mathbf{R}}$ by key generation and $Q_{\mathsf{SS}} > 2\lceil B_{\mathbf{R}}\rceil + 1$, every entry of $\mathbf{R}$ lies in the injective range of the centered encoding. Hence, $\mathsf{Decode}_{\mathsf{SS}}(\widehat{\mathbf{R}}) = \mathbf{R}$ exactly. Furthermore,

$$\mathbf{F}_{\mathsf{dig}_S, f_\lambda, \mathsf{tg}}\mathbf{R} = \mathbf{D} \pmod q, \tag{4}$$

so the corresponding consistency check in $\mathcal{D}$ succeeds.

We next analyze the lattice decryption performed using the reconstructed matrix $\mathbf{R}$. Let $\gamma \leftarrow \mathsf{GenCoeff}(\iota(S))$ and $\mathbf{Y}_S := \gamma^\top \otimes \mathbf{I}_m$. Thus, $\mathsf{ID}(x)\gamma^\top = P_{\iota(S)}(\iota(x))$. Since $x \in S$ and ι is injective, $\iota(x) \in \iota(S)$, and therefore, $\mathsf{ID}(x)\gamma^\top = 0$. Recall

that $\mathbf{c}_{\mathsf{in}} = \mathbf{s}\mathbf{A} + \mathbf{e}_0$ and $\mathbf{c}_{\mathsf{bat}} = \mathbf{s}(\mathbf{Q} + \mathbf{C}_{\mathsf{ID}}) + \mathbf{e}_{\mathsf{bat}}$. The circuit computes

$$\begin{aligned}\mathbf{c}_{\mathsf{dig}} &= [\mathbf{c}_{\mathsf{in}} \mid \mathbf{c}_{\mathsf{bat}}] \begin{pmatrix} -\mathbf{Z}_x \\ -\mathbf{V} \end{pmatrix} \mathbf{Y}_S \\ &= \mathbf{s}\big(-\mathbf{A}\mathbf{Z}_x - (\mathbf{Q} + \mathbf{C}_{\mathsf{ID}})\mathbf{V}\big)\mathbf{Y}_S + \mathbf{e}_{\mathsf{dig}}, \end{aligned} \tag{5}$$

where $\mathbf{e}_{\mathsf{dig}} := [\mathbf{e}_0 \mid \mathbf{e}_{\mathsf{bat}}]\left(\begin{smallmatrix} -\mathbf{Z}_x \\ -\mathbf{V} \end{smallmatrix}\right)\mathbf{Y}_S$.
By Lemma 10, $\mathbf{C}_{\mathsf{ID}}\mathbf{V} = \mathsf{ID}(x) \otimes \mathbf{G} - \mathbf{A}\mathbf{Z}_x$. Using $(\mathsf{ID}(x) \otimes \mathbf{G})(\gamma^\top \otimes \mathbf{I}_m) = (\mathsf{ID}(x)\gamma^\top)\mathbf{G}$, Eq. (5) gives

$$\mathbf{c}_{\mathsf{dig}} = \mathbf{s}\,\mathsf{dig}_S - \mathbf{s}\big(\mathsf{ID}(x)\gamma^\top\big)\mathbf{G} + \mathbf{e}_{\mathsf{dig}} = \mathbf{s}\,\mathsf{dig}_S + \mathbf{e}_{\mathsf{dig}}. \tag{6}$$

We record a deterministic upper bound on $\mathbf{e}_{\mathsf{dig}}$. Let $d_B := \kappa(B+1)$. Recall the deterministic quantity $T_{\max}$ defined above. Except with negligible probability, $\|\mathbf{T}\|_\infty \le T_{\max}$.

Let $V_{\max}$ and $Z_{\max}$ be deterministic bounds satisfying $\|\mathbf{V}\|_\infty \le V_{\max} = O(T_{\max} m^4 \log q)$ and $\|\mathbf{Z}_x\|_\infty \le Z_{\max} = O(T_{\max} m^7 \log q \log(d_B m))$, as guaranteed by Lemma 10. Moreover, $\|\mathbf{V}\|_2 \le m\sqrt{d_B}\, V_{\max}$, $\|\mathbf{Z}_x\|_2 \le m\sqrt{d_B}\, Z_{\max}$, and $\|\mathbf{Y}_S\|_2 = \|\gamma\|_2 \le \sqrt{d_B}$. By Lemma 2, except with negligible probability, $\|\mathbf{e}_0\|_2 \le c_{\mathsf{G}}\chi\sqrt{m}$ and $\|\mathbf{e}_{\mathsf{bat}}\|_2 \le c_{\mathsf{G}}\chi'\sqrt{m}$. Consequently,

$$\begin{aligned}\|\mathbf{e}_{\mathsf{dig}}\|_2 &\le \big(\|\mathbf{e}_0\|_2\|\mathbf{Z}_x\|_2 + \|\mathbf{e}_{\mathsf{bat}}\|_2\|\mathbf{V}\|_2\big)\|\mathbf{Y}_S\|_2 \\ &\le c_{\mathsf{G}} m^{3/2} d_B\big(\chi Z_{\max} + \chi' V_{\max}\big) =: \Delta_{\mathsf{dig}}. \end{aligned} \tag{7}$$

For every $a \in [k]$, the inner ciphertext contains $\mathbf{c}_a = \mathbf{s}(x_a\mathbf{G} + \mathbf{B}_a) + \mathbf{e}_0\mathbf{S}_a$. By Lemmas 2 and 3, $\|\mathbf{S}_a\|_2 \le 20\sqrt{m}$ except with negligible probability. Hence, $\|\mathbf{e}_0\mathbf{S}_a\|_2 \le 20 c_{\mathsf{G}}\chi m$. The correctness property of key-homomorphic evaluation (Section 3.7), therefore, gives

$$\begin{aligned}\mathbf{c}_{f_\lambda} &= \mathbf{s}\big(f_\lambda(x)\mathbf{G} + \mathbf{B}_{f_\lambda}\big) + \mathbf{e}_{f_\lambda}, \\ \|\mathbf{e}_{f_\lambda}\|_2 &\le 20 c_{\mathsf{G}}\alpha_{\mathcal{F}}\chi m =: \Delta_f. \end{aligned}$$

Since $f_\lambda(x) = 0$, $\mathbf{c}_{f_\lambda} = \mathbf{s}\mathbf{B}_{f_\lambda} + \mathbf{e}_{f_\lambda}$.
Together with $\mathbf{c}_{\mathsf{tg}} = \mathbf{s}(\mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1) + \mathbf{e}_{\mathsf{tg}}$ and Eq. (6), the circuit obtains

$$\overline{\mathbf{c}} = \mathbf{s}\mathbf{F}_{\mathsf{dig}_S, f_\lambda, \mathsf{tg}} + \mathbf{e}_{\mathsf{ct}}, \tag{8}$$

where $\mathbf{e}_{\mathsf{ct}} := [\mathbf{e}_0 \mid \mathbf{e}_{\mathsf{dig}} \mid \mathbf{e}_{\mathsf{tg}} \mid \mathbf{e}_{f_\lambda}]$. Again by Lemma 2, $\|\mathbf{e}_{\mathsf{tg}}\|_2 \le c_{\mathsf{G}}\chi'\sqrt{m}$ except with negligible probability. Thus,

$$\|\mathbf{e}_{\mathsf{ct}}\|_2 \le \Delta_{\mathsf{ct}} := c_{\mathsf{G}}(\chi + \chi')\sqrt{m} + \Delta_{\mathsf{dig}} + \Delta_f. \tag{9}$$

The important point is that Shamir reconstruction recovers the exact matrix $\mathbf{R}$ before any noisy lattice computation is performed. Combining Eqs. (4) and (8) gives

$$\overline{\mathbf{c}}\mathbf{R} = \mathbf{s}\mathbf{D} + \mathbf{e}_{\mathsf{ct}}\mathbf{R} \pmod q. \tag{10}$$

Since $\|\mathbf{R}\|_2 \le B_{\mathbf{R}} = 2\sigma\sqrt{m\ell_{\mathsf{lock}}}$, we have

$$\|\mathbf{e}_{\mathsf{ct}}\mathbf{R}\|_2 \le 2\sigma\sqrt{m\ell_{\mathsf{lock}}}\,\Delta_{\mathsf{ct}}. \tag{11}$$

In particular, no Shamir interpolation coefficient appears in the lattice noise bound.
Finally, $\mathbf{c}_{\mathsf{out}} = \mathbf{s}\mathbf{D} + \mathbf{e}_{\mathsf{out}} + \lfloor q/2 \rfloor \varsigma$. Therefore,

$$\mathbf{v} = \mathbf{c}_{\mathsf{out}} - \overline{\mathbf{c}}\mathbf{R} = \left\lfloor \frac{q}{2} \right\rfloor \varsigma + \mathbf{e}_{\mathsf{out}} - \mathbf{e}_{\mathsf{ct}}\mathbf{R} \pmod q. \tag{12}$$

By Lemma 2, $\|\mathbf{e}_{\mathsf{out}}\|_2 \le c_{\mathsf{G}}\chi'\sqrt{\ell_{\mathsf{lock}}}$ except with negligible probability. Together with Eq. (11), the total error has Euclidean norm at most $c_{\mathsf{G}}\chi'\sqrt{\ell_{\mathsf{lock}}} + 2\sigma\sqrt{m\ell_{\mathsf{lock}}}\,\Delta_{\mathsf{ct}}$. By Eq. (3), the total error is strictly smaller than $\frac{q-2}{4}$. Hence, every coordinate of the error has magnitude strictly smaller than $(q-2)/4$. Since q is odd and a bit 1 is encoded as $\lfloor q/2 \rfloor = (q-1)/2$, this error bound guarantees that rounding to the nearest multiple of $q/2$ recovers the encoded bit correctly. Therefore, $\varsigma' = \lfloor (2/q)\mathbf{v} \rceil \bmod 2 = \varsigma$. Perfect correctness of compute-and-compare obfuscation therefore gives $\mathsf{CC.Eval}\left(\widetilde{Q}, \left(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \{d_i\}_{i\in U_0}\right)\right) = \mu$. The Gaussian-tail, Gaussian-matrix, Gaussian-preimage-sampling, and trapdoor-generation good events used above fail only with negligible probability. Since only polynomially many such events are invoked, a union bound yields

$$\Pr\left[\mathsf{Combine}\left(\mathsf{mpk}, \mathsf{ct}, \{d_i\}_{i\in U}\right) = \mu\right] \ge 1 - \mathsf{negl}(\lambda).$$

For completeness, suppose instead that $x \notin S$ or $f_\lambda(x) \neq 0$. In the first case, the decryption circuit outputs $0^{\ell_{\mathsf{lock}}}$ at its batch-membership check, while in the second case it does so at its predicate check. Since the actual lock ς is uniform over $\{0,1\}^{\ell_{\mathsf{lock}}}$ and $\ell_{\mathsf{lock}} \ge \lambda$, the probability that $\varsigma = 0^{\ell_{\mathsf{lock}}}$ is at most $2^{-\ell_{\mathsf{lock}}} \le 2^{-\lambda}$. Therefore, the compute-and-compare program outputs $\perp$ except with negligible probability. This completes the proof.

Theorem 3 (Query-Selective One-Sided Security of BT-PE). *Assume the $2m^2$-succinct LWE assumption of Assumption 3.9, the properties of Shamir secret sharing (Appendix 3.3), the post-quantum digital signature scheme of Section 3.2, and the security of the compute-and-compare obfuscator of Appendix 3.1. Assume further that q is prime and that the parameters satisfy the hypotheses of Lemmas 1, 5, and 6, and Corollary 1, for every matrix and trapdoor occurring below. Then, the construction of Section 5 satisfies the security notion of Section 4.3.*

Proof. Fix an admissible QPT adversary $\mathcal{A}$, together with its selectively chosen challenge attributes $x_0^*, x_1^* \in \{0,1\}^k$ and challenge label $\mathsf{tg}^* \in \Gamma$. Let $\mathcal{Q}^* = \{(i, S_i^*, f_i^*)\}_{i\in\mathcal{C}^*}$ be its declared jointly challenge-compatible query set, where $|\mathcal{C}^*| \le T-1$. By the definition in Section 4.3, the same compatibility restriction applies to both $\mathcal{O}_{\mathsf{KeyGen}}$ and $\mathcal{O}_{\mathsf{PDec}}$, irrespective of the ciphertext supplied to the latter oracle. Consequently, every admissible query outside $\mathcal{Q}^*$ is challenge-incompatible with respect to both challenge attributes. For a jointly challenge-compatible key instance $\theta = (S, f, \mathsf{tg}^*)$, define $\mathcal{C}_\theta^* := \{i \in \mathcal{C}^* : (i, S, f) \in \mathcal{Q}^*\}$.

Then, $|\mathcal{C}^*_\theta| \le T-1$. Fix $b \in \{0,1\}$ and define $\mathsf{ID}^*_b := (1, \iota(x^*_b), \ldots, \iota(x^*_b)^B) \otimes \mathbf{g}'$ and $\mathbf{C}^*_b \leftarrow \mathsf{Com}(\mathsf{pp}_{\mathsf{com}}, \mathsf{ID}^*_b)$. We consider the following sequence of hybrids.

$\mathbf{Hyb}_0(b)$. This is $\mathsf{Exp}^{\mathsf{bt\text{-}pe}}_{\mathcal{A}}(\lambda, b)$. At challenge time the challenger samples $\mathsf{cid}^* \leftarrow \{0,1\}^{2\lambda}$ and $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$, constructs $\mathsf{ct}^*_{\mathsf{inner}}$ for (x^*_b, tg^*), and returns $\mathsf{ct}^* = (\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}}, \widetilde{Q}^*)$, $\widetilde{Q}^* \leftarrow \mathsf{CC.Obf}(1^\lambda, \mathcal{D}^*_b, \mu_b, \varsigma)$.

$\mathbf{Hyb}_1(b)$. This is a distributional reference experiment. Replace the setup matrix generated by $\mathsf{TrapGen}$ with $\mathbf{A} \leftarrow \mathbb{Z}_q^{\ell \times m}$ uniformly. Sample $\mathbf{W} \leftarrow \mathbb{Z}_q^{2m^2 \ell \times m}$ as before, but replace the public matrix-commitment trapdoor by a direct canonical sample $\mathbf{T} \leftarrow D_{\sigma_0}\left(\Lambda_q^{\mathbf{I}_{2m^2} \otimes \mathbf{G}}\left([\mathbf{I}_{2m^2} \otimes \mathbf{A} \mid \mathbf{W}]\right)\right)$. Likewise, whenever an honest key instance (S, f, tg) is generated, replace the output of $\mathsf{SamplePre}$ by a direct sample from the corresponding canonical Gaussian coset $D_\sigma\left(\Lambda_q^{\mathbf{D}}(\mathbf{F}_{\mathsf{dig}_S, f, \mathsf{tg}})\right)$, conditioned on the same norm check $\|\mathbf{R}\|_2 \le B_{\mathbf{R}}$. All Shamir sharing, signature-key generation, certification, ciphertext-core-specific signing, encryption, and obfuscation are otherwise unchanged.

$\mathbf{Hyb}_2(b)$. Sample $\mathbf{U}_{\mathsf{bat}}, \mathbf{U}_{\mathsf{tg}} \leftarrow \{0,1\}^{m \times m}$, $\mathbf{U}_{\mathbf{D}} \leftarrow \{0,1\}^{m \times \ell_{\mathsf{lock}}}$, and $\mathbf{S}^*_1, \ldots, \mathbf{S}^*_k \leftarrow \{\pm 1\}^{m \times m}$. Generate $(\mathbf{L}_1, \mathbf{T}_{\mathbf{L}_1}) \leftarrow \mathsf{TrapGen}(1^\ell, 1^m, q)$ and program

$$\mathbf{Q} := \mathbf{A}\mathbf{U}_{\mathsf{bat}} - \mathbf{C}^*_b, \quad \mathbf{L}_0 := \mathbf{A}\mathbf{U}_{\mathsf{tg}} - \mathsf{tg}^*\mathbf{L}_1,$$
$$\mathbf{D} := \mathbf{A}\mathbf{U}_{\mathbf{D}}, \qquad \mathbf{B}_a := \mathbf{A}\mathbf{S}^*_a - x^*_{b,a}\mathbf{G} \qquad (a \in [k]).$$

The matrices $\mathbf{S}^*_a$ are also used in the challenge ciphertext.

$\mathbf{Hyb}_3(b)$. For each jointly challenge-compatible key instance $\theta = (S, f, \mathsf{tg}^*)$, the challenger does not generate the underlying lattice preimage $\mathbf{R}_\theta$. Instead, it samples $\mathsf{kid}_\theta \leftarrow \{0,1\}^\lambda$ and, independently for each $i \in \mathcal{C}^*_\theta$, samples $\mathsf{sh}_i \leftarrow \mathbb{F}_{\mathsf{SS}}^{4m \times \ell_{\mathsf{lock}}}$ uniformly. For each such i, generate $(\mathsf{tvk}_i, \mathsf{tsk}_i) \leftarrow \mathsf{SIG.Gen}(1^\lambda)$, let $\mathsf{dig}_S \leftarrow \mathsf{Digest}(\mathsf{mpk}, S)$, and define

$$\mathsf{auth}_i := (i, \mathsf{kid}_\theta, f, \mathsf{tg}^*, \mathsf{dig}_S, \mathsf{sh}_i, \mathsf{tvk}_i), \qquad \mathsf{cert}_i := \mathsf{SIG.Sign}(\mathsf{ssk}_{\mathsf{root}}, \mathsf{auth}_i).$$

A compatible key query returns the corresponding programmed secret-key share. For a compatible partial-decryption query on $\mathsf{ct} = (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, \widetilde{Q})$, the challenger forms $\mathsf{tok}_i := (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, i, S, \mathsf{kid}_\theta, f, \mathsf{tg}^*, \mathsf{dig}_S, \mathsf{sh}_i)$, and computes $\tau_i \leftarrow \mathsf{SIG.Sign}(\mathsf{tsk}_i, \mathsf{tok}_i)$. Hence, every returned partial-decryption token is cryptographically bound to the ciphertext core $(\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}})$. All challenge-incompatible key instances remain canonical Gaussian samples.

$\mathbf{Hyb}_4(b)$. This hybrid is identical to $\mathbf{Hyb}_3(b)$, except that the experiment records two authentication bad events. Let $\mathsf{Bad}_{\mathsf{root}}$ be the event that the adversary produces an authentication tuple $\mathsf{auth} = (i, \mathsf{kid}, f, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}, \mathsf{tvk})$ and certificate cert satisfying $\mathsf{SIG.Ver}(\mathsf{vk}_{\mathsf{root}}, \mathsf{auth}, \mathsf{cert}) = 1$, although the challenger never signed auth under $\mathsf{ssk}_{\mathsf{root}}$. Similarly, let $\mathsf{Bad}_{\mathsf{tok}}$ be the event that, for some tvk_i whose signing key tsk_i is never revealed, the adversary produces (tok, τ) satisfying $\mathsf{SIG.Ver}(\mathsf{tvk}_i, \mathsf{tok}, \tau) = 1$, even though the challenger never signed tok under tsk_i, where $\mathsf{tok} := (\mathsf{cid}, \mathsf{ct}_{\mathsf{inner}}, i, S, \mathsf{kid}, f, \mathsf{tg}, \mathsf{dig}, \mathsf{sh}_i)$.

No token-forgery bad event is defined for a verification key whose corresponding signing key is revealed, since the adversary can then legitimately generate signatures under that key. If either $\mathsf{Bad}_{\mathsf{root}}$ or $\mathsf{Bad}_{\mathsf{tok}}$ occurs, the experiment outputs a fixed bit. Otherwise, it outputs exactly as in $\mathbf{Hyb}_3(b)$.

$\mathbf{Hyb}_5(b)$. This hybrid is identical to $\mathbf{Hyb}_4(b)$, except that the challenge ciphertext is rerandomized. Let $\mathbf{c}^*_{\mathsf{in}} = \mathbf{sA} + \mathbf{e}_0$ and, for each $a \in [k]$, set $\mathbf{c}^*_a := \mathbf{c}^*_{\mathsf{in}}\mathbf{S}^*_a$. The remaining challenge components are replaced by

$$\mathbf{c}^*_{\mathsf{bat}} := \mathsf{ReRand}\left(\mathbf{U}_{\mathsf{bat}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right), \quad \mathbf{c}^*_{\mathsf{tg}} := \mathsf{ReRand}\left(\mathbf{U}_{\mathsf{tg}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right),$$
$$\mathbf{c}^*_{\mathsf{out}} := \mathsf{ReRand}\left(\mathbf{U}_{\mathbf{D}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right) + \left\lfloor \frac{q}{2} \right\rfloor \varsigma.$$

Whenever the challenge ciphertext core changes, form the corresponding modified decryption circuit $\mathcal{D}^*_b$ with the current $(\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}})$ hardwired and recompute $\widetilde{Q}^* \leftarrow \mathsf{CC.Obf}(1^\lambda, \mathcal{D}^*_b, \mu_b, \varsigma)$.

Every subsequent $\mathcal{O}_{\mathsf{PDec}}$ response is generated with respect to the ciphertext core supplied in that query. In particular, a response to a query on the current challenge ciphertext is signed with respect to the current challenge core $(\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}})$. The authentication bad-event rule introduced in $\mathbf{Hyb}_4(b)$ continues to apply.

$\mathbf{Hyb}_6(b)$. For every challenge-incompatible key instance (S, f, tg) satisfying $x^*_b \notin S$, replace canonical key sampling by $\mathsf{SamplePre}$ using an alternative trapdoor obtained from the public gadget trapdoor $\mathbf{T}_{\mathbf{G}}$. Shamir sharing and both signature layers are subsequently generated honestly. Apply the same norm-rejection rule as in honest key generation: if $\|\mathbf{R}\|_2 > B_{\mathbf{R}}$, repeat the sampling step.

$\mathbf{Hyb}_7(b)$. For every remaining challenge-incompatible tuple satisfying $\mathsf{tg} \neq \mathsf{tg}^*$, generate its lattice preimage using $\mathbf{T}_{\mathbf{L}_1}$ and an alternative trapdoor for the correctly ordered key matrix $\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}}$.

$\mathbf{Hyb}_8(b)$. Every remaining challenge-incompatible tuple satisfies $x^*_b \in S$, $\mathsf{tg} = \mathsf{tg}^*$, and $f(x^*_b) = 1$. Use $\mathsf{Eval}_{\mathsf{sim}}$ together with the public gadget trapdoor to generate its lattice preimage. After this hybrid, every admissible oracle query can be answered without $\mathbf{T}_{\mathbf{A}}$.

$\mathbf{Hyb}_9(b)$. Replace $\mathbf{c}^*_{\mathsf{in}} = \mathbf{sA} + \mathbf{e}_0$ by $\mathbf{c}^*_{\mathsf{in}} \leftarrow \mathbb{Z}_q^{1\times m}$ uniformly. Compute $\mathbf{c}^*_a$, $\mathbf{c}^*_{\mathsf{bat}}$, $\mathbf{c}^*_{\mathsf{tg}}$, and $\mathbf{c}^*_{\mathsf{out}}$ from this uniform value exactly as in $\mathbf{Hyb}_5(b)$.

$\mathbf{Hyb}_{10}(b)$. Replace $\mathbf{c}^*_{\mathsf{bat}}$ and $\mathbf{c}^*_{\mathsf{tg}}$ by independently uniform vectors in $\mathbb{Z}_q^{1\times m}$.

$\mathbf{Hyb}_{11}(b)$. Replace $\mathbf{c}^*_{\mathsf{out}} = \mathsf{ReRand}\left(\mathbf{U}_{\mathbf{D}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right) + \left\lfloor \frac{q}{2} \right\rfloor \varsigma$ by $\mathbf{c}^*_{\mathsf{out}} = \mathsf{ReRand}\left(\mathbf{U}_{\mathbf{D}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right)$. The comparison value of the compute-and-compare obfuscator remains the independently sampled uniform string $\varsigma \in \{0,1\}^{\ell_{\mathsf{lock}}}$.

$\mathbf{Hyb}_{12}(b)$. Pad the challenge decryption circuit to a common size $s_{\mathcal{D}}$ and replace $\widetilde{Q}^* = \mathsf{CC.Obf}(1^\lambda, \mathcal{D}^*_b, \mu_b, \varsigma)$ by $\widetilde{Q}^* \leftarrow \mathsf{Sim}_{\mathsf{CC}}(1^\lambda, 1^{s_{\mathcal{D}}}, 1^h)$.

We will show by investigating the adjacent hybrids.

Lemma 11. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_0(b) \approx_s \boldsymbol{Hyb}_1(b)$.*

Proof. In the real setup, $(\mathbf{A}, \mathbf{T}_\mathbf{A}) \leftarrow \mathsf{TrapGen}(1^\ell, 1^m, q)$. By Lemma 1, the marginal distribution of $\mathbf{A}$ is negligibly close to uniform over $\mathbb{Z}_q^{\ell \times m}$. Conditioned on $\mathbf{A}$ and $\mathbf{W}$, the setup algorithm computes a valid trapdoor $\mathbf{T}'$ for $[\mathbf{I}_{2m^2} \otimes \mathbf{A} \mid \mathbf{W}]$ and invokes $\mathsf{SamplePre}$ with target $\mathbf{I}_{2m^2} \otimes \mathbf{G}$. By Lemma 1, its output is statistically close to $D_{\sigma_0}\left(\Lambda_q^{\mathbf{I}_{2m^2}\otimes\mathbf{G}}([\mathbf{I}_{2m^2} \otimes \mathbf{A} \mid \mathbf{W}])\right)$. Thus, the public triple $(\mathbf{A}, \mathbf{W}, \mathbf{T})$ is statistically close to the normalized distribution used in $\mathbf{Hyb}_1(b)$.

Similarly, conditioned on any public key matrix $\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}}$ and target $\mathbf{D}$, every honest invocation of $\mathsf{SamplePre}$ is statistically close to the canonical distribution $D_\sigma(\Lambda_q^{\mathbf{D}}(\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}}))$. Conditioning on the common norm test $\|\mathbf{R}\|_2 \le B_\mathbf{R}$ changes this distance only negligibly, because the rejected Gaussian tail has negligible probability.

After these replacements, all remaining algorithms, such as Shamir secret sharing, token-key generation, certification, partial-decryption signing, encryption, and obfuscation, form a common randomized response kernel determined by the normalized public parameters. The claim therefore extends to adaptive oracle interaction by induction over the adversary's polynomially many queries. Hence, $\mathbf{Hyb}_0(b) \approx_s \mathbf{Hyb}_1(b)$.

Lemma 12. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_1(b) \approx_s \boldsymbol{Hyb}_2(b)$.*

Proof. In $\mathbf{Hyb}_1(b)$, the matrix $\mathbf{A}$ is exactly uniform, so Lemma 5 applies directly. For a binary matrix $\mathbf{U}$, define $\widetilde{\mathbf{U}} := 2\mathbf{U} - \mathbf{J}$. Then $\widetilde{\mathbf{U}} \in \{\pm 1\}^{m \times d}$ and, since q is odd, $\mathbf{AU} = \frac{1}{2}\left(\mathbf{A}\widetilde{\mathbf{U}} + \mathbf{AJ}\right) \pmod q$. Thus, the leftover-hash statement for sign matrices implies the corresponding statement for the binary matrices $\mathbf{U}_{\mathsf{bat}}$, $\mathbf{U}_{\mathsf{tg}}$, and $\mathbf{U}_\mathbf{D}$.

Apply Lemma 5 jointly to the concatenation of these matrices together with $\mathbf{S}_1^*, \ldots, \mathbf{S}_k^*$. For the attribute matrices, condition on the fixed value of the encryption error $\mathbf{e}_0$. The lemma preserves the correlated side information $\mathbf{e}_0\mathbf{S}_a^*$. We obtain that $\mathbf{AU}_{\mathsf{bat}}$, $\mathbf{AU}_{\mathsf{tg}}$, $\mathbf{AU}_\mathbf{D}$, and $\mathbf{AS}_a^*$ are jointly statistically close to independent uniform matrices with the required correlated ciphertext values. Note that in this hybrid the challenge ciphertext is still generated honestly, so $\mathbf{U}_{\mathsf{bat}}$, $\mathbf{U}_{\mathsf{tg}}$ and $\mathbf{U}_\mathbf{D}$ enter the adversary's view only through $\mathbf{Q}$, $\mathbf{L}_0$ and $\mathbf{D}$; the rerandomized components that depend on these matrices in a stronger way are introduced only in $\mathbf{Hyb}_5(b)$ and are handled there and in Lemmas 20 and 21. The public canonical trapdoor $\mathbf{T}$ causes no difficulty: conditioned on $(\mathbf{A}, \mathbf{W})$, it is sampled independently of all matrices $\mathbf{U}_{\mathsf{bat}}$, $\mathbf{U}_{\mathsf{tg}}$, $\mathbf{U}_\mathbf{D}$, and $\mathbf{S}_a^*$. Hence, $(\mathbf{W}, \mathbf{T})$ may be appended as side information.

Affine shifts by $-\mathbf{C}_b^*$, $-\mathsf{tg}^*\mathbf{L}_1$, and $-x_{b,a}^*\mathbf{G}$ preserve statistical distance. Finally, the marginal of $\mathbf{L}_1$ produced by $\mathsf{TrapGen}$ is statistically close to uniform. Since all key outputs in the normalized experiment are canonical Gaussian post-

processing of the resulting public matrices, adaptive oracle responses preserve statistical indistinguishability.

Lemma 13. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_2(b) \approx_s \boldsymbol{Hyb}_3(b)$ up to negligible norm-tail events.*

Proof. Fix a jointly compatible key instance $\theta = (S, f, \mathsf{tg}^*)$. In $\mathbf{Hyb}_2(b)$, its lattice key $\mathbf{R}_\theta$ is canonically sampled and $\widehat{\mathbf{R}}_\theta = \mathsf{Encode}_{\mathsf{SS}}(\mathbf{R}_\theta)$ is Shamir-shared. Since $|\mathcal{C}^*_\theta| \le T-1$, the privacy property in Appendix 3.3 implies that $\{\mathsf{sh}_i\}_{i\in\mathcal{C}^*_\theta}$ is exactly uniform and independent of $\widehat{\mathbf{R}}_\theta$. Equivalently, these shares may be sampled first and the remaining shares may be completed using $\mathsf{SS.Complete}_{T,N}$. The resulting full sharing is identically distributed to an honest sharing. We stress that this argument is applied separately to each jointly compatible key instance, which is exactly what the per-instance form of the query-declaration constraint provides.

Moreover, $\mathsf{SS.Explain}_{T,N}$ recovers sharing randomness having exactly the honest distribution, so any internal sharing coins can also be completed consistently.

By the admissibility rule, no party outside $\mathcal{C}^*_\theta$ can obtain a share of this jointly compatible key instance through either oracle. Hence, the unexposed shares and $\mathbf{R}_\theta$ never enter the adversary's view and can be omitted. The token-signing key pairs, root certificates, and ciphertext-core-specific signatures are then generated honestly from identically distributed inputs. Applying the same argument to every jointly compatible key instance proves the claim.

Lemma 14. *Assume that $\prod_{\mathsf{SIG}}$ is post-quantum EUF-CMA secure. Then, for every $b \in \{0,1\}$, $\boldsymbol{Hyb}_3(b) \approx_c \boldsymbol{Hyb}_4(b)$.*

Proof. The two experiments are identical unless $\mathsf{Bad}_{\mathsf{root}}$ or $\mathsf{Bad}_{\mathsf{tok}}$ occurs. Hence,

$$\Big|\Pr[\mathbf{Hyb}_3(b)=1] - \Pr[\mathbf{Hyb}_4(b)=1]\Big| \le \Pr[\mathsf{Bad}_{\mathsf{root}}] + \Pr[\mathsf{Bad}_{\mathsf{tok}}]. \tag{13}$$

We first bound $\Pr[\mathsf{Bad}_{\mathsf{root}}]$. Construct a post-quantum EUF-CMA adversary $\mathcal{B}_{\mathsf{root}}$ against the root signature key. The reduction receives a challenge verification key vk and sets $\mathsf{vk}_{\mathsf{root}} := \mathsf{vk}$. Whenever the BT-PE simulation requires a root certificate on an authentication tuple auth_i, it submits auth_i to its signing oracle and uses the returned signature as cert_i. All remaining components are generated exactly as in $\mathbf{Hyb}_3(b)$.

If $\mathsf{Bad}_{\mathsf{root}}$ occurs, the simulation obtains an accepted pair $(\mathsf{auth}, \mathsf{cert})$ with $\mathsf{SIG.Ver}(\mathsf{vk}_{\mathsf{root}}, \mathsf{auth}, \mathsf{cert}) = 1$, although auth was never submitted to the signing oracle. Thus, $(\mathsf{auth}, \mathsf{cert})$ is a valid EUF-CMA forgery, so

$$\Pr[\mathsf{Bad}_{\mathsf{root}}] \le \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\prod_{\mathsf{SIG}},\mathcal{B}_{\mathsf{root}}}(\lambda).$$

We next bound $\Pr[\mathsf{Bad}_{\mathsf{tok}}]$. At most $q_{\mathsf{auth}}(\lambda)$ token-verification keys are generated. The reduction $\mathcal{B}_{\mathsf{tok}}$ samples $j^* \leftarrow [q_{\mathsf{auth}}(\lambda)]$ uniformly and guesses that the j^*-th generated token-verification key is involved in $\mathsf{Bad}_{\mathsf{tok}}$.

When the j^*-th token-verification key is generated, $\mathcal{B}_{\mathsf{tok}}$ embeds its $\mathsf{EUF\text{-}CMA}$ challenge verification key as tvk_{j^*}. Whenever a partial-decryption token must be signed under this key, the reduction submits the corresponding token message to its signing oracle and uses the returned signature. Every other token-signing key pair is generated honestly.

If the signing key corresponding to the embedded verification key is later requested or otherwise revealed, $\mathcal{B}_{\mathsf{tok}}$ aborts that execution. This causes no loss when the guessed key is the one associated with $\mathsf{Bad}_{\mathsf{tok}}$, since such a bad event is defined only for a token key whose signing key is never revealed.

Conditioned on guessing the relevant token-verification key, $\mathcal{B}_{\mathsf{tok}}$ obtains an accepted pair (tok, τ) satisfying $\mathsf{SIG.Ver}(\mathsf{tvk}_{j^*}, \mathsf{tok}, \tau) = 1$, where tok was never submitted to its signing oracle. Hence, (tok, τ) is a valid $\mathsf{EUF\text{-}CMA}$ forgery. Since the correct key is guessed with probability at least $1/q_{\mathsf{auth}}(\lambda)$,

$$\Pr[\mathsf{Bad}_{\mathsf{tok}}] \leq q_{\mathsf{auth}}(\lambda)\, \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\prod_{\mathsf{SIG}}, \mathcal{B}_{\mathsf{tok}}}(\lambda).$$

Substituting these two bounds into Eq. (13) yields $\Big|\Pr[\mathbf{Hyb}_3(b) = 1] - \Pr[\mathbf{Hyb}_4(b) = 1]\Big| \leq \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\prod_{\mathsf{SIG}}, \mathcal{B}_{\mathsf{root}}}(\lambda) + q_{\mathsf{auth}}(\lambda)\, \mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\prod_{\mathsf{SIG}}, \mathcal{B}_{\mathsf{tok}}}(\lambda)$. Post-quantum $\mathsf{EUF\text{-}CMA}$ security therefore gives $\mathbf{Hyb}_3(b) \approx_c \mathbf{Hyb}_4(b)$.

Lemma 15. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_4(b) \approx_s \boldsymbol{Hyb}_5(b)$.*

Proof. For every $a \in [k]$, the programmed public attribute matrix satisfies $\mathbf{B}_a = \mathbf{A}\mathbf{S}^*_a - x^*_{b,a}\mathbf{G}$, and hence, $x^*_{b,a}\mathbf{G} + \mathbf{B}_a = \mathbf{A}\mathbf{S}^*_a$. Therefore, the honest challenge attribute component satisfies

$$\begin{aligned}\mathbf{c}^*_a &= \mathbf{s}(x^*_{b,a}\mathbf{G} + \mathbf{B}_a) + \mathbf{e}_0\mathbf{S}^*_a \\ &= \mathbf{s}\mathbf{A}\mathbf{S}^*_a + \mathbf{e}_0\mathbf{S}^*_a = (\mathbf{s}\mathbf{A} + \mathbf{e}_0)\mathbf{S}^*_a = \mathbf{c}^*_{\mathsf{in}}\mathbf{S}^*_a.\end{aligned}$$

Thus, replacing the challenge attribute components introduces no statistical error.

By the programming of $\mathbf{Q}$, $\mathbf{L}_0$, and $\mathbf{D}$, we also have $\mathbf{Q} + \mathbf{C}^*_b = \mathbf{A}\mathbf{U}_{\mathsf{bat}}$, $\mathbf{L}_0 + \mathsf{tg}^*\mathbf{L}_1 = \mathbf{A}\mathbf{U}_{\mathsf{tg}}$, and $\mathbf{D} = \mathbf{A}\mathbf{U}_{\mathbf{D}}$.

Recall that $\mathbf{c}^*_{\mathsf{in}} = \mathbf{s}\mathbf{A} + \mathbf{e}_0$, where $\mathbf{e}_0 \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times m}, \chi}$. Hence, the honest challenge batch component is distributed as $\mathbf{c}^*_{\mathsf{bat}} = \mathbf{s}\mathbf{A}\mathbf{U}_{\mathsf{bat}} + \mathbf{e}_{\mathsf{bat}}$, where $\mathbf{e}_{\mathsf{bat}} \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times m}, \chi'}$. Applying Lemma 6 with $\mathbf{R} = \mathbf{U}_{\mathsf{bat}}$ and $\rho = \chi'/(2\chi)$ gives

$$\mathbf{c}^*_{\mathsf{bat}} \approx_s \mathsf{ReRand}\left(\mathbf{U}_{\mathsf{bat}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right),$$

provided $\chi'/(2\chi) > \|\mathbf{U}_{\mathsf{bat}}\|_2$.

The same argument with $\mathbf{R} = \mathbf{U}_{\mathsf{tg}}$ gives the tag-component replacement provided $\chi'/(2\chi) > \|\mathbf{U}_{\mathsf{tg}}\|_2$.

Likewise, using $\mathbf{R} = \mathbf{U_D}$ gives

$$\mathsf{ReRand}\left(\mathbf{U_D}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right) \approx_s \mathbf{sD} + \mathbf{e}_{\mathsf{out}},$$

where $\mathbf{e}_{\mathsf{out}} \leftarrow \mathcal{D}_{\mathbb{Z}^{1\times \ell_{\mathsf{lock}}},\chi'}$, provided $\chi'/(2\chi) > \|\mathbf{U_D}\|_2$. Adding the common deterministic shift $\lfloor q/2 \rfloor \varsigma$ preserves statistical distance, so the output component is replaced with only negligible statistical loss.

By the noise-width condition imposed in Section 5, namely $\chi'/(2\chi) > \sqrt{m \max\{m, \ell_{\mathsf{lock}}\}}$ together with $\|\mathbf{U}\|_2 \le \|\mathbf{U}\|_F \le \sqrt{md}$ for binary $m \times d$ matrices,

$$\frac{\chi'}{2\chi} > \max\left\{\|\mathbf{U}_{\mathsf{bat}}\|_2,\ \|\mathbf{U}_{\mathsf{tg}}\|_2,\ \|\mathbf{U_D}\|_2\right\}.$$

Thus, by three applications of Lemma 6 and the triangle inequality, the challenge ciphertext cores in $\mathbf{Hyb}_4(b)$ and $\mathbf{Hyb}_5(b)$ are statistically indistinguishable.

Whenever the challenge ciphertext core changes, the corresponding decryption circuit $\mathcal{D}^*_b$ is formed with the current $(\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}})$ hardwired, and the challenge obfuscation is recomputed as $\widetilde{Q}^* \leftarrow \mathsf{CC.Obf}(1^\lambda, \mathcal{D}^*_b, \mu_b, \varsigma)$. Circuit construction and $\mathsf{CC.Obf}$ are efficient randomized post-processing of statistically close challenge-core distributions and therefore do not increase their statistical distance.
Similarly, every post-challenge $\mathcal{O}_{\mathsf{PDec}}$ response is generated relative to the ciphertext core supplied in its query. The resulting ciphertext-core-specific signatures, certificates, and other oracle outputs are therefore efficient randomized post-processing of statistically close underlying values.

Finally, the authentication bad-event rule from $\mathbf{Hyb}_4(b)$ is applied identically in both experiments. It is common post-processing of the respective transcripts and does not increase statistical distance. The same reasoning applies inductively over the adversary's polynomially many adaptive oracle queries. Consequently, $\mathbf{Hyb}_4(b) \approx_s \mathbf{Hyb}_5(b)$.

Lemma 16. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_5(b) \approx_s \boldsymbol{Hyb}_6(b)$.*

Proof. Fix an incompatible tuple (S, f, tg) satisfying $x^*_b \notin S$. Let $\gamma \leftarrow \mathsf{GenCoeff}(\iota(S))$, $\mathbf{Y}_S := \gamma^\top \otimes \mathbf{I}_m$, and $\beta_S := P_{\iota(S)}(\iota(x^*_b))$. Then $\beta_S \in \mathbb{Z}^*_q$.
Using the programming of $\mathbf{Q}$ and Lemma (10), we obtain

$$\begin{aligned}\mathsf{dig}_S &= -\mathbf{QVY}_S \\ &= \mathbf{AR}_S + \beta_S \mathbf{G},\end{aligned}$$

where $\mathbf{R}_S := -(\mathbf{U}_{\mathsf{bat}}\mathbf{V} + \mathbf{Z}^*_b)\mathbf{Y}_S$ and $\mathbf{Z}^*_b := \mathsf{Open}(\mathsf{pp}_{\mathsf{com}}, \mathsf{ID}^*_b)$.

Since $\Lambda^\perp_q(\beta_S \mathbf{G}) = \Lambda^\perp_q(\mathbf{G})$, $\mathbf{T_G}$ is a short trapdoor for $\beta_S \mathbf{G}$. Applying $\mathsf{ExtendLeft}$ yields a trapdoor for $[\mathbf{A} \mid \mathsf{dig}_S]$ in the correct column order; the tag and predicate blocks are then appended using $\mathsf{ExtendRight}$. By Lemma 1, the Gram-Schmidt norm of the resulting trapdoor is at most $\|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1 + \|\mathbf{R}_S\|_2) \le \|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1 + \rho_{\mathsf{bat}}) \le$

τ_{sim}, so the condition $\sigma \geq \tau_{\mathsf{sim}}\,\omega(\sqrt{\log 4m})$ imposed in Section 5 allows $\mathsf{SamplePre}$ to be invoked with the same width σ as in honest key generation.

By Lemma 1, sampling with this alternative trapdoor is statistically close to the canonical Gaussian distribution already used in $\mathbf{Hyb}_5(b)$. Shamir sharing and the signature layers preserve this distance.

Lemma 17. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_6(b) \approx_s \boldsymbol{Hyb}_7(b)$.*

Proof. Consider a remaining incompatible tuple with $\mathsf{tg} \neq \mathsf{tg}^*$. Let $\delta_{\mathsf{tg}} := \mathsf{tg} - \mathsf{tg}^* \in \mathbb{Z}_q^*$. Then

$$\mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 = \mathbf{A}\mathbf{U}_{\mathsf{tg}} + \delta_{\mathsf{tg}}\mathbf{L}_1.$$

Set $\mathbf{F}_0 := [\mathbf{A} \mid \mathsf{dig}]$. Then

$$\mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1 = \mathbf{F}_0 \begin{pmatrix} \mathbf{U}_{\mathsf{tg}} \\ \mathbf{0} \end{pmatrix} + \delta_{\mathsf{tg}}\mathbf{L}_1.$$

Since $\mathbf{T}_{\mathbf{L}_1}$ is also a trapdoor for $\delta_{\mathsf{tg}}\mathbf{L}_1$, $\mathsf{ExtendLeft}$ gives a trapdoor for

$$[\mathbf{A} \mid \mathsf{dig} \mid \mathbf{L}_0 + \mathsf{tg}\mathbf{L}_1]$$

in exactly the required order. The predicate block is appended by $\mathsf{ExtendRight}$. Its Gram-Schmidt norm is at most $\|\widetilde{\mathbf{T}}_{\mathbf{L}_1}\|(1 + \|(\mathbf{U}_{\mathsf{tg}}; \mathbf{0})\|_2) \leq \|\widetilde{\mathbf{T}}_{\mathbf{L}_1}\|(1 + \rho_{\mathsf{tg}}) \leq \tau_{\mathsf{sim}}$, so the width σ fixed in Section 5 again suffices.

The resulting preimage distribution is statistically close to the canonical Gaussian distribution of $\mathbf{Hyb}_6(b)$.

Lemma 18. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_7(b) \approx_s \boldsymbol{Hyb}_8(b)$.*

Proof. Every remaining incompatible tuple satisfies $x_b^* \in S$, $\mathsf{tg} = \mathsf{tg}^*$, and $f(x_b^*) = 1$. Thus, we get, $\mathsf{Eval}_{\mathsf{sim}}$ gives $\mathbf{S}_f$ satisfying $\mathbf{B}_f = \mathbf{A}\mathbf{S}_f - \mathbf{G}$.
Let $\mathbf{F}_0 := [\mathbf{A} \mid \mathsf{dig} \mid \mathbf{L}_0 + \mathsf{tg}^*\mathbf{L}_1]$. Then

$$\mathbf{B}_f = \mathbf{F}_0 \begin{pmatrix} \mathbf{S}_f \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} - \mathbf{G}.$$

The public trapdoor $\mathbf{T}_{\mathbf{G}}$ is also a trapdoor for $-\mathbf{G}$. Thus, $\mathsf{ExtendLeft}$ gives a trapdoor directly for $\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}^*}$ in its actual column order. Its Gram-Schmidt norm is at most $\|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1 + \|\mathbf{S}_f\|_2) \leq \|\widetilde{\mathbf{T}}_{\mathbf{G}}\|(1 + \alpha_{\mathcal{F}}) \leq \tau_{\mathsf{sim}}$, which is dominated by σ by the choice of parameters. The resulting Gaussian preimage distribution is statistically close to the canonical distribution in the preceding hybrid. All subsequent operations are post-processing.

Lemma 19. *Under Assumption 3.9, for every $b \in \{0,1\}$, $\boldsymbol{Hyb}_8(b) \approx_c \boldsymbol{Hyb}_9(b)$.*

Proof. The setup-normalization hybrid is essential here. A succinct LWE challenge provides $(\mathbf{A}, \mathbf{u}, \mathbf{W}, \mathbf{T})$ with $\mathbf{A}$ exactly uniform and $\mathbf{T}$ sampled from the

canonical Gaussian preimage distribution. This is precisely the setup distribution carried from $\mathbf{Hyb}_1$ onward.

The reduction sets $\mathsf{pp}_{\mathsf{com}} := (\mathbf{A}, \mathbf{W}, \mathbf{T})$, samples the programming matrices and the root signature key pair, and constructs the public parameters exactly as in $\mathbf{Hyb}_8(b)$. Compatible queries are answered using the perfect Shamir programming of $\mathbf{Hyb}_3$. Incompatible queries are answered using the alternative trapdoors of $\mathbf{Hyb}_6$–$\mathbf{Hyb}_8$. Thus, no trapdoor for $\mathbf{A}$ is required.

At challenge time the reduction sets $\mathbf{c}^*_{\mathsf{in}} := \mathbf{u}$, $\mathbf{c}^*_a := \mathbf{u}\mathbf{S}^*_a$, and computes all remaining components as in $\mathbf{Hyb}_5$.

If $\mathbf{u} = \mathbf{s}\mathbf{A} + \mathbf{e}_0$, the resulting experiment is $\mathbf{Hyb}_8(b)$; if $\mathbf{u}$ is uniform, it is $\mathbf{Hyb}_9(b)$. Hence, a distinguisher contradicts Assumption 3.9.

Lemma 20. *For every $b \in \{0,1\}$, $\boldsymbol{Hyb}_9(b) \approx_s \boldsymbol{Hyb}_{10}(b)$.*

Proof. By setup normalization, $\mathbf{A}$ is exactly uniform in this hybrid, so Corollary 1 applies directly.
Since $\mathbf{Q} + \mathbf{C}^*_b = \mathbf{A}\mathbf{U}_{\mathsf{bat}}$ and $\mathbf{c}^*_{\mathsf{in}}$ is uniform, Corollary 1 implies that $\mathbf{c}^*_{\mathsf{bat}}$ is statistically close to an independent uniform vector.
Similarly, $\mathbf{L}_0 + \mathsf{tg}^*\mathbf{L}_1 = \mathbf{A}\mathbf{U}_{\mathsf{tg}}$, so a second application gives the same conclusion for $\mathbf{c}^*_{\mathsf{tg}}$.

We next justify carrying the oracle transcript as side information. The public canonical trapdoor $\mathbf{T}$ causes no difficulty: conditioned on $(\mathbf{A}, \mathbf{W})$, it is sampled independently of $\mathbf{U}_{\mathsf{bat}}$ and $\mathbf{U}_{\mathsf{tg}}$. For jointly challenge-compatible key instances, the exposed Shamir shares are uniform and independent of the underlying lattice key and of the programming witnesses.

For challenge-incompatible key instances, the samples produced in $\mathbf{Hyb}_6$–$\mathbf{Hyb}_8$ using alternative trapdoors are only statistically close to the corresponding canonical Gaussian distributions. We therefore first conceptually replace every such sample by $D_\sigma\left(\Lambda_q^{\mathbf{D}}(\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}})\right)$ conditioned on $\|\mathbf{R}\|_2 \le B_{\mathbf{R}}$. By Lemmas 16-18, this changes the complete view by only negligible statistical distance. Conditioned on the public matrices and target $\mathbf{D}$, these canonical key distributions do not depend on the particular witnesses $\mathbf{U}_{\mathsf{bat}}$ or $\mathbf{U}_{\mathsf{tg}}$.

Hence, the canonicalized oracle transcript may be appended as side information to the two applications of Corollary 1. After applying output masking, the alternative-trapdoor samples may be restored, again paying only negligible statistical distance. For adaptive post-challenge queries, argue inductively over the transcript. Conditioned on statistically corresponding histories, the adversary submits the same next-query distribution, and the oracle applies the same lock-independent response kernel. Ciphertext-core-specific signatures are generated on the core of the current hybrid. Thus, statistical indistinguishability is preserved. Finally, because the modified batch and tag components belong to $\mathsf{ct}^*_{\mathsf{inner}}$, we form the corresponding modified decryption circuit and recompute its challenge obfuscation honestly. This is efficient randomized post-processing and therefore preserves the statistical indistinguishability established above.

Lemma 21. *For every* $b \in \{0,1\}$, $\boldsymbol{Hyb}_{10}(b) \approx_s \boldsymbol{Hyb}_{11}(b)$.

Proof. Again $\mathbf{A}$ is exactly uniform. Since $\mathbf{D} = \mathbf{A}\mathbf{U}_{\mathbf{D}}$ and $\mathbf{c}^*_{\mathsf{in}}$ is uniform, Corollary 1 gives

$$\mathsf{ReRand}\left(\mathbf{U}_{\mathbf{D}}, \mathbf{c}^*_{\mathsf{in}}, \frac{\chi'}{2\chi}\right) \approx_s \mathbf{u},$$

where $\mathbf{u}$ is independent and uniform in $\mathbb{Z}_q^{1\times \ell_{\mathsf{lock}}}$, jointly with $(\mathbf{A}, \mathbf{D}, \mathbf{c}^*_{\mathsf{in}})$.
Translation of a uniform vector by the fixed value $\lfloor q/2 \rfloor \varsigma$ preserves uniformity. Hence, the distributions of $\mathbf{u} + \left\lfloor \frac{q}{2} \right\rfloor \varsigma$ and $\mathbf{u}$ are identical. It follows that the challenge output component may be changed from an encryption of ς to an encryption of $0^{\ell_{\mathsf{lock}}}$. We emphasise that this transition is statistical, and therefore holds even against an adversary that is able to assemble an accepting input to the obfuscated decryption circuit; in particular it does not rely on unforgeability of $\prod_{\mathsf{SIG}}$ or on any binding property of the digest.

The same side-information and adaptive-transcript argument used in Lemma 20 applies here, since all oracle response mechanisms are independent of the witness $\mathbf{U}_{\mathbf{D}}$ once the public target $\mathbf{D}$ is fixed.

Lemma 22. *For every* $b \in \{0,1\}$, $\boldsymbol{Hyb}_{11}(b) \approx_c \boldsymbol{Hyb}_{12}(b)$.

Proof. We give an explicit reduction to compute-and-compare security, rather than treating the adaptive post-challenge transcript as static auxiliary information.
Suppose $\mathcal{A}$ distinguishes the two hybrids. Construct a compute-and-compare distinguisher $\mathcal{B}$ as follows. The reduction first generates the normalized BT-PE setup, answers all pre-challenge oracle queries exactly as in $\mathbf{Hyb}_{11}(b)$, and constructs the challenge ciphertext core $(\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}})$. By $\mathbf{Hyb}_{11}(b)$, the inner ciphertext encrypts the all-zero lock. Therefore, the resulting decryption circuit $\mathcal{D}^*_b$, the message μ_b, the public parameters, and the entire pre-challenge simulation state are generated independently of $\varsigma \leftarrow \{0,1\}^{\ell_{\mathsf{lock}}}$.

The reduction submits $(\mathcal{D}^*_b, \mu_b)$, together with its lock-independent pre-challenge state as auxiliary information, to the compute-and-compare experiment and receives a candidate string $\widetilde{Q}$, which is either $\mathsf{CC.Obf}(1^\lambda, \mathcal{D}^*_b, \mu_b, \varsigma)$ or $\mathsf{Sim}_{\mathsf{CC}}(1^\lambda, 1^{s_{\mathcal{D}}}, 1^h)$.
It gives $\mathsf{ct}^* = (\mathsf{cid}^*, \mathsf{ct}^*_{\mathsf{inner}}, \widetilde{Q})$ to $\mathcal{A}$ and continues running $\mathcal{A}$. Every subsequent BT-PE oracle query is answered by $\mathcal{B}$ itself. Crucially, after the zero-lock hybrid, none of these oracle response algorithms uses ς: compatible shares are generated from the programmed Shamir state, incompatible shares from the corresponding alternative trapdoors, and partial-decryption signatures depend only on the ciphertext core supplied in the query.

Thus, $\mathcal{B}$ can simulate the entire adaptive continuation without knowing ς. If the candidate is a real obfuscation, $\mathcal{A}$ sees $\mathbf{Hyb}_{11}(b)$. If it is simulated, $\mathcal{A}$ sees $\mathbf{Hyb}_{12}(b)$. Any non-negligible distinguishing advantage contradicts the security property in Appendix 3.1.

Lemma 23. *The two experiments* $\boldsymbol{Hyb}_{12}(0)$ *and* $\boldsymbol{Hyb}_{12}(1)$ *are statistically indistinguishable, i.e.,*

$$\boldsymbol{Hyb}_{12}(0) \approx_s \boldsymbol{Hyb}_{12}(1).$$

Proof. The simulated challenge obfuscation is independent of the challenge message and challenge attribute. It remains to compare the normalized public parameters, the exposed challenge core, and the adaptive oracle transcript.

First consider $\mathbf{Q} = \mathbf{A}\mathbf{U}_{\mathsf{bat}} - \mathbf{C}_b^*$. Since $\mathbf{U}_{\mathsf{bat}}$ is binary, define $\widetilde{\mathbf{U}}_{\mathsf{bat}} := 2\mathbf{U}_{\mathsf{bat}} - \mathbf{J}$. Then

$$\mathbf{A}\mathbf{U}_{\mathsf{bat}} = \frac{1}{2}\left(\mathbf{A}\widetilde{\mathbf{U}}_{\mathsf{bat}} + \mathbf{A}\mathbf{J}\right) \pmod q.$$

Because $\mathbf{A}$ is uniform, Lemma 5 implies that $\mathbf{A}\widetilde{\mathbf{U}}_{\mathsf{bat}}$, and hence, $\mathbf{A}\mathbf{U}_{\mathsf{bat}}$, is statistically close to an independent uniform matrix. After the affine shift by $-\mathbf{C}_b^*$, $\mathbf{Q}$ is therefore statistically close to uniform independently of x_b^*. The challenge batch component is already independent uniform by $\mathbf{Hyb}_{10}$.
Next concatenate all attribute sign matrices as

$$\mathbf{S}^* := [\mathbf{S}_1^* \mid \cdots \mid \mathbf{S}_k^*] \in \{\pm 1\}^{m \times km}.$$

Condition on the uniform row $\mathbf{c}_{\mathsf{in}}^*$. Lemma 5, applied once with output width km, gives

$$(\mathbf{A}, \mathbf{A}\mathbf{S}^*, \mathbf{c}_{\mathsf{in}}^*\mathbf{S}^*) \approx_s (\mathbf{A}, \mathbf{U}^*, \mathbf{c}_{\mathsf{in}}^*\mathbf{S}^*),$$

where $\mathbf{U}^*$ is independently uniform. Splitting into m-column blocks and subtracting $x_{b,a}^*\mathbf{G}$ from the a-th block shows that the joint distribution of $\{(\mathbf{B}_a, \mathbf{c}_a^*)\}_{a\in[k]}$ is statistically close to a distribution independent of x_b^*.

The tag programming is independent of b because the challenge tag tg^* is common to both experiments. The target $\mathbf{D} = \mathbf{A}\mathbf{U}_{\mathbf{D}}$ is also independent of b. By $\mathbf{Hyb}_{10}$ and $\mathbf{Hyb}_{11}$, the challenge tag component is uniform and the output component contains no lock. Hence, after the above statistical replacements, the entire public parameter tuple and challenge core have a common reference distribution independent of b.

It remains to compare adaptive oracle responses. We proceed inductively over the adversary's polynomially many queries. Conditioned on statistically corresponding histories, admissibility is identical in the two experiments, since the security notion of Section 4.3 is formulated jointly with respect to x_0^* and x_1^* and does not depend on b.

For a jointly challenge-compatible key instance, every exposed Shamir share is uniform and independent of its underlying lattice key by the privacy property in Appendix 3.3. The token-signing keys, root certificates, and ciphertext-core-specific signatures are generated by the same algorithms on statistically corresponding inputs.

For a challenge-incompatible tuple, the alternative-trapdoor hybrids show that its lattice key is statistically close to the canonical distribution $D_\sigma\left(\Lambda_q^{\mathbf{D}}(\mathbf{F}_{\mathsf{dig},f,\mathsf{tg}})\right)$,

conditioned only on the corresponding public query tuple. Once the public parameters have been replaced by their common reference distribution, this is the same response kernel in both experiments. Shamir sharing and the two signature layers are common post-processing.

Thus, each adaptive query preserves statistical indistinguishability. Induction over the polynomially many queries yields $\mathbf{Hyb}_{12}(0) \approx_s \mathbf{Hyb}_{12}(1)$.

Combining Lemmas 11-23 with the triangle inequality gives

$$\begin{aligned}\mathsf{Adv}^{\mathsf{bt\text{-}pe}}_{\mathcal{A}}(\lambda) \le\ & 2\,\mathsf{Adv}^{\mathsf{succLWE}}_{\mathcal{B}_1}(\lambda) + 2\,\mathsf{Adv}^{\mathsf{CC}}_{\mathcal{B}_2}(\lambda) \\ & + 2\,\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{B}_{\mathsf{root}}}(\lambda) + 2q_{\mathsf{auth}}(\lambda)\,\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathcal{B}_{\mathsf{tok}}}(\lambda) \\ & + \mathsf{negl}(\lambda).\end{aligned}$$

where the negligible term contains the losses from setup normalization, the matrix leftover-hash replacements, Gaussian sampling and norm conditioning, the three rerandomization replacements, the three output-masking replacements, the alternative-trapdoor transitions, and the statistical closeness of $\mathsf{TrapGen}$ to uniform. By Assumption 3.9 and the security of the compute-and-compare obfuscator, and the post-quantum $\mathsf{EUF\text{-}CMA}$ security of $\prod_{\mathsf{SIG}}$, the right-hand side is negligible. Therefore, $\prod_{\mathsf{BT\text{-}PE}}$ is query-selectively one-sided secure.

7 Conclusion

We introduced batched threshold predicate encryption, in which decryption is simultaneously restricted to a designated batch of attributes, to attributes satisfying an expressive predicate, and to coalitions of at least T out of N parties. Our lattice-based construction supports polynomial-size arithmetic-circuit predicates, represents a batch by a digest of size polylogarithmic in the batch size, and gives each party a single key share of size polylogarithmic in both the batch size and the number of parties. The key technical idea is to Shamir-share one short lattice preimage and reconstruct it exactly inside a compute-and-compare-protected decryption circuit, so that threshold interpolation never interacts with lattice noise and the modulus is independent of N and T. We proved query-selective one-sided security under the succinct LWE assumption, together with compute-and-compare obfuscation and post-quantum signatures. Several questions remain open.

- **Stronger security.** Our notion fixes the challenge attributes and the compatible query set in advance. Achieving adaptive security, or removing the query declaration, would considerably strengthen the guarantee.
- **Shorter ciphertexts.** Because the ciphertext contains an obfuscated decryption circuit, its size grows polynomially with B, T, and the predicate size. A construction that avoids generic compute-and-compare obfuscation and achieves ciphertexts independent of B, as in batched IBE [2, 13, 33], is an interesting goal.

– **Removing the trusted dealer.** Key generation is performed by an authority that knows the shared preimage $\mathbf{R}$. Distributed key generation, or a silent setup in the spirit of [13, 29], would remove this trust assumption.
– **Share-hiding partial decryption.** A partial-decryption token contains the party's Shamir share, so once T tokens of a key instance are revealed, the guarantee is lost for every ciphertext that this instance authorizes. This matches batch semantics, where these ciphertexts are opened together, but tokens that are useful only for the ciphertext at hand would be preferable.
– **Concrete efficiency.** Instantiating the construction over structured lattices and implementing it remain open.

Acknowledgments. This work is supported in part by the Anusandhan National Research Foundation (ANRF), Department of Science and Technology (DST), Government of India, under File No. EEQ/2023/000164, Defence Research and Development Organisation, Government of India, under File No. SAG/4600/TCID/CARS/C-90 and the Information Security Education and Awareness (ISEA) Project Phase-III initiatives of the Ministry of Electronics and Information Technology (MeitY) under Grant No. F.No. L-14017/1/2022-HRD. We thank the anonymous reviewers of ICISC 2026 for their helpful comments.

References

1. Agarwal, A., Das, S., Gilkalaye, B.P., Rindal, P., Shoup, V.: Btx: Simple and efficient batch threshold encryption. Cryptology ePrint Archive (2026)
2. Agarwal, A., Fernando, R., Pinkas, B.: Efficiently-thresholdizable batched identity based encryption, with applications. In: Kalai, Y.T., Kamara, S.F. (eds.) CRYPTO 2025, Part III. LNCS, vol. 16002, pp. 69–100. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 17–21, 2025). https://doi.org/10.1007/978-3-032-01881-6_3
3. 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 Berlin Heidelberg, Germany, French Riviera (May 30 – Jun 3, 2010). https://doi.org/10.1007/978-3-642-13190-5_28
4. 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 Berlin Heidelberg, Germany, Copenhagen, Denmark (May 11–15, 2014). https://doi.org/10.1007/978-3-642-55220-5_30
5. Boneh, D., Laufer, E., Tas, E.N.: Batch decryption without epochs and its application to encrypted mempools. Cryptology ePrint Archive (2025)
6. Choudhuri, A.R., Garg, S., Policharla, G.V., Wang, M.: Practical mempool privacy via one-time setup batched threshold encryption. In: 34th USENIX Security Symposium (USENIX Security 25). pp. 3477–3495 (2025)
7. Cini, V., Lai, R.W.F., Woo, I.K.Y.: Lattice-based multi-authority/client attribute-based encryption for circuits. CiC **1**(4), 1 (2024). https://doi.org/10.62056/ahmpgy4e-

8. Das, S., Datta, P., Partap, A., Sasmal, S., Zhandry, M.: Optimal threshold traitor tracing. Cryptology ePrint Archive, Report 2025/2154 (2025), https://eprint.iacr.org/2025/2154
9. Desmedt, Y., Frankel, Y.: Threshold cryptosystems. In: Brassard, G. (ed.) CRYPTO'89. LNCS, vol. 435, pp. 307–315. Springer, New York, USA, Santa Barbara, CA, USA (Aug 20–24, 1990). https://doi.org/10.1007/0-387-34805-0_28
10. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer Berlin Heidelberg, Germany, Interlaken, Switzerland (May 2–6, 2004). https://doi.org/10.1007/978-3-540-24676-3_31
11. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC. pp. 197–206. ACM Press, Victoria, BC, Canada (May 17–20, 2008). https://doi.org/10.1145/1374376.1374407
12. Goldreich, O.: Foundations of Cryptography: Basic Applications, vol. 2. Cambridge University Press, Cambridge, UK (2004). https://doi.org/10.1017/CBO9780511721656
13. Gong, J., Waters, B., Wee, H., Wu, D.J.: Threshold batched identity-based encryption from pairings in the plain model. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 548–578. Springer (2026)
14. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Attribute-based encryption for circuits. In: Boneh, D., Roughgarden, T., Feigenbaum, J. (eds.) 45th ACM STOC. pp. 545–554. ACM Press, Palo Alto, CA, USA (Jun 1–4, 2013). https://doi.org/10.1145/2488608.2488677
15. Gorbunov, S., Vaikuntanathan, V., Wee, H.: Predicate encryption for circuits from LWE. In: Gennaro, R., Robshaw, M.J.B. (eds.) CRYPTO 2015, Part II. LNCS, vol. 9216, pp. 503–523. Springer Berlin Heidelberg, Germany, Santa Barbara, CA, USA (Aug 16–20, 2015). https://doi.org/10.1007/978-3-662-48000-7_25
16. Goyal, R., Koppula, V., Vusirikala, S., Waters, B.: On perfect correctness in (lockable) obfuscation. In: Pass, R., Pietrzak, K. (eds.) TCC 2020, Part I. LNCS, vol. 12550, pp. 229–259. Springer, Cham, Switzerland, Durham, NC, USA (Nov 16–19, 2020). https://doi.org/10.1007/978-3-030-64375-1_9
17. Goyal, R., Koppula, V., Waters, B.: Lockable obfuscation. In: Umans, C. (ed.) 58th FOCS. pp. 612–621. IEEE Computer Society Press, Berkeley, CA, USA (Oct 15–17, 2017). https://doi.org/10.1109/FOCS.2017.62
18. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Juels, A., Wright, R.N., De Capitani di Vimercati, S. (eds.) ACM CCS 2006. pp. 89–98. ACM Press, Alexandria, Virginia, USA (Oct 30 – Nov 3, 2006). https://doi.org/10.1145/1180405.1180418, available as Cryptology ePrint Archive Report 2006/309
19. Håstad, J., Impagliazzo, R., Levin, L.A., Luby, M.: A pseudorandom generator from any one-way function. SIAM Journal on Computing **28**(4), 1364–1396 (1999)
20. Katsumata, S., Yamada, S.: Partitioning via non-linear polynomial functions: More compact IBEs from ideal lattices and bilinear maps. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part II. LNCS, vol. 10032, pp. 682–712. Springer Berlin Heidelberg, Germany, Hanoi, Vietnam (Dec 4–8, 2016). https://doi.org/10.1007/978-3-662-53890-6_23

21. 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 Berlin Heidelberg, Germany, Istanbul, Turkey (Apr 13–17, 2008). https://doi.org/10.1007/978-3-540-78967-3_9
22. Pedersen, T.P.: A threshold cryptosystem without a trusted party (extended abstract) (rump session). In: Davies, D.W. (ed.) EUROCRYPT'91. LNCS, vol. 547, pp. 522–526. Springer Berlin Heidelberg, Germany, Brighton, UK (Apr 8–11, 1991). https://doi.org/10.1007/3-540-46416-6_47
23. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC. pp. 84–93. ACM Press, Baltimore, MA, USA (May 22–24, 2005). https://doi.org/10.1145/1060590.1060603
24. Sahai, A., Waters, B.R.: Fuzzy identity-based encryption. In: Cramer, R. (ed.) EUROCRYPT 2005. LNCS, vol. 3494, pp. 457–473. Springer Berlin Heidelberg, Germany, Aarhus, Denmark (May 22–26, 2005). https://doi.org/10.1007/11426639_27
25. Sarkar, R., Murshid, S., Mandal, M.: Batched attribute-based encryption from bilinear pairings. Cryptology ePrint Archive (2026)
26. Shamir, A.: How to share a secret. Communications of the Association for Computing Machinery **22**(11), 612–613 (Nov 1979). https://doi.org/10.1145/359168.359176
27. Shen, E., Shi, E., Waters, B.: Predicate privacy in encryption systems. In: Reingold, O. (ed.) TCC 2009. LNCS, vol. 5444, pp. 457–473. Springer Berlin Heidelberg, Germany (Mar 15–17, 2009). https://doi.org/10.1007/978-3-642-00457-5_27
28. Waters, B.: Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization. In: Catalano, D., Fazio, N., Gennaro, R., Nicolosi, A. (eds.) PKC 2011. LNCS, vol. 6571, pp. 53–70. Springer Berlin Heidelberg, Germany, Taormina, Italy (Mar 6–9, 2011). https://doi.org/10.1007/978-3-642-19379-8_4
29. Waters, B., Wu, D.J.: Silent threshold cryptography from pairings: Expressive policies in the plain model. Cryptology ePrint Archive, Report 2025/1547 (2025), https://eprint.iacr.org/2025/1547
30. Wee, H.: Circuit ABE with $\mathsf{poly}(\mathsf{depth}, \lambda)$-sized ciphertexts and keys from lattices. In: Reyzin, L., Stebila, D. (eds.) CRYPTO 2024, Part III. LNCS, vol. 14922, pp. 178–209. Springer, Cham, Switzerland, Santa Barbara, CA, USA (Aug 18–22, 2024). https://doi.org/10.1007/978-3-031-68382-4_6
31. Wee, H.: Almost optimal KP and CP-ABE for circuits from succinct LWE. In: Fehr, S., Fouque, P.A. (eds.) EUROCRYPT 2025, Part III. LNCS, vol. 15603, pp. 34–62. Springer, Cham, Switzerland, Madrid, Spain (May 4–8, 2025). https://doi.org/10.1007/978-3-031-91131-6_2
32. Wichs, D., Zirdelis, G.: Obfuscating compute-and-compare programs under LWE. In: Umans, C. (ed.) 58th FOCS. pp. 600–611. IEEE Computer Society Press, Berkeley, CA, USA (Oct 15–17, 2017). https://doi.org/10.1109/FOCS.2017.61
33. Xiong, S., Zhang, Y., Chen, J.: Efficient batched IBE from lattices in the standard model. IACR Communications in Cryptology **3**(2) (2026). https://doi.org/10.62056/aksude-3