Post-quantum Secure Non-Committing Registered Functional Encryption

Ramprasad Sarkar[0000−0002−9907−3319]

Cryptology and Security Research Unit
R C Bose Centre for Cryptology and Security
Indian Statistical Institute Kolkata, Kolkata, India-700108
rpsarkar123@gmail.com

Abstract. Non-committing encryption (NCE) is a key primitive for proving security against adaptive corruptions, enabling simulators to generate ciphertexts before the encrypted message is known. Existing non-committing constructions for attribute-based encryption primitives [Hiroka *et al.*, ASIACRYPT 2021; Goyal *et al.*, PKC 2025] typically rely on centralized trust that generate users' secret keys. However, modern cryptographic systems increasingly aim to eliminate such trust assumptions through decentralized frameworks such as registered encryption, where users independently generate and register their own keys.

In this work, we initiate the study of *non-committing registered functional encryption* (NC-RFE) as a generalization of non-committing attribute-based encryption in the decentralized framework. We formalize the notion of NC-RFE by adapting classical non-committing security to the registered setting. We then present a generic construction based on indistinguishability obfuscation and zero-knowledge arguments, and prove its security assuming a secure underlying registered functional encryption (RFE) scheme. We further provide a lattice-based instantiation, yielding a post-quantum secure NC-RFE scheme based on the hardness of the plain LWE and equivocal LWE assumptions.

Keywords: Functional Encryption · Non-committing Security · Registered Cryptography · Post-quantum Security

1 Introduction

Modern cryptographic protocols are often deployed in dynamic environments where parties may become corrupted over time. Achieving security in such settings requires strong simulation techniques that allow a simulator to generate protocol transcripts before all inputs are known. A fundamental primitive enabling such simulations is *non-committing encryption* (NCE), introduced by Canetti *et al.* [6]. Informally, an NCE scheme allows a simulator to generate a public key and a ciphertext without committing to the encrypted message. Later, once the message is determined, the simulator can generate suitable explanations, typically in the form of randomness or keys, showing that the same

ciphertext corresponds to an encryption of that message. This flexibility allows protocol transcripts to remain consistent even when the simulator learns new information after earlier messages have already been generated. Over the years, several variants and relaxations of non-committing encryption [4, 8, 9, 21, 26, 31] have been studied. One particularly useful notion is *receiver non-committing encryption* (RNCE) [17], where the simulator only needs to explain the secret key corresponding to a public key, rather than all the randomness used during encryption [17, 22]. This notion has proven sufficient for many applications, including secure multiparty computation [6, 7], adaptive secure attribute-based encryption for Turing Machines [17], selective opening secure schemes [20] and more. This naturally raises the following question:

Can non-committing security be extended to advanced encryption primitives that provide more expressive forms of access control?

Non-committing encryption beyond PKE. Early work on NCE [6,7,9,21,26] mainly focused on standard public-key encryption. In contrast, several widely studied encryption paradigms provide more expressive mechanisms for controlling access to encrypted data, including identity-based encryption (IBE) [28], attribute-based encryption (ABE) [18, 27], and functional encryption (FE) [3]. These primitives allow flexible control over encrypted data: decryption capabilities may depend on identities, attributes, or even functions applied to the encrypted message.

Motivated by this growing importance of expressive encryption, recent works [16, 22] has begun exploring non-committing variants of such primitives [18,27,28]. In 2021, Hiroka *et al.* [22] introduced non-committing attribute-based encryption in the context of quantum encryption with certified deletion. More recently, Goyal *et al.* [16] studied receiver non-committing identity-based encryption (RNC-IBE), providing constructions under standard assumptions and highlighting connections to incompressible cryptography. These works suggest that extending non-committing security beyond classical public-key encryption is both a natural and important research direction.

Decentralization and the registered model. At the same time, another important direction in cryptographic research is the removal of centralized trust assumptions. In most advanced encryption systems [3, 18, 27, 28], a central authority generates and distributes secret keys to users. Although convenient, this approach introduces key-escrow risks and creates a single point of failure.

Registration-based encryption (RBE) [14] addresses the above-mentioned limitations by decentralizing key generation procedure. In the RBE setting, users generate their own key pairs and register only their public keys, while a curator aggregates them into a master public key without holding any private information. This model eliminates the problem of central key escrow while maintaining global encryptability. Building on this foundation, registered attribute-based encryption (RABE) [23] extends decentralization to attribute-controlled settings: users independently register both their keys and attributes, and encryption policies are evaluated relative to the registered attributes.

Towards non-committing security in advanced encryption. Despite progress in both non-committing encryption and decentralized cryptographic frameworks, most existing works on non-committing encryption [6, 7, 9, 16, 21, 22, 26] still rely on centralized setup assumptions. This reliance becomes particularly restrictive when considering highly expressive primitives such as functional encryption. A notable exception is the recent work of Murshid *et al.* [25], which proposes non-committing registered attribute-based encryption in the context of certified deletion without requiring a centralized authority.

Functional encryption (FE) [3] allows users to compute specific functions on encrypted data without revealing the underlying message. In an FE scheme, a trusted authority holds a master secret key and issues function-specific secret keys sk_f to users. Given a ciphertext that encrypts a message m, a user possessing sk_f can only learn the value $f(m)$. Due to its generality, functional encryption subsumes many existing encryption paradigms, including IBE, ABE, and predicate encryption. Recently, Datta *et al.* [12, 13] extended this idea to the decentralized setting by introducing registered functional encryption (RFE). These developments naturally lead to the following motivating question:

Can we achieve non-committing security in a decentralized functional encryption setting where no trusted authority exists?

Combining these two directions introduces several conceptual and technical challenges. In a registered system, users independently generate and register public keys, while encryption depends on a global aggregation of these registrations. Achieving non-committing security requires a simulator capable of generating consistent global parameters, ciphertexts, and registration states without knowing the encrypted message in advance. Later, the simulator must reconcile these objects with a valid set of secret keys corresponding to registered functions. Ensuring such consistency across registration, key generation, and encryption becomes highly non-trivial due to the intricate dependencies between these components.

Our Contribution. In this work, we initiate the study of *non-committing registered functional encryption* (NC-RFE). Our main contributions are summarized below.

– We introduce the notion of *non-committing registered functional encryption* (NC-RFE) and formalize its security definition. Our formulation extends the classical non-committing security framework of [6,7] to the registered encryption setting. Informally, the definition requires that a simulator can generate master public parameters and ciphertexts without knowing the encrypted message in advance, and later produce consistent user secret keys and helper decryption keys when users are adaptively corrupted. At the same time, the definition preserves the fundamental property of registered encryption: honest users independently generate their own keys, and the key curator does not possess any secret information. This notion naturally extends non-committing security to a decentralized functional encryption framework and

provides a useful foundation for building adaptively secure protocols in distributed environments where users manage their own cryptographic keys.

- We present a generic construction of NC-RFE based on indistinguishability obfuscation and zero-knowledge arguments. Our design shows how to achieve non-committing security in the registered functional encryption setting while maintaining decentralized key generation.
- We prove that our construction satisfies non-committing security assuming the security of the underlying RFE scheme, the existence of indistinguishability obfuscation for P/poly, and an interactive zero-knowledge argument system for NP. Our proof demonstrates that the simulator can consistently explain previously generated parameters and ciphertexts even in the presence of adaptive corruptions.
- Finally, we present a lattice-based instantiation of NC-RFE based on the hardness of plain LWE, and Equivocal LWE assumptions. This instantiation provides post-quantum security for the proposed framework and shows that non-committing security can be achieved in decentralized functional encryption systems.

Organisation. The rest of the paper is structured as follows. In Section 2, we provide a high-level technical overview of our construction. Section 3 introduces the required preliminaries. In Section 4, we describe the system model and security definition for NC-RFE, and present its generic construction together with correctness and compactness properties. The security analysis of the proposed NC-RFE scheme is given in Section 5. Section 6 presents a lattice-based instantiation of our construction. Finally, Section 7 concludes the paper.

2 Technical Overview

We first introduce the notation used throughout the paper and then present a high-level technical overview of our constructions for non-committing registered functional encryption.

Notations. The notation $x \leftarrow X$ denotes that x is selected uniformly at random from a finite set X. If A is an algorithm (which may be classical, probabilistic, or quantum), then $y \leftarrow A(x)$ represents the output of A on input x, and we write $y \leftarrow A(x; r)$ when A uses explicit randomness r. Let λ and τ be a security parameter and a policy-family parameter. For a distribution D, we write $x \leftarrow D$ to denote sampling according to D. The assignment $y := z$ indicates that the variable y is set or defined to be equal to z. For a positive integer n, we denote the set $\{1, 2, \ldots, n\}$ by $[n] := \{1, \ldots, n\}$. The symbol λ denotes the security parameter. For classical strings x and y, their concatenation is written as $x \| y$. For a bit string $s \in \{0,1\}^n$, both s_i and $s[i]$ refer to its i-th bit. A function $f : \mathbb{N} \to \mathbb{R}$ is said to be negligible if, for every constant $c > 0$, there exists $\lambda_0 \in \mathbb{N}$ such that $f(\lambda) < \lambda^{-c}$ for all $\lambda > \lambda_0$. We use the notation $f(\lambda) \leq \mathsf{negl}(\lambda)$ to express that f is a negligible function. For a bit string x, we write x_j to mean the j-th bit of x.

Starting Point. As a preliminary step toward our construction, we revisit the notion of non-committing attribute-based encryption (NCABE) [22]. This primitive provides the simulation-based properties necessary to achieve non-committing security within the attribute-based encryption framework. An NCABE is defined over a message space $\mathcal{M}$, an attribute space $\mathcal{X}$, and a policy space $\mathcal{P}$ with the following algorithms.

- **Setup**: Given the security parameter 1^λ, the setup algorithm outputs a master public key mpk and a master secret key msk.
- **KeyGen**: On input of msk and a policy P, the key generation algorithm generates the corresponding secret key sk_P.
- **Encryption**: Using mpk, an attribute X, and a message μ, the encryption algorithm produces a ciphertext ct.
- **Decryption**: The decryption algorithm recovers μ' from ct using sk_P, or outputs $\perp$ in case of failure.
- **FakeSetup**: The fake setup algorithm produces a mpk together with auxiliary information aux, serving as an alternative to the real setup.
- **FakeCT**: Given mpk, aux, and an attribute X, the fake encryption algorithm outputs a fake ciphertext $\widetilde{\mathsf{ct}}$.
- **FakeSK**: The fake key generation algorithm generates a fake secret key $\widetilde{\mathsf{sk}}$ corresponding to policy P.
- **Reveal**: On input of mpk, aux, $\widetilde{\mathsf{ct}}$, and μ, the reveal algorithm outputs a fake master secret key $\widetilde{\mathsf{msk}}$.

These algorithms allow the simulator to generate protocol transcripts without committing to plaintext during encryption.

First Step: Towards NCFE from NCABE. Our first step is to extend the NCABE framework to support non-committing functional encryption (NCFE). Because functional encryption itself generalizes attribute-based encryption, the syntax of NCFE can be viewed as a natural extension of the algorithms used in NCABE. The key difference lies in the structure of the keys and the decryption functionality. In particular, the following changes are required:

- **KeyGen**: On input of msk and a function f, the key generation algorithm generates the corresponding secret key sk_f.
- **Encryption**: Using mpk, and a message m, the encryption algorithm produces a ciphertext ct.
- **Decryption**: The decryption algorithm recovers $f(m)$ from ct using sk_f.
- **FakeCT**: Given mpk, aux, the fake encryption algorithm outputs a fake ciphertext $\widetilde{\mathsf{ct}}$.
- **FakeSK**: The fake key generation algorithm generates a fake secret key $\widetilde{\mathsf{sk}}$ corresponding to function f.

Second Step: Towards NC-RFE from NCFE. Having established NCFE as an intermediate building block, we next extend the NCFE to the registered setting. In the NCFE, a trusted authority holds the master secret key msk and is responsible for generating user secret keys. However, the registered model imposes a crucial restriction: the curator responsible for managing registrations must remain transparent and publicly verifiable. Consequently, the curator cannot hold or access any secret information. This restriction makes a direct adaptation of NCFE to the registered setting infeasible. The main challenge for this extended framework, non-committing registered functional encryption (NC-RFE), is therefore to redesign the NCFE framework so that the functionality previously provided by the trusted authority can be simulated without relying on any hidden secret state.

Third Step: Architecture of NC-RFE. To overcome this challenge, we can integrate simulation procedures directly into the registered functional encryption (RFE) framework. In this perspective, we can be viewed as an RFE scheme augmented with additional algorithms that enable non-committing simulation. These simulation procedures are used only within the security experiment and therefore do not affect the correctness of the scheme. The resulting primitive, which we call non-committing registered functional encryption (NC-RFE), introduces several additional algorithms that allow the simulator to generate public keys, simulate registrations, produce ciphertexts, and later reconcile them with appropriate secret keys. At a high level, the simulator maintains auxiliary state information that records the simulated registration history. Using this state, it can generate ciphertexts that are independent of the underlying message. When the message is eventually revealed, the simulator produces secret keys that ensure the simulated ciphertext is consistently decrypted. The NC-RFE framework therefore augments the standard RFE algorithms with the following five simulation algorithms.

- **SimKeyGen**: Given crs, aux, and an ancillary dictionary B (possibly empty) indexed by public keys, the simulator runs the key-generation procedure to produce a simulated public key $\widetilde{\mathsf{pk}}$ along with an updated B.
- **SimRegPK**: The simulated registration procedure takes as input crs, aux, pk, and f, and outputs a simulated master public key $\widetilde{\mathsf{mpk}}$ together with an updated auxiliary state $\widetilde{\mathsf{aux}}$.
- **SimCorrupt**: The simulated corrupt algorithm, on input crs, pk, and B, returns a simulated secret key $\widetilde{\mathsf{sk}}$ corresponding to pk.
- **SimCT**: Given $\widetilde{\mathsf{mpk}}$, aux, and B, the ciphertext simulation algorithm produces a simulated ciphertext $\widetilde{\mathsf{ct}}$.
- **Reveal**: On input of $\widetilde{\mathsf{pk}}$, B, $\widetilde{\mathsf{ct}}$, and a target message m, the reveal algorithm outputs a simulated secret key $\widetilde{\mathsf{sk}}$ that decrypts $\widetilde{\mathsf{ct}}$ to μ.

Now, we need to argue for the security of NC-RFE, which we discuss in the following.

Security for NC-RFE. The security foundation of NC-RFE builds upon a receiver non-committing security framework that combines indistinguishability obfuscation with zero-knowledge arguments possessing statistical soundness and computational zero-knowledge properties. The security analysis employs a sequence of hybrid transformations that systematically transition from real-world operations to ideal-world simulations, ensuring that no polynomial-time adversary can distinguish between these worlds with non-negligible probability.

The proof methodology begins with the original NC-RFE security experiment, where all cryptographic operations are performed according to their original algorithms. The analysis then progressively introduces simulation components through computationally indistinguishable transformations. The first transformation replaces left-or-right decryption circuits embedded within obfuscated programs with left-only variants, thereby eliminating the scheme's dependence on the random selector string z, which determines the bit representation to use for each message position. This change relies on the security properties of indistinguishability obfuscation, which ensures that functionally equivalent circuits remain computationally indistinguishable when obfuscated.

Next, to integrate zero-knowledge arguments within the registered framework, we consider constructions that rely on a common reference string crs. In such settings, each new user registration would necessitate an update of the crs. Our initial attempt was to realize NC-RFE using existing NIZK schemes with updatable common reference strings [1, 19]. However, these schemes either fail to achieve statistical soundness or depend on pairing-based assumptions that are potentially insecure against quantum polynomial-time (QPT) adversaries.

To overcome these limitations, we employ a zero-knowledge argument system that does not require any common reference string, while still guaranteeing soundness and zero-knowledge properties against QPT adversaries. The analysis transitions to simulated ZKA components, where challenge proofs are generated using simulation algorithms: $\widetilde{\pi} \leftarrow \langle Sim \rangle(d^*, y)$, where d^*, y are the statement and a polynomial-length bit string over $\{0,1\}$, respectively. This transformation leverages the computational zero-knowledge property of the ZKA protocol to ensure that the simulated proofs are indistinguishable from the real ones. This makes the proof components independent of the actual challenge message.

Another hybrid transformation introduces asymmetric encryption patterns that break the symmetry in how different message bits are encrypted. For each bit position i, the construction encrypts different values depending on the selector bit $z[i]$: $\mathsf{ct}^*_{i,z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z[i]}, m[i])$ encrypts the actual message bit, while $\mathsf{ct}^*_{i,1-z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,1-z[i]}, 1 - m[i])$ encrypts its complement, where z, m are a polynomial-length bitstring over $\{0,1\}$ and a target message, respectively. This asymmetric approach is validated through a position-by-position hybrid argument that relies on the semantic security of the underlying RFE scheme.

Finally, the analysis achieves complete independence from the target message m employing a new random string $z^* \leftarrow \{0,1\}^{\ell_m}$ and encrypting fixed values: $\mathsf{ct}^*_{i,z^*[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z^*[i]}, 0)$ and $\mathsf{ct}^*_{i,1-z^*[i]} \leftarrow \mathsf{RFE.Encrypt}\ (\widetilde{\mathsf{mpk}}_{i,1-z^*[i]}, 1)$. This systematic elimination of dependencies, from the random selector string z, through real ZKA proofs, to the target message m itself, combined with the computational indistinguishability of consecutive transformations, establishes that no polynomial-time adversary can distinguish between real and ideal executions with non-negligible probability.

3 Preliminaries

We briefly recall the cryptographic preliminaries that are required in our construction.

3.1 NP Languages

In computational complexity theory, NP represents a non-deterministic polynomial time machine [11, 29] (*i.e.,* a machine that can move more than one state from a given configuration). A language $\mathcal{L}$ over an alphabet Σ is called an NP languages iff there exists a $k \in \mathbb{N}$ and a polynomial checking relation $R_{\mathcal{L}} \subseteq \Sigma^* \times \Sigma^*$ such that for all strings $x \in \Sigma^*$, $x \in \mathcal{L} \iff \exists y\ (|y| \leq |x|^k$ and $R_{\mathcal{L}}(x, y))$, where $|x|$ and $|y|$ are the lengths of the strings x and y, respectively.

3.2 Indistinguishability Obfuscation

Let $\{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a family of circuits. A PPT algorithm $i\mathcal{O}$ is an indistinguishability obfuscator for the circuit class if it satisfies the following:

- **Correctness:** For every security parameter $\lambda \in \mathbb{N}$, every circuit $C \in \mathcal{C}_\lambda$, and every input x, the obfuscated circuit preserves functionality:

$$\Pr[C'(x) = C(x) \mid C' \leftarrow i\mathcal{O}(C)] = 1.$$

- **Indistinguishability:** For any QPT distinguisher $\mathcal{D}$ and any pair of functionally equivalent circuits $C_0, C_1 \in \mathcal{C}_\lambda$ such that $C_0(x) = C_1(x)$ for all inputs x and $|C_0| = |C_1|$, the obfuscations are computationally indistinguishable:

$$\big|\Pr[\mathcal{D}(i\mathcal{O}(C_0)) = 1] - \Pr[\mathcal{D}(i\mathcal{O}(C_1)) = 1]\big| \leq \mathsf{negl}(\lambda).$$

In recent literature, candidates for indistinguishability obfuscation secure against QPT adversaries have been proposed [5, 15, 30]. Moreover, the work of [10] proposes a post-quantum $i\mathcal{O}$ candidate based on the NTRU assumption and a novel Equivocal LWE problem, employing a trapdoor-based Gaussian hint-release mechanism that ensures correctness while preventing structural leakage, with security proven against QPT adversaries.

3.3 Zero-Knowledge Arguments

We define a zero-knowledge argument system (ZKA) as an interactive protocol between two parties. Let $\langle A(w), B(z)\rangle(x)$ denote the interaction between interactive machines A and B on a common input x, where A receives auxiliary input w and B receives auxiliary input z. We refer to the protocol as $\prod_{\mathsf{ZKA}} = \langle A, B\rangle$.

Definition 1 (Classical Proof and Argument Systems for NP). *Let (P, V) be a protocol with an honest PPT prover P and an honest PPT verifier V for a language $\mathcal{L} \in \mathsf{NP}$, satisfying:*

Perfect Completeness: *For any $x \in \mathcal{L}$, $R_{\mathcal{L}}(x, w)$ and $z \in \{0,1\}^*$,*

$$\Pr[\langle P(w), V(z)\rangle(x) = 1] = 1.$$

Soundness: *For any quantum polynomial-size prover $P^* = \{P^*_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$, there exists a negligible function $\mu(\cdot)$ such that for any security parameter $\lambda \in \mathbb{N}$ and any $x \in \{0,1\}^\lambda \setminus \mathcal{L}$ and $z \in \{0,1\}^*$, $\Pr[\langle P^*_\lambda(\rho_\lambda), V(z)\rangle(x) = 1] \leq \mu(\lambda)$. A protocol with computational soundness is called an* argument.

Zero Knowledge: *There exists a quantum polynomial-time simulator* Sim *such that for any quantum polynomial-size verifier $V^* = \{V^*_\lambda, \rho_\lambda\}_{\lambda\in\mathbb{N}}$,*

$$\left\{\langle P(w), V^*_\lambda(\rho_\lambda)\rangle(x)\right\}_{\lambda,x,w} \overset{c}{\approx} \left\{\mathsf{Sim}(x, V^*_\lambda, \rho_\lambda)\right\}_{\lambda,x},$$

where $\lambda \in \mathbb{N}$, $x \in \mathcal{L} \cap \{0,1\}^\lambda$, and $R_{\mathcal{L}}(x, w)$. If V^* is a classical circuit, then the simulator is computable by a classical polynomial-time algorithm.

$$\{\langle P(w), V^*(z)\rangle(x)\}_{x,w,z} \overset{c}{\approx} \{\langle \mathsf{Sim}\rangle(x, z)\}_{x,z}$$

when the distinguishing gap is considered as a function of $|x|$.

If the LWE assumption holds against quantum polynomial-size adversaries, then there exists a post-quantum zero-knowledge interactive argument system $\langle P, V\rangle$ for all languages in NP [2].

3.4 Registered Functional Encryption

We now recall the notion of Registered Functional Encryption (RFE) [12, 13]. Let $\mathcal{U}_F = \{F_\lambda\}_{\lambda\in\mathbb{N}}$ denote a family of function classes indexed by the security parameter, and let $\mathcal{M}$ be the associated message space. A registered functional encryption scheme for function universe $\mathcal{U}_F$ and message space $\mathcal{M}$ consists of a collection of efficient algorithms $\mathsf{RFE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{RegPK}, \mathsf{Encrypt}, \mathsf{Update}, \mathsf{Dec})$, which operate as described below.

$\mathsf{Setup}(1^\lambda, 1^{\ell_f})$**:** On input the security parameter λ and an upper bound ℓ_f on the description length of functions in $\mathcal{U}_F$, this algorithm generates and outputs a common reference string crs.

$\mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux})$**:** Given the common reference string crs and auxiliary state information aux (which may be empty), this algorithm outputs a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f_{\mathsf{pk}})$**:** Given the reference string crs, the current auxiliary state aux, a public key pk, and an associated function $f_{\mathsf{pk}} \in F_\lambda$, this deterministic algorithm outputs an updated master public key mpk together with updated auxiliary state aux'.

$\mathsf{Encrypt}(\mathsf{mpk}, m)$**:** On input a master public key mpk and a message $m \in \mathcal{M}$, this algorithm produces a ciphertext ct.

$\mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk})$**:** Given the reference string crs, auxiliary state aux, and a public key pk, this deterministic procedure outputs a helper decryption key hsk.

$\mathsf{Dec}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct})$: On input a secret key sk, a helper decryption key hsk, and a ciphertext ct, this deterministic algorithm outputs a message m', the failure symbol $\perp$, or the instruction $\mathsf{GetUpdate}$ indicating that an updated helper secret key is required.

The algorithms are required to satisfy the following properties.

Correctness and Efficiency. For every security parameter $\lambda \in \mathbb{N}$, every message $m \in \mathcal{M}$, and every function $f \in F_\lambda$, consider the following experiment between an adversary $\mathcal{A}$ and a challenger.

- **Setup phase.** The challenger first samples a common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_f})$. It initializes the auxiliary state $\mathsf{aux} \leftarrow \perp$ and the initial master public key $\mathsf{mpk}_0 \leftarrow \perp$. Furthermore, it initializes the counters $\mathsf{ctr}[\mathsf{reg}] \leftarrow 0, \mathsf{ctr}[\mathsf{enc}] \leftarrow 0$, which track the number of registration and encryption queries, respectively. The challenger also sets $\mathsf{ctr}[\mathsf{reg}]^* \leftarrow \infty$ to denote the index of the target key. Finally, the challenger sends crs to $\mathcal{A}$.
- **Query phase.** During this phase, $\mathcal{A}$ may issue the following queries.
  - **Register non-target key query.** $\mathcal{A}$ submits a public key pk and a function $f \in \mathcal{U}_F$. The challenger increments the counter $\mathsf{ctr}[\mathsf{reg}] \leftarrow \mathsf{ctr}[\mathsf{reg}] + 1$, and registers the key by computing $(\mathsf{mpk}_{\mathsf{ctr}[\mathsf{reg}]}, \mathsf{aux}') \leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f)$. The auxiliary state is updated as $\mathsf{aux} \leftarrow \mathsf{aux}'$, and the challenger returns $(\mathsf{ctr}[\mathsf{reg}], \mathsf{mpk}_{\mathsf{ctr}[\mathsf{reg}]}, \mathsf{aux})$ to $\mathcal{A}$.
  - **Register target key query.** $\mathcal{A}$ specifies a function $f^* \in \mathcal{U}_F$. If a target-key registration query was previously issued, the challenger outputs $\perp$. Otherwise, the challenger increment the registration counter $\mathsf{ctr}[\mathsf{reg}] \leftarrow \mathsf{ctr}[\mathsf{reg}]+1$. It sample a key pair $(\mathsf{pk}^*, \mathsf{sk}^*) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux})$, and register the key $(\mathsf{mpk}_{\mathsf{ctr}[\mathsf{reg}]}, \mathsf{aux}') \leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*, f^*)$. It then compute the helper decryption key $\mathsf{hsk}^* \leftarrow \mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*)$. The challenger updates $\mathsf{aux} \leftarrow \mathsf{aux}'$ and records the target index $\mathsf{ctr}[\mathsf{reg}]^* \leftarrow \mathsf{ctr}[\mathsf{reg}]$. Finally it returns $(\mathsf{ctr}[\mathsf{reg}], \mathsf{mpk}_{\mathsf{ctr}[\mathsf{reg}]}, \mathsf{aux}, \mathsf{pk}^*, \mathsf{hsk}^*, \mathsf{sk}^*)$ to $\mathcal{A}$.
  - **Encryption query.** $\mathcal{A}$ submits an index $\mathsf{ctr}[\mathsf{reg}]^* \leq i \leq \mathsf{ctr}[\mathsf{reg}]$ together with a message $m_{\mathsf{ctr}[\mathsf{enc}]} \in \mathcal{M}$. If the target key has not yet

been registered, the challenger returns $\perp$. Otherwise it increments the encryption counter $\mathsf{ctr}[\mathsf{enc}] \leftarrow \mathsf{ctr}[\mathsf{enc}] + 1$ and computes $\mathsf{ct}_{\mathsf{ctr}[\mathsf{enc}]} \leftarrow \mathsf{Encrypt}(\mathsf{mpk}_i, m)$. The challenger replies with $(\mathsf{ctr}[\mathsf{enc}], \mathsf{ct}_{\mathsf{ctr}[\mathsf{enc}]})$.

– **Decryption query.** $\mathcal{A}$ provides a ciphertext index $1 \leq j \leq \mathsf{ctr}[\mathsf{enc}]$. The challenger computes $m'_j \leftarrow \mathsf{Dec}(\mathsf{sk}^*, \mathsf{hsk}^*, \mathsf{ct}_j)$. If $m'_j = \mathsf{GetUpdate}$, the challenger updates the helper key $\mathsf{hsk}^* \leftarrow \mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}^*)$, and recomputes the decryption. If $m'_j \neq f^*(m_j)$, the experiment terminates with output $b = 1$.

If $\mathcal{A}$ completes all queries and the experiment has not halted, the experiment outputs $b = 0$.

The scheme RFE is said to be correct and efficient if every adversary making a polynomial number of queries satisfies the following properties.

- **Correctness.** There exists a negligible function $\mathsf{negl}(\cdot)$ such that $\Pr[b = 1] = \mathsf{negl}(\lambda)$. The scheme achieves *perfect correctness* if $\Pr[b = 1] = 0$.
- **Compactness.** Let N denote the number of registration queries. There exists a polynomial $\mathrm{poly}(\cdot)$ such that for every $i \in$, $|\mathsf{mpk}_i| = \mathrm{poly}(\lambda, \ell_f, \log i)$. Moreover the helper key size satisfies $|\mathsf{hsk}^*| = \mathrm{poly}(\lambda, \ell_f, \log N)$.
- **Update efficiency.** Let N be the number of registrations. Throughout the experiment the challenger invokes Update at most $O(\log N)$ times, and each invocation runs in time $\mathrm{poly}(\log N)$ in the RAM model.

Security. Let $b \in \{0, 1\}$ be a bit. We define the following security experiment $\mathsf{Exp}_A^{\mathsf{RFE}}(1^\lambda, b)$ between an adversary $\mathcal{A}$ and a challenger.

- **Setup phase.** The challenger samples a common reference string $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_f})$. It initializes the auxiliary state $\mathsf{aux} \leftarrow \perp$ and the master public key $\mathsf{mpk} \leftarrow \perp$. It also sets a counter $\mathsf{ctr} \leftarrow 0$ that records the number of honest-key registration queries made by $\mathcal{A}$. Furthermore, it initializes an empty set of corrupted keys $C \leftarrow \emptyset$ and an empty dictionary $\mathsf{D} \leftarrow \emptyset$ that maps public keys to the functions registered for them. For convenience, if a public key $\mathsf{pk} \notin \mathsf{D}$ then we define $\mathsf{D}[\mathsf{pk}] := \emptyset$. Finally, the challenger gives crs to $\mathcal{A}$.
- **Query phase.** During this phase, the adversary $\mathcal{A}$ may issue the following queries.
  – **Registered corrupted key query.** $\mathcal{A}$ specifies a public key pk and a function $f \in \mathcal{U}_F$. The challenger registers this key by computing $(\mathsf{mpk}', \mathsf{aux}') \leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f)$. It then updates its internal state by setting $\mathsf{mpk} \leftarrow \mathsf{mpk}'$ and $\mathsf{aux} \leftarrow \mathsf{aux}'$, adds pk to the corrupted set C, and records the function by updating $\mathsf{D}[\mathsf{pk}] \leftarrow \mathsf{D}[\mathsf{pk}] \cup \{f\}$. Finally, it returns $(\mathsf{mpk}', \mathsf{aux}')$ to $\mathcal{A}$.
  – **Registered honest key query.** $\mathcal{A}$ submits a function $f \in \mathcal{U}_F$. The challenger first increments the counter $\mathsf{ctr} \leftarrow \mathsf{ctr}+1$ and generates a fresh key pair $(\mathsf{pk}_{\mathsf{ctr}}, \mathsf{sk}_{\mathsf{ctr}}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux})$. It then registers the public key

by computing $(\mathsf{mpk}', \mathsf{aux}') \leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}_{\mathsf{ctr}}, f)$. The challenger updates $\mathsf{mpk} \leftarrow \mathsf{mpk}'$ and $\mathsf{aux} \leftarrow \mathsf{aux}'$, records the function via $\mathsf{D}[\mathsf{pk}_{\mathsf{ctr}}] \leftarrow \mathsf{D}[\mathsf{pk}_{\mathsf{ctr}}] \cup \{f\}$, and returns $(\mathsf{ctr}, \mathsf{mpk}', \mathsf{aux}', \mathsf{pk}_{\mathsf{ctr}})$ to $\mathcal{A}$.

  - **Corrupt honest query.** $\mathcal{A}$ specifies an index $1 \leq i \leq \mathsf{ctr}$. Let $(\mathsf{pk}_i, \mathsf{sk}_i)$ denote the key pair generated in the i-th honest-key registration query. The challenger adds pk_i to the corrupted set C and returns the secret key sk_i to $\mathcal{A}$.

- **Challenge phase.** $\mathcal{A}$ outputs two challenge messages $m_0^*, m_1^* \in \mathcal{M}$. The challenger responds with the challenge ciphertext $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, m_b^*)$.
- **Output phase.** Finally, $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$, which is taken as the output of the experiment.

Let $S = \{ f_{\mathsf{pk}} \in \mathsf{D}[\mathsf{pk}] : \mathsf{pk} \in C \}$. We say that an adversary $\mathcal{A}$ is *admissible* if for every corrupted slot function $f_{\mathsf{pk}} \in S$, it holds that $f_{\mathsf{pk}}(m_0^*) = f_{\mathsf{pk}}(m_1^*)$.

The registered functional encryption scheme RFE is considered secure if for every admissible adversary $\mathcal{A}$ there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$, $|\Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathsf{RFE}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Exp}_{\mathcal{A}}^{\mathsf{RFE}}(1^\lambda, 1) = 1]| = \mathsf{negl}(\lambda)$.

4 Non-Committing Registered Functional Encryption

In this section, we first introduce the system model for *non-committing registered functional encryption* (NC-RFE) and formalize its security definition through a security game. We then present a generic construction of our NC-RFE scheme by modularly combining registered functional encryption, zero-knowledge arguments, and indistinguishability obfuscation. Finally, we discuss the correctness of our construction as well as its compactness and update efficiency properties.

4.1 System Model of NC-RFE

A NC-RFE scheme with function universe $\mathcal{U}_F$, and message space $\mathcal{M}$ is a tuple of efficient algorithms $\prod_{\mathsf{NC\text{-}RFE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{RegPK}, \mathsf{Encrypt}, \mathsf{Update}, \mathsf{Decrypt}, \mathsf{SimKeyGen}, \mathsf{SimRegPK}, \mathsf{SimCT}, \mathsf{SimCorrupt}, \mathsf{Reveal})$ with the following syntax:

- $(\mathsf{crs}) \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_f})$. On input the security parameter λ and an upper bound ℓ_f on the description length of functions in $\mathcal{U}_F$, this algorithm generates and outputs a common reference string crs.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux})$. Given the common reference string crs and auxiliary state information aux (which may be empty), this algorithm outputs a public/secret key pair $(\mathsf{pk}, \mathsf{sk})$.
- $(\mathsf{mpk}, \mathsf{aux}') \leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f_{\mathsf{pk}})$. Given the reference string crs, the current auxiliary state aux, a public key pk, and an associated function $f_{\mathsf{pk}} \in F_\lambda$, this deterministic algorithm outputs an updated master public key mpk together with updated auxiliary state aux'.

- $(\mathsf{ct}) \leftarrow \mathsf{Encrypt}(\mathsf{mpk}, m)$. On input a master public key mpk and a message $m \in \mathcal{M}$, this algorithm produces a ciphertext ct.
- $(\mathsf{hsk}) \leftarrow \mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk})$. This algorithm outputs a helper secret key hsk given crs, aux, and pk as inputs.
- $(m') \leftarrow \mathsf{Decrypt}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct})$. On input a secret key sk, a helper decryption key hsk, and a ciphertext ct, this deterministic algorithm outputs a message m'.
- $(\mathsf{pk}, \mathsf{B}) \leftarrow \mathsf{SimKeyGen}(\mathsf{crs}, \mathsf{aux}, \mathsf{B})$. Given crs, aux, and an ancillary dictionary B (possibly empty) indexed by public keys, the simulated key generation algorithm outputs pk together with an updated ancillary dictionary B.
- $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{aux}}) \leftarrow \mathsf{SimRegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f_{\mathsf{pk}})$. The simulated key registration algorithm, given crs, aux, pk, and f_{pk}, produces a simulated master public key $\widetilde{\mathsf{mpk}}$ and an updated auxiliary state $\widetilde{\mathsf{aux}}$.
- $(\widetilde{\mathsf{sk}}) \leftarrow \mathsf{SimCorrupt}(\mathsf{crs}, \mathsf{pk}, \mathsf{B})$. The simulated corruption algorithm takes as input crs, pk and B to return a simulated secret key $\widetilde{\mathsf{sk}}$ corresponding to pk.
- $(\widetilde{\mathsf{ct}}) \leftarrow \mathsf{SimCT}(\widetilde{\mathsf{mpk}}, \mathsf{aux}, \mathsf{B})$. The simulated ciphertext generation algorithm simulates the entire encryption algorithm using the simulated master public key $\widetilde{\mathsf{mpk}}$, aux, and B, and outputs a simulated ciphertext $\widetilde{\mathsf{ct}}$.
- $(\widetilde{\mathsf{sk}}) \leftarrow \mathsf{Reveal}(\mathsf{pk}, \mathsf{B}, \widetilde{\mathsf{ct}}, m)$. Given pk, B, $\widetilde{\mathsf{ct}}$, and a target message m, this algorithm generates a simulated secret key $\widetilde{\mathsf{sk}}$.

Correctness and Update Efficiency of NC-RFE. The NC-RFE protocol preserves the same correctness and efficiency guarantees as the underlying RFE scheme.

Compactness: Let N denote the number of registration queries. There exists a universal polynomial $\mathsf{poly}(\cdot,\cdot,\cdot)$ such that for all $i \in [N]$,

$$|\mathsf{mpk}_i|, |\widetilde{\mathsf{mpk}_i}| = \mathsf{poly}(\lambda, |\mathcal{U}_F|, \log i), \quad |\mathsf{hsk}^*| = \mathsf{poly}(\lambda, |\mathcal{U}_F|, \log N).$$

4.2 Security Model for NC-RFE

Let a non-committing registered functional encryption scheme $\prod_{\mathsf{NC\text{-}RFE}} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{RegPK}, \mathsf{Encrypt}, \mathsf{Update}, \mathsf{Decrypt}, \mathsf{SimKeyGen}, \mathsf{SimRegPK}, \mathsf{SimCT}, \mathsf{SimCorrupt}, \mathsf{Reveal})$ be defined over a function universe $\mathcal{U}_F$ and message space $\mathcal{M}$. For a security parameter λ, a bit $b \in \{0,1\}$, and an adversary $\mathcal{A}$, we define the following security game $\mathsf{Exp}^{\mathsf{NC\text{-}RFE}}_{\mathcal{A}}(\lambda, b)$ between $\mathcal{A}$ and a challenger:

- **Setup.** The challenger begins by generating $\mathsf{crs} \leftarrow \mathsf{Setup}(1^\lambda, 1^{\ell_f})$. It initializes its internal variables as follows: $\mathsf{aux} \leftarrow \perp$, $\mathsf{mpk} \leftarrow \perp$, and a counter $\mathsf{ctr} \leftarrow 0$ to track the number of honestly registered keys. Additionally, two empty sets $C \leftarrow \emptyset$ and $H \leftarrow \emptyset$ are initialized to store corrupted and honest public keys, respectively. The challenger also initializes empty dictionaries, $\mathsf{D} \leftarrow \emptyset$ and $\mathsf{B} \leftarrow \emptyset$, indexed by the public keys. For ease of notation, it is assumed that if $\mathsf{pk} \notin \mathsf{D}$, then $\mathsf{D}[\mathsf{pk}] := \emptyset$ and same for B. Finally, the challenger sends crs to $\mathcal{A}$.

- **Query Phase:** During this phase, $\mathcal{A}$ is allowed to make the following types of queries:
  - **Register corrupted key query.** Upon receiving pk and $f \in \mathcal{U}_F$ from $\mathcal{A}$, the challenger proceeds as follows:

$$(\mathsf{mpk}', \mathsf{aux}') \leftarrow \begin{cases} \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f), & \text{if } b = 0 \\ \mathsf{SimRegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f), & \text{if } b = 1 \end{cases}$$

    It then updates its internal state as:

$$\mathsf{mpk} \leftarrow \mathsf{mpk}', \mathsf{aux} \leftarrow \mathsf{aux}', C \leftarrow C \cup \{\mathsf{pk}\}, \mathsf{D}[\mathsf{pk}] \leftarrow \mathsf{D}[\mathsf{pk}] \cup \{f\}.$$

    The challenger returns $(\mathsf{mpk}', \mathsf{aux}')$ to $\mathcal{A}$.
  - **Register honest key query:** When $\mathcal{A}$ specifying a function $f \in \mathcal{U}_F$, the challenger increments its counter $\mathsf{ctr} \leftarrow \mathsf{ctr} + 1$, and performs the following based on the bit b:

$$\left.\begin{aligned} (\mathsf{pk}_{\mathsf{ctr}}, \mathsf{sk}_{\mathsf{ctr}}) &\leftarrow \mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux}) \\ (\mathsf{mpk}', \mathsf{aux}') &\leftarrow \mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}_{\mathsf{ctr}}, f) \end{aligned}\right\} \quad \text{if } \ b = 0$$

$$\left.\begin{aligned} (\mathsf{pk}_{\mathsf{ctr}}, \mathsf{B}') &\leftarrow \mathsf{SimKeyGen}(\mathsf{crs}, \mathsf{aux}, \mathsf{B}) \\ (\mathsf{mpk}', \mathsf{aux}') &\leftarrow \mathsf{SimRegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}_{\mathsf{ctr}}, f) \end{aligned}\right\} \quad \text{if } \ b = 1.$$

    It then updates:

$$\begin{aligned} &\mathsf{mpk} \leftarrow \mathsf{mpk}', \mathsf{aux} \leftarrow \mathsf{aux}', H \leftarrow H \cup \{\mathsf{pk}_{\mathsf{ctr}}\}, \\ &\mathsf{D}[\mathsf{pk}_{\mathsf{ctr}}] \leftarrow \mathsf{D}[\mathsf{pk}_{\mathsf{ctr}}] \cup \{f\}, B \leftarrow B'. \end{aligned}$$

    It then replies with $(\mathsf{ctr}, \mathsf{mpk}', \mathsf{aux}', \mathsf{pk}_{\mathsf{ctr}})$.
  - **Corrupt honest key query:** $\mathcal{A}$ may request the corruption of an honest key by specifying an index i such that $1 \leq i \leq \mathsf{ctr}$. The challenger responds based on the bit b:

    (a) If $b = 0$, it returns the actual secret key sk_i.

    (b) If $b = 1$, it computes a simulated secret key $\widetilde{\mathsf{sk}}_i \leftarrow \mathsf{SimCorrupt}(\mathsf{crs}, \mathsf{pk}_i, \mathsf{B})$.

    In either case, pk_i is removed from H and added to C.
- **Challenge phase:** $\mathcal{A}$ selects $m \in \mathcal{M}$ subject to the following restriction that for all $f \in S$, $f(m)$ is fixed and consistent with the adversary's view, where $S = \{f \in \mathsf{D}[\mathsf{pk}] \mid \mathsf{pk} \in C\}$. The challenger then prepares the challenge ciphertext as follows:

$$\mathsf{ct}^* \leftarrow \begin{cases} \mathsf{Encrypt}(\mathsf{mpk}, m) & \text{if } b = 0, \\ \mathsf{SimCT}(\mathsf{mpk}, \mathsf{aux}, \mathsf{B}) & \text{if } b = 1. \end{cases}$$

For every honest public key $\mathsf{pk}_i \in H$, the challenger generates the corresponding secret and helper keys as follows:

$$\mathsf{sk}_i^* \leftarrow \begin{cases} \mathsf{sk}_i & \text{if } b = 0, \\ \mathsf{Reveal}(\mathsf{pk}_i, \mathsf{B}, \mathsf{ct}^*, m) & \text{if } b = 1, \end{cases}$$
$$\mathsf{hsk}_i \leftarrow \mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}_i).$$

The challenger then returns to $\mathcal{A}$ the tuple: $(\mathsf{ct}^*, \{\mathsf{sk}_i^*\}_{\mathsf{pk}_i \in H}, \{\mathsf{hsk}_i\}_{\mathsf{pk}_i \in H})$. The post-challenge query is redundant beacuse the adversary already has maximum information after the challenge phase.

- **Output phase:** $\mathcal{A}$ finally outputs a bit $b' \in \{0, 1\}$.

A NC-RFE scheme is said to be secure if for every efficient quantum polynomial-time (QPT) adversary $\mathcal{A}$, there exists a negligible function $\mathsf{negl}(\cdot)$ such that for all $\lambda \in \mathbb{N}$,

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

4.3 Generic Construction of NC-RFE Protocol

In this section, we construct NC-RFE scheme with message space $\{0,1\}^{\ell_m}$, and function universe $\mathcal{U}_F$, where ℓ_m are some polynomials. Our construction builds upon the following cryptographic components:

- A registered functional encryption scheme with the tuple of algorithms $\prod_{\mathsf{RFE}} = \mathsf{RFE}.(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{RegPK}, \mathsf{Encrypt}, \mathsf{Update}, \mathsf{Decrypt})$ defined over the same $\mathcal{U}_F$.
- A zero-knowledge argument given by the interactive machines $\langle \mathcal{P}, \mathcal{V} \rangle$ for an NP language $\mathcal{L}$ characterized by the relation R defined as:

$$R := \Big\{ \big((\{\mathsf{mpk}_{i,0}, \mathsf{mpk}_{i,1}\}_{i \in [\ell_m]}, \{\mathsf{ct}_{i,0}, \mathsf{ct}_{i,1}\}_{i \in [\ell_m]}), \{(m[i], r_{i,0}, r_{i,1})\}_{i \in [\ell_m]} \big) \\ : \forall i \in [\ell_m],\ \forall b \in \{0,1\},\ \mathsf{ct}_{i,b} = \mathsf{RFE.Encrypt}(\mathsf{mpk}_{i,b}, m[i]; r_{i,b}) \Big\}.$$

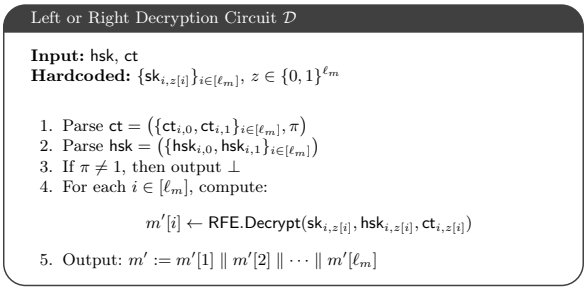
  Each $r_{i,b}$ is sampled uniformly from the randomness space of the RFE.Encrypt algorithm.
- An indistinguishability obfuscator $i\mathcal{O}$ for polynomial-size circuits.

The algorithms of NC-RFE are detailed below.

$\mathsf{Setup}(1^\lambda, 1^{\ell_f})$:
- For every $i \in [\ell_m]$ and $b \in \{0,1\}$, generate $\mathsf{crs}_{i,b} \leftarrow \mathsf{RFE.Setup}(1^\lambda, 1^{\ell_f})$.
- Sample $y \leftarrow \{0,1\}^{\ell_m}$ uniformly at random.
- Output $\mathsf{crs} := (\{\mathsf{crs}_{i,b}\}_{i \in [\ell_m],\ b \in \{0,1\}}, y)$.

$\mathsf{KeyGen}(\mathsf{crs}, \mathsf{aux})$:
- Parse $\mathsf{crs} = (\{\mathsf{crs}_{i,b}\}_{i,b}, y)$.

Fig. 1: Left or Right Decryption Circuit



- Sample $z \leftarrow \{0,1\}^{\ell_m}$ uniformly at random.
- For all $i \in [\ell_m]$, $b \in \{0,1\}$, generate $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b}) \leftarrow \mathsf{RFE.KeyGen}(\mathsf{crs}_{i,b}, \mathsf{aux}_{i,b})$.
- Set $\mathsf{pk} := (\{\mathsf{pk}_{i,b}\}_{i,b})$.
- Construct the secret key $\mathsf{sk} := iO(\mathcal{D}[\{\mathsf{sk}_{i,z[i]}\}_i, z])$ where the circuit is defined in Fig. 1.
- Output $(\mathsf{pk}, \mathsf{sk})$.

$\mathsf{RegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f)$:
- Parse $\mathsf{crs} = (\{\mathsf{crs}_{i,b}\}_{i,b}, y)$ and $\mathsf{pk} = (\{\mathsf{pk}_{i,b}\}_{i,b})$.
- If $\mathsf{aux} = \perp$, initialize $\mathsf{aux}_{i,b} := \perp$ for all i, b.
- For each i, b, compute $(\mathsf{mpk}_{i,b}, \mathsf{aux}'_{i,b}) \leftarrow \mathsf{RFE.RegPK}(\mathsf{crs}_{i,b}, \mathsf{aux}_{i,b}, \mathsf{pk}_{i,b}, f)$.
- Set $\mathsf{mpk} := (\{\mathsf{mpk}_{i,b}\}_{i,b}, y)$ and $\mathsf{aux}' := \{\mathsf{aux}'_{i,b}\}_{i,b}$.
- Output $(\mathsf{mpk}, \mathsf{aux}')$.

$\mathsf{Encrypt}(\mathsf{mpk}, m)$:
- Parse $\mathsf{mpk} = (\{\mathsf{mpk}_{i,b}\}_{i,b}, y)$
- For each i, b, compute $\mathsf{ct}_{i,b} \leftarrow \mathsf{RFE.Encrypt}(\mathsf{mpk}_{i,b}, m[i]; r_{i,b})$.
- Let $d := (\{\mathsf{mpk}_{i,b}, \mathsf{ct}_{i,b}\}_{i,b})$ and $w := (m, \{r_{i,b}\}_{i,b})$.
- Compute $\pi \leftarrow \langle \mathcal{P}(w), \mathcal{V}(y) \rangle(d)$.
- Output $\mathsf{ct} := (\{\mathsf{ct}_{i,b}\}_{i,b}, \pi)$.

$\mathsf{Update}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk})$:
- Parse $\mathsf{crs} = (\{\mathsf{crs}_{i,b}\}_{i,b}, y)$, $\mathsf{aux} = \{\mathsf{aux}_{i,b}\}_{i,b}$, and $\mathsf{pk} = \{\mathsf{pk}_{i,b}\}_{i,b}$.
- For each $i \in [\ell_m]$, $b \in \{0,1\}$, compute $\mathsf{hsk}_{i,b} \leftarrow \mathsf{RFE.Update}(\mathsf{crs}_{i,b}, \mathsf{aux}_{i,b}, \mathsf{pk}_{i,b})$.
- Output $\mathsf{hsk} := \{\mathsf{hsk}_{i,b}\}_{i,b}$.

$\mathsf{Decrypt}(\mathsf{sk}, \mathsf{hsk}, \mathsf{ct})$:
- Parse $\mathsf{sk} = \mathcal{D}_e$ and $\mathsf{hsk} = \{\mathsf{hsk}_{i,b}\}_{i,b}$.
- Compute and return $m' := \mathcal{D}_e(\mathsf{hsk}, \mathsf{ct})$.

$\mathsf{SimKeyGen}(\mathsf{crs}, \mathsf{aux}, \mathsf{B})$:

- Parse $\mathsf{crs} = (\{\mathsf{crs}_{i,b}\}_{i,b}, y)$.
- For all $i \in [\ell_m], b \in \{0,1\}$, compute $(\mathsf{pk}_{i,b}, \mathsf{sk}_{i,b}) \leftarrow \mathsf{RFE.KeyGen}(\mathsf{crs}_{i,b}, \mathsf{aux}_{i,b})$.
- Sample $z^*_{\mathsf{pk}} \leftarrow \{0,1\}^{\ell_m}$ uniformly at random.
- Set $\mathsf{pk} := (\{\mathsf{pk}_{i,b}\}_{i,b})$ and update $\mathsf{B}[\mathsf{pk}] = (\{\mathsf{sk}_{i,b}\}_{i,b}, z^*_{\mathsf{pk}})$.
- Output $(\widetilde{\mathsf{pk}}, \mathsf{B})$.

Fig. 2: Left Decryption Circuit

Left Decryption Circuit $\mathcal{D}_0$

Input: hsk, ct
Hardcoded: $\{\mathsf{sk}_{i,0}\}_{i\in[\ell_m]}$

1. Parse $\mathsf{ct} = \left(\{(\mathsf{ct}_{i,0}, \mathsf{ct}_{i,1})\}_{i\in[\ell_m]}, \pi\right)$
2. Parse $\mathsf{hsk} = \left(\{\mathsf{hsk}_{i,0}, \mathsf{hsk}_{i,1}\}_{i\in[\ell_m]}\right)$
3. If $\pi \neq 1$, output $\perp$
4. For each $i \in [\ell_m]$, compute:
$$m'[i] \leftarrow \mathsf{RFE.Decrypt}(\mathsf{sk}_{i,0}, \mathsf{hsk}_{i,0}, \mathsf{ct}_{i,0})$$
5. Output: $m' = m'[1] \parallel m'[2] \parallel \cdots \parallel m'[\ell_m]$

$\mathsf{SimRegPK}(\mathsf{crs}, \mathsf{aux}, \mathsf{pk}, f)$:

- Parse $\mathsf{crs} = (\{\mathsf{crs}_{i,b}\}_{i,b}, y)$ and $\mathsf{pk} = (\{\mathsf{pk}_{i,b}\}_{i,b})$.
- For each i, b, compute $(\widetilde{\mathsf{mpk}}_{i,b}, \widetilde{\mathsf{aux}}_{i,b}) \leftarrow \mathsf{RFE.RegPK}(\mathsf{crs}_{i,b}, \mathsf{aux}_{i,b}, \mathsf{pk}_{i,b}, f)$.
- Set $\widetilde{\mathsf{mpk}} := (\{\widetilde{\mathsf{mpk}}_{i,b}\}_{i,b}, y)$ and $\widetilde{\mathsf{aux}} := \{\widetilde{\mathsf{aux}}_{i,b}\}_{i,b}$.
- Output $(\widetilde{\mathsf{mpk}}, \widetilde{\mathsf{aux}})$.

$\mathsf{SimCorrupt}(\mathsf{crs}, \mathsf{pk}, \mathsf{B})$:

- Parse $\mathsf{pk} = (\{\mathsf{pk}_{i,b}\}_{i,b})$ and $\mathsf{B}[\mathsf{pk}] = (\{\mathsf{sk}_{i,b}\}_{i,b}, z^*_{\mathsf{pk}})$.
- Compute $\widetilde{\mathsf{sk}} := iO(\mathcal{D}_0[\{\mathsf{sk}_{i,0}\}_i])$, where the circuit is defined in Fig. 2.
- Output $\widetilde{\mathsf{sk}}$.

$\mathsf{SimCT}(\widetilde{\mathsf{mpk}}, \mathsf{aux}, \mathsf{B})$:

- Parse $\widetilde{\mathsf{mpk}} = (\{\widetilde{\mathsf{mpk}}_{i,b}\}_{i,b}, y)$.
- Compute $z^* := \bigoplus_{\mathsf{pk}\in\mathsf{B}} z^*_{\mathsf{pk}}$ where $\mathsf{B}[\mathsf{pk}] = (\{\mathsf{sk}_{i,b}\}_{i,b}, z^*_{\mathsf{pk}})$.
- For each $i \in [\ell_m]$, compute:
$$\mathsf{ct}^*_{i,z^*[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z^*[i]}, 0),$$
$$\mathsf{ct}^*_{i,1-z^*[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,1-z^*[i]}, 1).$$

- Define $d^* := (\{\widetilde{\mathsf{mpk}}_{i,0}, \widetilde{\mathsf{mpk}}_{i,1}, \mathsf{ct}^*_{i,0}, \mathsf{ct}^*_{i,1}\}_i)$.
- Compute $\widetilde{\pi} \leftarrow \langle \mathsf{Sim} \rangle(d^*, y)$.
- Output $\widetilde{\mathsf{ct}} := (\{\mathsf{ct}^*_{i,b}\}_{i,b}, \widetilde{\pi})$.

$\mathsf{Reveal}(\mathsf{pk}, \mathsf{B}, \widetilde{\mathsf{ct}}, m)$:

- Parse $\mathsf{B}[\mathsf{pk}] = (\{\mathsf{sk}_{i,b}\}_{i,b}, z^*_{\mathsf{pk}})$.
- Parse $m = m[1] \parallel m[2] \parallel \cdots \parallel m[\ell_m]$.
- Compute $z^* := \bigoplus_{\mathsf{pk} \in \mathsf{B}} z^*_{\mathsf{pk}}$ where $\mathsf{B}[\mathsf{pk}] = (\{\mathsf{sk}_{i,b}\}_{i,b}, z^*_{\mathsf{pk}})$.
- Compute $\widetilde{\mathsf{sk}} := iO(\mathcal{D}[\{\mathsf{sk}_{i,z^*[i] \oplus m[i]}\}_i, z^* \oplus m])$.
- Output $\widetilde{\mathsf{sk}}$.

Theorem 1. *The NC-RFE protocol is correct, update-efficient, and compact.*

Proof. The correctness of the NC-RFE protocol follows directly from the correctness of RFE and the perfect completeness of $\prod_{\mathsf{ZKA}}$. The simulated algorithms, SimKeyGen, SimRegPK, SimCT, SimCorrupt, Reveal, play no role in establishing correctness; they affect only the security properties of the protocol. Moreover, NC-RFE inherits the update-efficiency and compactness properties of $\prod_{\mathsf{RFE}}$.

5 Security Proof

We now analyze the security of our NC-RFE construction and prove the following theorem.

Theorem 2. *If $\prod_{RFE}$ is secure and assuming the existence of an indistinguishability obfuscator $i\mathcal{O}$ for P/poly, and an interactive zero-knowledge argument system $\prod_{\mathsf{ZKA}}$ for NP language, then the NC-RFE construction is secure.*

Proof. We define a sequence of hybrid experiments to transition between real and ideal security games.

Hyb_0: This corresponds precisely to the security experiment $\mathsf{Exp}^{\mathsf{NC\text{-}RFE}}_{\mathcal{A}}(\lambda, 0)$. Throughout the analysis, m refers to the target message chosen by $\mathcal{A}$.

Hyb_1: This hybrid modifies Hyb_0 by altering the generation of secret keys during the query phase. Instead of using the left-or-right decryption circuit $\mathcal{D}[\{\mathsf{sk}_{i,z[i]}\}_{i \in [\ell_m]}, z]$, the challenger now employs the left-only decryption circuit $\mathcal{D}_0[\{\mathsf{sk}_{i,0}\}_{i \in [\ell_m]}]$. Consequently, for each key-generation query, it provides $\widetilde{\mathsf{sk}} = iO\left(\mathcal{D}_0[\{\mathsf{sk}_{i,0}\}_{i \in [\ell_m]}]\right)$, instead of $\mathsf{sk} = iO\left(\mathcal{D}[\{\mathsf{sk}_{i,z[i]}\}_{i \in [\ell_m]}, z]\right)$. As a result, the secret keys in this hybrid no longer depend on the random selection bit string z.

Hyb_2: This game is the same as Hyb_1 apart from the fact that it transitions from real to simulated machine $\langle Sim \rangle$ of an interactive proof system given by the interactive machines $\langle \mathcal{P}, \mathcal{V} \rangle$ for an NP language $\mathcal{L}$ characterized by the relation R in the ciphertext.

Hyb_3**:** This game is the same as Hyb_2 except that it introduces asymmetric encryption patterns in the challenge ciphertext. Rather than encrypting the same message bit under both master public keys, the challenger now generates position-dependent encryptions. For the combined randomness $z := \bigoplus z_{\text{user}}$, where z_{user} is sampled in KeyGen for every user and each bit position $i \in [\ell_m]$, it computes:

$$\mathsf{ct}^*_{i,z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z[i]}, m[i]),$$
$$\mathsf{ct}^*_{i,1-z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,1-z[i]}, 1-m[i]).$$

Hyb_4: This game is the same as Hyb_3, except that it is completely independent of the target message m. The challenger uses the same combined randomness $z^* := \bigoplus z^*_{\text{user}}$, where z^*_{user} is sampled in KeyGen for every user and generates challenge ciphertexts according to:

$$\mathsf{ct}^*_{i,z^*[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z^*[i]}, 0),$$
$$\mathsf{ct}^*_{i,1-z^*[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,1-z^*[i]}, 1).$$

When revealing secret keys in the final phase, the challenger computes for each honest user with randomness z^*_{user}:

$$\mathsf{sk} := iO(\mathcal{D}[\{\mathsf{sk}_{i,z^*_{\text{user}}[i]\oplus m[i]}\}_{i\in[\ell_m]}, z^*_{\text{user}} \oplus m]).$$

Note that the game Hyb_4 is identical to the simulated experiment $\mathsf{Exp}^{\mathsf{NC\text{-}RFE}}_{\mathcal{A}}(\lambda, 1)$.

Let the distribution of the output of an execution of Hyb_j, with $\mathcal{A}$, be denoted as $\mathsf{Hyb}_j(\mathcal{A})$. We establish computational indistinguishability between consecutive hybrid games through the reductions.

Lemma 1. *Suppose that $\prod_{RFE}$ satisfies perfect correctness, $\prod_{\mathsf{ZKA}}$ achieves soundness property, and the indistinguishability obfuscator $i\mathcal{O}$ ensures security under indistinguishability. Then, the following holds,*

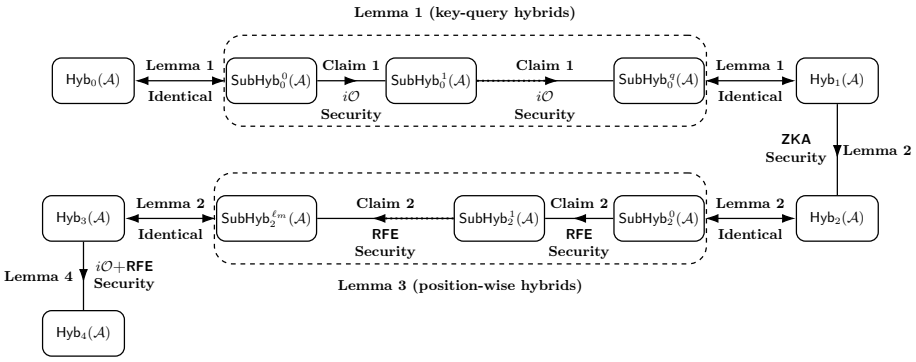
$$|\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda).$$

Table 1: Hybrid sequence for the security proof of NC-RFE

Hybrid	Secret Keys	uNIZK Proofs	Challenge Ciphertext	Dependence on m
Hyb_0	$iO(\mathcal{D}[\{\mathsf{sk}_{i,z[i]}\}_{i\in[\ell_m]}, z])$	Real proofs	$\mathsf{ct}^*_{i,0}, \mathsf{ct}^*_{i,1}$ encrypt $m[i]$	Yes (depends on m)
Hyb_1	$iO(\mathcal{D}_0[\{\mathsf{sk}_{i,0}\}_{i\in[\ell_m]}])$	↓	↓	↓
Hyb_2	↓	Simulated proofs using $\langle Sim\rangle$	↓	↓
Hyb_3	↓	↓	Asymmetric encryptions: $\mathsf{ct}^*_{i,z[i]}, \mathsf{ct}^*_{i,1-z[i]}$	↓
Hyb_4	$iO(\mathcal{D}[\{\mathsf{sk}_{i,z^*_{\text{user}}[i]\oplus m[i]}\}, z^*_{\text{user}} \oplus m])$	↓	Encrypt fixed bits 0/1 under z^* randomness	No (independent of m)

Simulated proof $\langle Sim\rangle$ appears in Hyb_2, and asymmetric challenge ciphertexts are introduced in Hyb_3. In Hyb_4, the ciphertext becomes entirely independent of the challenge message m. Arrows (↓) indicate no change from the previous hybrid.

Fig. 3: Hybrid argument for the security proof



Proof. We employ a hybrid argument over the adversary's key-generation queries. Let q denote the total number of such queries made by $\mathcal{A}$. Now, we define a sequence of hybrid sub-experiments SubHyb_0^j for each $j \in [0, q]$, where the challenger responds to the adversary's key queries as follows:

- For queries $k \in [j+1, q]$, the challenger returns $\mathsf{sk} = i\mathcal{O}\left(\mathcal{D}\left[\{\mathsf{sk}_{i,z[i]}\}_{i\in[\ell_m]}, z\right]\right)$.
- For queries $k \in [1, j]$, the challenger returns $\widetilde{\mathsf{sk}} = i\mathcal{O}\left(\mathcal{D}_0\left[\{\mathsf{sk}_{i,0}\}_{i\in[\ell_m]}\right]\right)$.

We observe that $\mathsf{SubHyb}_0^0(\mathcal{A}) = \mathsf{Hyb}_0(\mathcal{A})$, and $\mathsf{SubHyb}_0^q(\mathcal{A}) = \mathsf{Hyb}_1(\mathcal{A})$.

Let $\mathsf{Invalid}$ denote the event that there exists a statement $d^\dagger \notin \mathcal{L}$ such that for $\langle \mathcal{P}(w), \mathcal{V}(y)\rangle(d) = 1$. By the soundness of $\prod_{\mathsf{ZKA}}$, it holds that $\Pr[\mathsf{Invalid}] \leq \mathsf{negl}(\lambda)$.

Conditioned on the event $\neg\mathsf{Invalid}$, the circuits $\mathcal{D}$ and $\mathcal{D}_0$ are functionally equivalent on all valid inputs. This equivalence follows from their construction and the perfect correctness of the underlying RFE scheme. We now formalize the indistinguishability of adjacent subgames.

Claim 1. For every $j \in [1, q]$, the distinguishing advantage between hybrid subexperiments $\mathsf{SubHyb}_0^{j-1}(\mathcal{A})$ and $\mathsf{SubHyb}_0^j(\mathcal{A})$ is negligible, *i.e.,*

$$\left|\Pr[\mathsf{SubHyb}_0^{j-1}(\mathcal{A}) = 1] - \Pr[\mathsf{SubHyb}_0^j(\mathcal{A}) = 1]\right| \leq \mathsf{negl}(\lambda).$$

Proof. Consider the transition between hybrid sub-experiments SubHyb_0^{j-1} and SubHyb_0^j, which differ only in the j-th key query. In SubHyb_0^{j-1}, the response is $\mathsf{sk} = i\mathcal{O}\left(\mathcal{D}\left[\{\mathsf{sk}_{i,z[i]}\}_{i\in[\ell_m]}, z\right]\right)$, while in SubHyb_0^j, the response is $\widetilde{\mathsf{sk}} = i\mathcal{O}(\mathcal{D}_0\left[\{\mathsf{sk}_{i,0}\}_{i\in[\ell_m]}\right])$. Under the assumption $\neg\mathsf{Invalid}$, the two circuits $\mathcal{D}$ and $\mathcal{D}_0$ are functionally equivalent. Hence, by the security of indistinguishability obfuscator $i\mathcal{O}$, any QPT distinguisher can distinguish between these responses with at most negligible probability.

Applying the hybrid sub-experiments argument across the q key queries and invoking the above claim repeatedly, we conclude that

$$|\Pr[\mathsf{Hyb}_0(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1]| \leq q \cdot \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda),$$

which completes the proof of the lemma.

Lemma 2. *If $\prod_{ZKA}$ satisfies the zero-knowledge property, then*

$$|\Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda).$$

Proof. The hybrid experiments Hyb_1 and Hyb_2 differ solely in the generation of the ciphertext. Suppose that $|\Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1]|$ is non-negligible. Then there exists a distinguisher that separates the ciphertext distributions used in Hyb_1 and Hyb_2 with non-negligible advantage.

This would further imply that the distributions of the real and simulated proof, namely $\langle \mathcal{P}(w), \mathcal{V}(y)\rangle(d)$ and $\langle \mathsf{Sim}\rangle(d^*, y)$, are computationally distinguishable. Such a distinguisher contradicts the zero-knowledge property of $\prod_{\mathsf{ZKA}}$, since it would allow one to efficiently distinguish the real view of the verifier from a simulated one for instances of size $|y|$ with non-negligible advantage.

Hence, by the zero-knowledge property, the advantage $\big|\Pr[\mathsf{Hyb}_1(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1]\big|$ must be negligible.

Lemma 3. *If the underlying $\prod_{RFE}$ scheme is secure, then*

$$|\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1]| \leq \mathsf{negl}(\lambda).$$

Proof. We employ a position-by-position hybrid argument across the message bits. For each position $i \in [\ell_m]$, we demonstrate that modifying the encryption pattern at position i introduces at most a negligible distinguishing advantage.

We now define the hybrid sub-experiments games SubHyb_2^j for $j \in [0, \ell_m]$, where the challenger responds as follows:

- For positions $i \in [j+1, \ell_m]$, both ciphertexts encrypt the same message bit.

$$\mathsf{ct}^*_{i,b} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,b}, m[i]) \text{ for } b \in \{0,1\}$$

- For positions $i \in [1, j]$, the ciphertexts are generated using the following computation.

$$\mathsf{ct}^*_{i,1-z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,1-z[i]}, 1 - m[i]),$$
$$\mathsf{ct}^*_{i,z[i]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{i,z[i]}, m[i])$$

By construction, $\mathsf{SubHyb}_2^0(\mathcal{A}) = \mathsf{Hyb}_2(\mathcal{A})$ and $\mathsf{SubHyb}_2^{\ell_m}(\mathcal{A}) = \mathsf{Hyb}_3(\mathcal{A})$. To complete the proof, we introduce the following claim.

Claim 2. For every $j \in [1, \ell_m]$, the hybrids $\mathsf{SubHyb}_2^{j-1}(\mathcal{A})$ and $\mathsf{SubHyb}_2^{j}(\mathcal{A})$ are computationally indistinguishable under the security of the RFE scheme.

Proof. To show that adjacent games SubHyb_2^{j-1} and SubHyb_2^{j} for $i \in [1, \ell_m]$, are computationally indistinguishable, we define a reduction $\mathcal{B}_{\mathsf{RFE}}$ against the security of the underlying RFE scheme. $\mathcal{B}_{\mathsf{RFE}}$ receives master public key $\widetilde{\mathsf{mpk}}$ from the RFE challenger of $\mathsf{Exp}_{\mathcal{A}}^{\mathsf{RFE}}(\lambda, b^{'})$ for $b^{'} \in \{0,1\}$ and sets $\widetilde{\mathsf{mpk}}_{j,1-z[j]} := \widetilde{\mathsf{mpk}}$. For all other public keys, $(i,b) \neq (j, 1-z[j])$, it generates $\widetilde{\mathsf{mpk}}_{i,b}$ itself. It then compiles the overall master public key as $\widetilde{\mathsf{mpk}} := \left(\{\widetilde{\mathsf{mpk}}_{i,b}\}_{i\in[\ell_m], b\in\{0,1\}}, y\right)$, which is given to $\mathcal{A}$. Similar to before, $\mathcal{B}_{\mathsf{RFE}}$ receives $\mathsf{sk}_{j,1-z[j]}$ from the RFE challenger, and locally generates all other secret keys. Since these keys are never used in any decryption or response, the simulation remains indistinguishable. $\mathcal{B}_{\mathsf{RFE}}$ then constructs $\widetilde{\mathsf{sk}} := iO\left(\mathcal{D}_0[\{\mathsf{sk}_{i,0}\}_{i\in[\ell_m]}]\right)$. At the challenge phase, $\mathcal{A}$ submits m to the $\mathcal{B}_{\mathsf{RFE}}$ and it sends the message pair $(m[j], 1-m[j])$ to the RFE challenger. It then computes $\mathsf{ct}^*_{j,z[j]} \leftarrow \mathsf{RFE.Encrypt}\ (\widetilde{\mathsf{mpk}}_{j,z[j]},\ m[j])$, and all remaining ciphertexts $\mathsf{ct}^*_{i,b}$ for $i \neq j$, as in the hybrids SubHyb_2^{j-1} and SubHyb_2^{j}. Finally, it outputs whatever $\mathcal{A}$ outputs.

The reduction proceeds as follows:

- If $b' = 0$, then $\mathsf{ct}^*_{j,1-z[j]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{j,1-z[j]}, m[j])$, which implies that $\mathcal{B}_{\mathsf{RFE}}$ perfect simulation of the hybrid sub-experiment SubHyb_2^{j-1}.
- If $b' = 1$, then $\mathsf{ct}^*_{j,1-z[j]} \leftarrow \mathsf{RFE.Encrypt}(\widetilde{\mathsf{mpk}}_{j,1-z[j]}, 1-m[j])$, which implies that $\mathcal{B}_{\mathsf{RFE}}$ perfect simulation of the hybrid sub-experiment SubHyb_2^{j-1}.

Hence, any non-negligible advantage by $\mathcal{A}$ in distinguishing between SubHyb_2^{j-1} and SubHyb_2^{j} can be used to break the security of the RFE scheme. Therefore, the hybrid sub-experiments are computationally indistinguishable.

Applying the hybrid sub-experiment argument to all ℓ_m positions and using the above claim iteratively, we conclude that

$$|\Pr[\mathsf{Hyb}_2(\mathcal{A}) = 1] - \Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1]| \leq \ell_m \cdot \mathsf{negl}(\lambda) = \mathsf{negl}(\lambda).$$

This concludes the proof of Lemma 3.

Lemma 4. *The hybrid games satisfy perfect indistinguishability:*

$$\Pr\left[\mathsf{Hyb}_3(\mathcal{A}) = 1\right] = \Pr\left[\mathsf{Hyb}_4(\mathcal{A}) = 1\right].$$

Proof. This hybrid transition is a purely formal change that preserves the adversary's view. Specifically, the two games become identical under the variable substitution $z := z^* \oplus m$.

In game Hyb_3, the challenge ciphertexts are given by: $\mathsf{ct}^*_{i,1-z[i]}$ encrypts $1 - m[i]$, and $\mathsf{ct}^*_{i,z[i]}$ encrypts $m[i]$. After applying the substitution $z := z^* \oplus m$, the ciphertexts in Hyb_4 take the form $\mathsf{ct}^*_{i,z^*[i]}$ encrypts 0, $\mathsf{ct}^*_{i,1-z^*[i]}$ encrypts 1.

Since the secret keys are generated independently of the selection string z in both hybrids, this change of variables does not affect the underlying distributions. Consequently, the adversary's view is identical across the two hybrids, and we conclude that $\Pr[\mathsf{Hyb}_3(\mathcal{A}) = 1] = \Pr[\mathsf{Hyb}_4(\mathcal{A}) = 1]$. This concludes the lemma.

Consequently, by combining Lemmas 1, 2, 3, and 4, we conclude that the NC-RFE construction satisfies security against all quantum polynomial-time adversaries.

6 Instantiation of NC-RFE from Lattices

Our generic construction of NC-RFE, presented in Section 4.3, is fully modular and can be instantiated from any secure registered FE scheme which satisfy perfect correctness, zero-knowledge argument system, and indistinguishability obfuscation. In this section, we give a concrete lattice-based instantiation of each component, resulting in a quantum-secure NC-RFE scheme under the hardness of plain LWE, equivocal LWE assumption.
We instantiate our generic framework using the following state-of-the-art lattice-based primitives:

- **Registered FE Component:** We employ the registered FE scheme for general function of Datta *et al.* [12, 13]. This scheme supports an arbitrary number of users from indistinguishability obfuscation [10] and somewhere statistically binding hash functions [24].
- **ZKA Component:** We utilize the post-quantum secure zero-knowledge argument system of Bitansky *et al.* [2] for arbitrary NP languages. Their construction achieves soundness and zero-knowledge under the plain LWE assumption against quantum adversaries.
- **Indistinguishability Obfuscation Component:** We utilize the lattice-based $i\mathcal{O}$ construction of Cini *et al.* [10], which provides post-quantum security under the equivocal LWE assumption in conjunction with standard NTRU lattice assumptions. This represents the first plausibly post-quantum secure $i\mathcal{O}$ construction from well-studied lattice problems.

Adapting the above-mentioned component as a building block, we can construct a post-quantum secure NC-RFE protocol from lattices.

7 Conclusion

In this work, we introduced non-committing registered functional encryption (NC-RFE), extending non-committing security to the registered setting. We presented a generic construction of NC-RFE based on indistinguishability obfuscation and zero-knowledge arguments, and proved its security under a secure underlying registered functional encryption scheme. In addition, we presented a lattice-based instantiation, yielding a post-quantum secure realization of NC-RFE. An interesting direction for future research is to design more efficient NC-RFE schemes from standard assumptions such as LWE, without relying on indistinguishability obfuscation.

Acknowledgments. This study is funded by 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. I would also like to express my sincere gratitude to Shayeef Murshid for the valuable discussions and insights.

References

1. Abdolmaleki, B., Ramacher, S., Slamanig, D.: Lift-and-shift: obtaining simulation extractable subversion and updatable snarks generically. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security. pp. 1987–2005 (2020)
2. Bitansky, N., Shmueli, O.: Post-quantum zero knowledge in constant rounds. In: Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing. pp. 269–279 (2020)
3. Boneh, D., Sahai, A., Waters, B.: Functional encryption: Definitions and challenges. In: Theory of Cryptography Conference. pp. 253–273. Springer (2011)
4. Brakerski, Z., Branco, P., Döttling, N., Garg, S., Malavolta, G.: Constant ciphertext-rate non-committing encryption from standard assumptions. In: Theory of Cryptography Conference. pp. 58–87. Springer (2020)
5. Brakerski, Z., Döttling, N., Garg, S., Malavolta, G.: Factoring and pairings are not necessary for io: Circular-secure lwe suffices. Cryptology ePrint Archive (2020)
6. Canetti, R., Feige, U., Goldreich, O., Naor, M.: Adaptively secure multi-party computation. In: Proceedings of the twenty-eighth annual ACM symposium on Theory of computing. pp. 639–648 (1996)
7. Canetti, R., Halevi, S., Katz, J.: Adaptively-secure, non-interactive public-key encryption. In: Theory of Cryptography Conference. pp. 150–168. Springer (2005)
8. Canetti, R., Poburinnaya, O., Raykova, M.: Optimal-rate non-committing encryption. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 212–241. Springer (2017)
9. Choi, S.G., Dachman-Soled, D., Malkin, T., Wee, H.: Improved non-committing encryption with applications to adaptively secure protocols. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 287–302. Springer (2009)
10. Cini, V., Lai, R.W., Woo, I.K.: Lattice-based obfuscation from NTRU and Equivocal LWE. Cryptology ePrint Archive (2025), `https://link.springer.com/chapter/10.1007/978-3-032-01907-3_2`
11. Cook, S.: The P versus NP problem. Clay Mathematics Institute **2** (2000), `https://www.claymath.org/wp-content/uploads/2022/06/pvsnp.pdf`
12. Datta, P., Pal, T.: Registration-based functional encryption. IACR Cryptol. ePrint Arch. **2023**, 457 (2023)
13. Datta, P., Pal, T., Yamada, S.: Registered fe beyond predicates:(attribute-based) linear functions and more. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 65–104. Springer (2024)
14. Garg, S., Hajiabadi, M., Mahmoody, M., Rahimi, A.: Registration-based encryption: removing private-key generator from ibe. In: Theory of Cryptography Conference. pp. 689–718. Springer (2018)
15. Gay, R., Pass, R.: Indistinguishability obfuscation from circular security. In: Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing. pp. 736–749 (2021)

16. Goyal, R., Kitagawa, F., Koppula, V., Nishimaki, R., Rajasree, M.S., Yamakawa, T.: Non-committing identity based encryption: constructions and applications. In: IACR International Conference on Public-Key Cryptography. pp. 158–193. Springer (2025)
17. Goyal, R., Syed, R., Waters, B.: Bounded collusion abe for tms from ibe. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 371–402. Springer (2021)
18. Goyal, V., Pandey, O., Sahai, A., Waters, B.: Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM conference on Computer and communications security. pp. 89–98 (2006), https://doi.org/10.1145/1180405.1180418
19. Groth, J., Kohlweiss, M., Maller, M., Meiklejohn, S., Miers, I.: Updatable and universal common reference strings with applications to zk-snarks. In: Annual International Cryptology Conference. pp. 698–728. Springer (2018)
20. Hazay, C., Patra, A., Warinschi, B.: Selective opening security for receivers. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 443–469. Springer (2015)
21. Hemenway, B., Ostrovsky, R., Richelson, S., Rosen, A.: Adaptive security with quasi-optimal rate. In: Theory of Cryptography Conference. pp. 525–541. Springer (2015)
22. Hiroka, T., Morimae, T., Nishimaki, R., Yamakawa, T.: Quantum encryption with certified deletion, revisited: Public key, attribute-based, and classical communication. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 606–636. Springer (2021)
23. Hohenberger, S., Lu, G., Waters, B., Wu, D.J.: Registered attribute-based encryption. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 511–542. Springer (2023)
24. Koppula, V., Lewko, A.B., Waters, B.: Indistinguishability obfuscation for turing machines with unbounded memory. In: Proceedings of the forty-seventh annual ACM symposium on Theory of Computing. pp. 419–428 (2015)
25. Murshid, S., Sarkar, R., Mandal, M.: Registered attribute-based encryption with publicly verifiable certified deletion, everlasting security, and more. Cryptology ePrint Archive (2025)
26. Nielsen, J.B.: Separating random oracle proofs from complexity theoretic proofs: The non-committing encryption case. In: Annual International Cryptology Conference. pp. 111–126. Springer (2002)
27. Sahai, A., Waters, B.: Fuzzy identity-based encryption. In: Annual international conference on the theory and applications of cryptographic techniques. pp. 457–473. Springer (2005), https://doi.org/10.1007/11426639_27
28. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Workshop on the theory and application of cryptographic techniques. pp. 47–53. Springer (1984)
29. Turing, A.M.: On computable numbers, with an application to the entscheidungsproblem. Proceedings of the London Mathematical Society **s2-42**(1), 230–265 (1936), https://doi.org/10.1112/plms/s2-42.1.230
30. Wee, H., Wichs, D.: Candidate obfuscation via oblivious lwe sampling. In: Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 127–156. Springer (2021)
31. Yoshida, Y., Kitagawa, F., Tanaka, K.: Non-committing encryption with quasi-optimal ciphertext-rate based on the ddh problem. In: International Conference on the Theory and Application of Cryptology and Information Security. pp. 128–158. Springer (2019)