# Unidirectional Updatable Encryption and Proxy Re-encryption from DDH*

Peihan Miao[1], Sikhar Patranabis[2], and Gaven Watson[3]

[1] Brown University, `peihan_miao@brown.edu`
[2] IBM Research India, `sikhar.patranabis@ibm.com`
[3] Meta, `gavenwatson@meta.com`

**Abstract**

Updatable Encryption (UE) and Proxy Re-encryption (PRE) allow *re-encrypting* a ciphertext from one key to another in the symmetric-key and public-key settings, respectively, without decryption. A longstanding open question has been the following: do *unidirectional* UE and PRE schemes (where ciphertext re-encryption is permitted in only one direction) necessarily require stronger/more structured assumptions as compared to their bidirectional counterparts? Known constructions of UE and PRE seem to exemplify this "gap" – while bidirectional schemes can be realized as relatively simple extensions of public-key encryption from standard assumptions such as DDH or LWE, unidirectional schemes typically rely on stronger assumptions such as FHE or indistinguishability obfuscation (iO), or highly structured cryptographic tools such as bilinear maps or lattice trapdoors.

In this paper, we bridge this gap by showing the first feasibility results for realizing unidirectional UE and PRE from a new generic primitive that we call Key and Plaintext Homomorphic Encryption (KPHE) – a public-key encryption scheme that supports additive homomorphisms on its plaintext and key spaces simultaneously. We show that KPHE can be instantiated from DDH. This yields the first constructions of unidirectional UE and PRE from DDH.

Our constructions achieve the strongest notions of *post-compromise security* in the standard model. Our UE schemes also achieve "backwards-leak directionality" of key updates (a notion we discuss is equivalent, from a security perspective, to that of unidirectionality with no-key updates). Our results establish (somewhat surprisingly) that unidirectional UE and PRE schemes satisfying such strong security notions *do not*, in fact, require stronger/more structured cryptographic assumptions as compared to bidirectional schemes.

 Authors 1 and 2 did part of the work while at VISA Research. Author 3 contributed solely while at VISA Research. P. Miao is supported in part by the NSF CNS Award 2055358, a 2020 DPI Science Team Seed Grant, and a Meta award. The authors thank the anonymous reviewers of IACR PKC 2023 for their helpful comments and suggestions.

# Contents

## 1 Introduction

Cryptographic encryption is a powerful tool for ensuring data confidentiality. A common security guarantee offered by any encryption scheme (either symmetric-key or public-key) is the following: encrypted data can only be decrypted using a certain secret key. However, a limitation of traditional encryption schemes is that once data is encrypted, it is generally hard to allow a third party to transform the ciphertext so that it can be decrypted with a different key, without sharing either the original or the new secret key with the third party.

Re-encryption schemes such as Proxy Re-encryption (PRE) [BBS98] and Updatable Encryption (UE) [BLMR13] circumvent this limitation by enabling a public transformation of ciphertexts from encryption under one key to that of another, while protecting the underlying secret keys. Classic applications for such schemes include key rotation for secure outsourced storage [BBB$^+$12, Pay18], access control, the delegation of email access, and many more.

**Proxy Re-encryption (PRE).** PRE is a public-key encryption scheme which enables a party Alice, with the help of a proxy, to re-encrypt her ciphertexts for decryption by an alternate party Bob. To facilitate re-encryption, Alice and Bob, with key pairs $(\mathsf{pk}_A, \mathsf{sk}_A)$ and $(\mathsf{pk}_B, \mathsf{sk}_B)$ respectively, will together compute a re-encryption key $\mathsf{rk}_{AB}$ and then provide this to the proxy. Whenever the proxy needs to perform re-encryption, it can use $\mathsf{rk}_{AB}$ to transform a ciphertext encrypted under $\mathsf{pk}_A$ into a ciphertext encrypted under $\mathsf{pk}_B$. Security of the PRE scheme guarantees that the proxy learns nothing about the underlying plaintext during the re-encryption process.

**Updatable Encryption (UE).** UE was introduced by Boneh et al. [BLMR13] to address the problem of key rotation for secure outsourced storage. UE addresses re-encryption by using similar techniques to those of PRE, with two main differences: (1) UE is a symmetric-key encryption scheme, and (2) UE typically only allows sequential updates. More specifically, in UE we divide time into a series of epochs. In the first epoch a fresh symmetric key $\mathtt{k}_0$ is chosen and used to encrypt all data. When we rotate a key from $\mathtt{k}_{\mathsf{e}-1}$ to $\mathtt{k}_{\mathsf{e}}$, we transition to the next epoch by calculating an update token $\Delta_{\mathsf{e}}$. All new ciphertexts are encrypted under the new key $\mathtt{k}_{\mathsf{e}}$ and all existing ciphertexts $\mathsf{ct}_{\mathsf{e}-1}$ are re-encrypted using the update token $\Delta_{\mathsf{e}}$ so that they can be decrypted by $\mathtt{k}_{\mathsf{e}}$. The benefit of this approach is that the storage server can perform the re-encryption of data using the update token without the risk of exposing any plaintext data.

There are two variants of UE schemes, *ciphertext-dependent* schemes [EPRS17, BEKS20] and *ciphertext-independent* schemes [LT18, KLR19, BDGJ20, Jia20]. In ciphertext-dependent UE schemes, the update token $\Delta_{\mathsf{e},\mathsf{ctx}_{\mathsf{e}-1}}$ depends on the ciphertext $\mathsf{ctx}_{\mathsf{e}-1}$ to be updated, while in ciphertext-independent schemes, the update token $\Delta_{\mathsf{e}}$ is generated independent of the updated ciphertext, hence a single token can be used to update all ciphertexts on the storage server. In the rest of the paper, when we refer to UE, we mean a ciphertext-independent scheme unless otherwise specified.

**Directionality of PRE and UE.** Re-encryption schemes are either *bidirectional* or *unidirectional*. A scheme is said to be bidirectional if a re-encryption key/update token can be used to re-encrypt a ciphertext to either the next party/epoch or the previous party/epoch. In contrast, the re-encryption key/update token of a unidirectional scheme can only be used to re-encrypt a ciphertext to the next party/epoch and *not* the previous. So far we have only discussed the directionality within the context of ciphertext updates. The (uni)directionality with regards to keys differs slightly between PRE and UE, as we discuss next.

**Bidirectionality vs. Unidirectionality in PRE.** In bidirectional PRE schemes, the re-encryption key $\mathsf{rk}_{AB}$ from Alice to Bob is generated from Alice's key-pair $(\mathsf{pk}_A, \mathsf{sk}_A)$ and Bob's key-pair $(\mathsf{pk}_B, \mathsf{sk}_B)$. Given $\mathsf{rk}_{AB}$ along with $\mathsf{sk}_A$ (resp. $\mathsf{sk}_B$), it is usually possible to derive $\mathsf{sk}_B$ (resp. $\mathsf{sk}_A$). In unidirectional PRE schemes, the re-encryption key $\mathsf{rk}_{AB}$ is derived from $((\mathsf{pk}_A, \mathsf{sk}_A), \mathsf{pk}_B)$; Bob's secret key $\mathsf{sk}_B$ is not used. In fact, given the re-encryption key $\mathsf{rk}_{AB}$ and Alice's secret key $\mathsf{sk}_A$, it should be impossible to derive any knowledge of Bob's secret key $\mathsf{sk}_B$.

**Unidirectionality in UE.** For UE schemes, there is an extra level of subtlety regarding the *directionality of keys* in addition to ciphertexts. A recent work of Jiang [Jia20] extensively studied the question: given an update token $\Delta_{\mathsf{e}}$ along with either $\mathtt{k}_{\mathsf{e}-1}$ or $\mathtt{k}_{\mathsf{e}}$, is it possible to derive the other key? A scheme has bidirectional key updates if $\Delta_{\mathsf{e}}$ can be used to derive keys in both directions, and has unidirectional key updates if $\Delta_{\mathsf{e}}$ can be used in one direction, to derive $\mathtt{k}_{\mathsf{e}}$ from $\mathtt{k}_{\mathsf{e}-1}$. Jiang [Jia20] showed that UE with bidirectional key and ciphertext updates implies UE with unidirectional key and ciphertext updates.

In the same work, Jiang postulated that to capture the same security level as the unidirectional PRE schemes, one requires even stronger UE schemes with *no-directional key updates*, where $\mathtt{k}_{\mathsf{e}}$ cannot be derived from $\mathtt{k}_{\mathsf{e}-1}$ and $\Delta_{\mathsf{e}}$. In Jiang [Jia20], the definition of no-directional key updates intuitively requires that it is *also* impossible to derive $\mathtt{k}_{\mathsf{e}-1}$ from $\Delta_{\mathsf{e}}$ and $\mathtt{k}_{\mathsf{e}}$. The recent work of Nishimaki [Nis21] proposed a seemingly weaker notion called *backward-leak unidirectional key updates* where $\Delta_{\mathsf{e}}$ can only be used in one direction to derive $\mathtt{k}_{\mathsf{e}-1}$ from $\mathtt{k}_{\mathsf{e}}$. However, we observe that this new notion is essentially equivalent to no-directional key updates because derivation of $\mathtt{k}_{\mathsf{e}-1}$ does not increase the adversary's advantage in breaking the scheme. In particular, if the adversary obtains a ciphertext $\mathsf{ct}_{\mathsf{e}-1}$ and corrupts $\Delta_{\mathsf{e}}$ and $\mathtt{k}_{\mathsf{e}}$, then it can first update the ciphertext to $\mathsf{ct}_{\mathsf{e}}$ and decrypt it using $\mathtt{k}_{\mathsf{e}}$. Jiang emphasized that UE with no-directional key updates is the ideal security model, which by our argument above, extends to backwards-leak key updates. Henceforth, when we refer to unidirectional UE, we mean unidirectional UE with backwards-leak directional key updates unless otherwise specified.

**Gap between Unidirectionality and Bidirectionality.** In general, unidirectional UE and PRE schemes are more ideally suited to real-world applications as compared to their bidirectional counterparts due to their superior security guarantees. For example, unlike bidirectional UE schemes, unidirectional UE schemes guarantee security of data as if "freshly encrypted" in epoch $\mathsf{e}$ (i.e., not re-encrypted from epoch $(\mathsf{e}-1)$) even if the adversary gains access to the secret key $\mathtt{k}_{\mathsf{e}-1}$ and the update token $\Delta_{\mathsf{e}}$. Unidirectional PRE schemes also offer similarly superior security guarantees over their bidirectional counterparts.

Another natural point of comparison between unidirectional and bidirectional UE and PRE schemes is the nature of cryptographic assumptions from which such schemes can be realized. Known constructions of UE and PRE seemingly exemplify an apparent "gap" in terms of the assumptions required – unidirectional schemes have historically relied on stronger/more structured cryptographic assumptions as compared to their bidirectional counterparts.

Blaze et al. [BBS98] showed how to construct bidirectional PRE schemes from the Decisional Diffie-Hellman (DDH) assumption by suitably extending the well-known ElGamal encryption scheme [Gam85]. Similarly, a long line of works [BLMR13,LT18,KLR19,BDGJ20,Jia20] have shown how to realize bidirectional UE schemes as relatively simple extensions of public-key encryption from standard assumptions such as DDH and Learning With Errors (LWE).

On the other hand, unidirectional UE and PRE schemes typically rely on a stronger set of assumptions such as FHE [Gen09] and indistinguishability obfuscation (iO) [BGI+12], or highly structured cryptographic tools such as bilinear maps [BF03] and "hard" lattice trapdoors [GPV08]. Examples of constructions of unidirectional PRE from FHE and/or structured lattice trapdoors can be found in [NX15, CCL+14, Kir14, NAL15, PWA+16, FL17, PRSV17]. Constructions of unidirectional PRE schemes have also been shown to exist from bilinear maps [AFGH06, LV11]; however, these constructions are restricted to the *single-hop* setting in the sense that they only permit a single re-encryption of a ciphertext. Known constructions of unidirectional UE include the construction in [SS21] (which relies on bilinear maps), and two constructions in [Nis21] (one which achieves backward-leak key updates from lattice-specific techniques, and one which achieves no-directional key updates from iO). Sehrawat and Desmedt show a construction of UE from bi-homomorphic lattice-based pseudorandom functions [SD19]; however, their construction only achieves unidirectional ciphertext updates while still incurring bidirectional key updates (and is hence effectively bidirectional as per the recent findings in [Jia20]). To date, there exist no constructions of unidirectional PRE or UE from the plain DDH assumption (to our knowledge).

In this paper, we are motivated by the following longstanding open question in the study of UE and PRE:

*Do unidirectional UE and PRE schemes necessarily require stronger/more structured assumptions as compared to their bidirectional counterparts?*

More concretely, we ask the following question:

*Can we construct unidirectional UE/PRE schemes from DDH?*

### 1.1 Our Results

In this paper, we bridge this gap between the assumptions for unidirectional and bidirectional UE/PRE. We establish (somewhat surprisingly) that unidirectional UE and PRE schemes *do not*, in fact, require stronger/more structured cryptographic assumptions as compared to their bidirectional counterparts.

More concretely, we present generic constructions of unidirectional UE and PRE from a new primitive that we call Key and Plaintext Homomorphic Encryption (KPHE). We also show that such a KPHE scheme can be instantiated from the BHHO encryption scheme [BHHO08] based on the DDH assumption. This yields the first constructions of unidirectional UE and PRE from the plain DDH assumption.

Our main result is summarized by the following (informal) theorem:

**Theorem 1.1** (Informal)**.** *Assuming the existence of a Key and Plaintext Homomorphic Encryption (KPHE) scheme that satisfies certain special properties, there exist post-compromise secure unidirectional UE and PRE schemes.*

**On KPHE.** The KPHE scheme with special properties required in our constructions can be viewed as a generalization of the BHHO public-key encryption scheme due to Boneh et al. [BHHO08].

It is a public key encryption scheme where the secret key is a bit-string $\mathsf{sk} \in \{0,1\}^{\ell}$ and the plaintext is also a bit-string $\mathsf{m} \in \{0,1\}^{\ell'}$ (in our constructions we use $\ell = \ell' = 2n$). The specialized KPHE scheme satisfies the following three properties:

- **Distributional Semantic Security:** We require a KPHE scheme to achieve semantic security even when the secret keys are sampled from a specific distribution. In particular, we use KPHE schemes with $2n$-bit secret keys where the secret key is uniformly random subject to the constraint that it has equally many 0 and 1 bits (i,e., $n$ bits of 0 and $n$ bits of 1).

- **Additive Key and Plaintext Homomorphisms:** We require a KPHE scheme to satisfy the following property: let $T, T'$ be two arbitrary affine transformations that map 0-1 vectors to 0-1 vectors of the same length (in our constructions we use permutation maps over the bits of a $2n$-bit string). Then, given a public key $\mathsf{pk}$ corresponding to some secret key $\mathsf{sk}$ and a ciphertext $\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$, one can generate a public key $\mathsf{pk}'$ corresponding to the secret key $T(\mathsf{sk})$ and a ciphertext $\mathsf{ct}' \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}', T'(\mathsf{m}))$, without the knowledge of the original secret key $\mathsf{sk}$ or the original message $\mathsf{m}$.

- **Blinding:** We also require the KPHE scheme to satisfy an associated security property called "blinding", that (informally) argues that the public key and ciphertext generated via the aforementioned homomorphic transformations are indistinguishable from freshly generated public keys and ciphertexts (we make this more formal in Section 2).

For our PRE constructions, we also require that the KPHE scheme satisfies a notion of distributional circular security (i.e., circular security when the secret keys are sampled from a specific distribution). This is not required for our UE constructions.

**Instantiating KPHE.** We show how to concretely instantiate a KPHE scheme satisfying all of the aforementioned properties from DDH (based on the BHHO scheme [BHHO08]).

**Lemma 1.2** (Informal)**.** *Assuming decisional Diffie-Hellman (DDH) holds, there exists a secure construction of KPHE that satisfies the aforementioned properties.*

**Corollary 1.3** (Informal)**.** *Assuming the decisional Diffie-Hellman (DDH) assumption holds, there exist post-compromise secure unidirectional UE and PRE schemes.*

**Security of Our Constructions.** Our constructions of unidirectional UE and PRE achieve the strongest notions of *post-compromise security* in the standard model. Our construction of unidirectional UE achieves the state-of-the-art post-compromise security definition due to Boyd et al. [BDGJ20], while also ensuring backward-leak unidirectional key updates [Nis21]. Our unidirectional PRE construction achieves the post-compromise security definition recently proposed by Davidson et al. [DDLM19], which is, to our knowledge, the only notion of post-compromise PRE security to be proposed to date. We present a more detailed discussion on post-compromise security (and other related security notions) of UE and PRE in the next subsection. Table 1 presents a comparison of our results with those in the existing literature.

| | scheme | dir. (ctx) | dir. (key) | security | assumption |
|---|---|---|---|---|---|
| UE | [BLMR13] | bi | bi | IND-ENC | DDH/LWE |
| UE | [LT18, KLR19, BDGJ20] | bi | bi | IND-UE | DDH |
| UE | [Jia20] | bi | bi | IND-UE | DLWE |
| UE | [SD19] | uni | bi | IND-UE | LWE |
| UE | [Nis21] | uni | bwd-uni | IND-UE | LWE |
| UE | [Nis21] | uni | no | IND-UE | iO |
| UE | [SS21] | uni | no | IND-UE | SXDH |
| UE | Ours | uni | bwd-uni | IND-UE | DDH |
| PRE | [BBS98] | bi | bi | IND-CPA | DDH |
| PRE | [CCL+14] | uni | uni | IND-CPA | DLWE |
| PRE | [PWA+16] | uni | uni | IND-CCA | LWE |
| PRE | [PRSV17] | uni | uni | IND-CPA | RLWE |
| PRE | Ours | uni | uni | IND-HRA | DDH |

Table 1: Summary of bi/unidirectional UE and PRE schemes. We focus on ciphertext-independent UE and multi-hop PRE. In this table, "bi, uni, bwd-uni, no" stand for bidirectional, unidirectional, backward-leak unidirectional, no-directional, respectively. We note that the notion of *key-directionality* differs for UE and PRE; in the case of UE, unidirectionality of key updates implies that, given the source (secret) key and the update token, the destination (secret) key can be computed. This is not the case for PRE, where unidirectional key update simply denotes that the re-key generation algorithm takes as input the source secret key and the destination public key (as opposed to bidirectional key update, where the re-key generation algorithm takes as input both secret keys).

### 1.2 Background and Related Work

There has been extensive research on both UE and PRE, including various settings, definitions, and constructions. Below we only mention works that are the most directly relevant. For both UE and PRE, we focus on the CPA-type definitions, which are by far the most well-studied notions.

**Security Notions for UE.** Since the introduction of UE in [BLMR13], several works have explored its security notions [EPRS17, LT18, AMP19, KLR19, BDGJ20, Jia20]. Most notable is the work of Lehmann and Tackmann [LT18], which improved the model and studied the notion of post-compromise security for UE. Their Indistinguishability of Update notion (IND-UPD) returns a challenge ciphertext $\mathsf{ct}^*$ which is either the re-encryption of a ciphertext $\mathsf{ct}_0$ or $\mathsf{ct}_1$. A scheme is IND-UPD secure if an adversary is unable to determine which of the ciphertexts was re-encrypted.

In subsequent works a stronger combined notion of IND-UE security has been used, first defined by Boyd et al. [BDGJ20]. The IND-UE notion requires an adversary be unable to distinguish between a fresh encryption of a plaintext $\mathsf{m}$ and the re-encryption of a ciphertext $\mathsf{ct}$. As a result this notion captures both CPA (specifically IND-ENC) and IND-UPD security.

**Security Notions for PRE.** In the context of PRE, the traditional notion of IND-CPA security [ID03, AFGH06] have been shown to be insufficient in practice. To address this, Cohen [Coh19] introduced the notion of Honest Re-Encryption Attack (HRA) security where an adversary is additionally permitted to re-encrypt (from honest to corrupt users) ciphertexts previously output by the encryption oracle. While only recently considered in the analysis of PRE, the essence of this notion is also fundamental in formalizing security for UE.

More recently, Davidson et al. [DDLM19] have investigated achieving post-compromise secure PRE schemes. They introduced a notion of IND-PCS security for PRE, which can be viewed as the analogue of IND-UPD security of UE in the context of PRE, albeit for more complex re-encryption graphs. To date this is the only paper that studies the PCS security of PRE schemes. Their work again demonstrates the challenges in constructing such schemes in the unidirectional setting. They discuss two PCS-secure constructions which are based on a prior unidirectional PRE scheme, Construction 7b of Fuchsbauer et al. [FKKP19] and an extension of BV-PRE [PRSV17].

**Updatable Public Key Encryption.** In order to achieve forward security in public key encryption (PKE), a notion called *updatable PKE (UPKE)* has recently been proposed and studied [JMM19,ACDT20,DKW21], where any sender (encryptor) can initiate a key update by sending a special update ciphertext to the receiver (decryptor). This ciphertext updates the public key and also, once processed by the receiver, will update its secret key. These are PKE schemes that encrypt messages under different public keys and aim to achieve forward security. In contrast, UE and PRE schemes studied in this paper aim to update ciphertexts encrypted under an old key to a new key without leaking the message content. The notions of UE/PRE as well as our techniques are very different from UPKE despite the partial naming collision.

**Comparison with Umbral.** There exists a practically deployed construction of unidirectional PRE, namely Umbral [Nun18], from the DDH Assumption, albeit in the random oracle model. It turns out that the Umbral construction is only single-hop, and focuses on achieving threshold PRE rather than multi-hop PRE. In particular, the Umbral construction crucially relies on the Diffie-Hellman key change, and it is unclear how to extend the construction to multiple hops. On the other hand, our primary aim is to achieve multi-hop unidirectional PRE in the traditional non-threshold setting. We note additionally that Umbral would not achieve post-compromise security, which is an important property provided by our constructions. Fundamentally, this is due to the fact that Umbral adopts a KEM-DEM style approach where only the KEM is re-encrypted.

**Concurrent Work.** A concurrent work by Galteland and Pan [GP23] constructs unidirectional UE with backward-leak unidirectional key update from public key encryption (PKE) schemes with certain properties, which can be realized from the DDH or LWE assumption. Their techniques are significantly different from ours and do not trivially extend to the PRE setting. The authors of [GP23] also demonstrate a formal proof that the security definition for unidirectional UE with backward-leak unidirectional key updates is equivalent to the one with no-directional key updates, which confirms our observation discussed earlier.

## 1.3 Technical Overview

In this section, we provide a high-level overview of our techniques for constructing unidirectional UE and PRE from any generic KPHE scheme satisfying the special properties described earlier.

**IND-ENC Secure UE.** Our first attempt is to build a unidirectional IND-ENC secure UE scheme, and we start with a naïve idea. Take an arbitrary symmetric-key encryption scheme and each epoch key is a freshly generated key of this encryption scheme. The update token $\Delta_\mathsf{e}$ from $\mathtt{k}_{\mathsf{e}-1}$ to $\mathtt{k}_\mathsf{e}$ is an encryption of $\mathtt{k}_{\mathsf{e}-1}$ under $\mathtt{k}_\mathsf{e}$, namely $\Delta_\mathsf{e} = \mathsf{Enc}_{\mathtt{k}_\mathsf{e}}(\mathtt{k}_{\mathsf{e}-1})$. When we update a ciphertext from epoch $(\mathsf{e}-1)$ to epoch $\mathsf{e}$, we just attach the update token $\Delta_\mathsf{e}$ to the end of the ciphertext. For a message $\mathsf{m}$ first encrypted in epoch $\mathsf{e}$ and then updated through epoch $\mathsf{e}'$, the resulting ciphertext is of the form:

$$\mathsf{ct}_{\mathsf{e}'} = \left(\mathsf{Enc}_{\mathtt{k}_\mathsf{e}}(\mathsf{m}), \mathsf{Enc}_{\mathtt{k}_{\mathsf{e}+1}}(\mathtt{k}_\mathsf{e}), \ldots, \mathsf{Enc}_{\mathtt{k}_{\mathsf{e}'}}(\mathtt{k}_{\mathsf{e}'-1})\right).$$

Given $\mathtt{k}_{\mathsf{e}'}$, one can easily decrypt $\mathsf{ct}_{\mathsf{e}'}$ layer by layer to recover $\mathsf{m}$.

This naïve approach does not achieve IND-ENC security. We show a concrete attack in the following. Let $\mathsf{e}^*$ be the challenge epoch and $\mathsf{m}^*$ be the challenge message queried by the adversary. Let $\mathsf{ct}_{\mathsf{e}^*} = \mathsf{Enc}_{\mathtt{k}_{\mathsf{e}^*}}(\mathsf{m}^*)$ be the challenge ciphertext. To extract the secret key $\mathtt{k}_{\mathsf{e}^*}$, the adversary proceeds as follows. It first queries for an encryption of an arbitrary message $\mathsf{m}$ in epoch 0 and then updates it to epoch $\mathsf{e}$ (for some $\mathsf{e} > \mathsf{e}^*$) via a sequence of update queries. This way the adversary obtains a ciphertext of $\mathsf{m}$ of the form:

$$\mathsf{ct}_\mathsf{e} = (\mathsf{Enc}_{\mathtt{k}_0}(\mathsf{m}), \mathsf{Enc}_{\mathtt{k}_1}(\mathtt{k}_0), \ldots, \mathsf{Enc}_{\mathtt{k}_\mathsf{e}}(\mathtt{k}_{\mathsf{e}-1})).$$

Now the adversary corrupts the secret key $\mathtt{k}_\mathsf{e}$. Then it can recover all the previous keys from $\mathtt{k}_0$ to $\mathtt{k}_{\mathsf{e}-1}$ (including $\mathtt{k}_{\mathsf{e}^*}$) during decryption of the ciphertext $\mathsf{ct}_\mathsf{e}$.

Nonetheless, this simple approach demonstrates some nice properties of unidirectionality. For key updates, it is impossible to derive $\mathtt{k}_\mathsf{e}$ from $\mathtt{k}_{\mathsf{e}-1}$ and $\Delta_\mathsf{e}$. For ciphertext updates, given a *fresh* ciphertext $\mathsf{ct}_\mathsf{e}$ in epoch $\mathsf{e}$ and the previous update token $\Delta_\mathsf{e}$ (from epoch $(\mathsf{e}-1)$ to $\mathsf{e}$), it is impossible to transition the ciphertext $\mathsf{ct}_\mathsf{e}$ to the previous epoch $\mathsf{ct}_{\mathsf{e}-1}$ (i.e. the epoch prior to its existence). In fact, Cohen [Coh19] applied this idea to PRE and showed a CPA-secure but not HRA-secure PRE scheme (HRA security is inherently required in IND-ENC UE schemes).

**Re-randomizing the Secret Keys.** The problem with these chained ciphertexts is that during decryption of a single ciphertext, all the previous secret keys are also leaked. To resolve this problem, our hope is to somehow re-randomize all the previous secret keys in the chain, in a consistent and homomorphic manner. In particular, we want the ciphertext to be of the form

$$\mathsf{ct}_\mathsf{e} = \left(\mathsf{Enc}_{\overline{\mathtt{k}}_0}(\mathsf{m}), \mathsf{Enc}_{\overline{\mathtt{k}}_1}(\overline{\mathtt{k}}_0), \ldots, \mathsf{Enc}_{\overline{\mathtt{k}}_{\mathsf{e}-1}}(\overline{\mathtt{k}}_{\mathsf{e}-2}), \mathsf{Enc}_{\mathtt{k}_\mathsf{e}}(\overline{\mathtt{k}}_{\mathsf{e}-1})\right),$$

where $\overline{\mathtt{k}}_0, \overline{\mathtt{k}}_1, \ldots, \overline{\mathtt{k}}_{\mathsf{e}-1}$ are all re-randomized secret keys that are different for each ciphertext. During the decryption of $\mathsf{ct}_\mathsf{e}$, only these re-randomized secret keys are leaked, which does not affect the security of other ciphertexts.

To enable such re-randomization, our idea is inspired by the re-randomizable Yao's garbled circuits [GHV10]. We propose a new primitive called Key and Plaintext Homomorphic Encryption (KPHE), which can be seen as a generalization of the circular secure encryption scheme of Boneh et al. [BHHO08]. Instead of using an arbitrary symmetric-key encryption scheme, we use the KPHE scheme for encryption, where the UE secret key $\mathtt{k}_\mathsf{e}$ is a key pair $(\mathsf{pk}_\mathsf{e}, \mathsf{sk}_\mathsf{e})$ of the KPHE scheme. The update token is a KPHE encryption of the previous epoch's secret key under the current epoch's public key, namely $\Delta_\mathsf{e} = \mathsf{KPHE.Enc}_{\mathsf{pk}_\mathsf{e}}(\mathsf{sk}_{\mathsf{e}-1})$.

To update a ciphertext we exploit the two homomorphism properties of the KPHE scheme, in both the message space and the key space. Given an update token $\Delta_\mathsf{e}$ and a ciphertext of the form

$$\mathsf{ct}_{\mathsf{e}-1} = \left(\mathsf{KPHE.Enc}_{\overline{\mathsf{pk}}_0}(\mathsf{m}), \mathsf{KPHE.Enc}_{\overline{\mathsf{pk}}_1}(\overline{\mathsf{sk}}_0), \ldots, \mathsf{KPHE.Enc}_{\mathsf{pk}_{\mathsf{e}-1}}(\overline{\mathsf{sk}}_{\mathsf{e}-2})\right),$$

we focus on the last component $\mathsf{ctx} = \mathsf{KPHE.Enc}_{\mathsf{pk}_{\mathsf{e}-1}}(\overline{\mathsf{sk}}_{\mathsf{e}-2})$ and the update token $\Delta_\mathsf{e} = \mathsf{KPHE.Enc}_{\mathsf{pk}_\mathsf{e}}(\mathsf{sk}_{\mathsf{e}-1})$, In our update operation we first generate a random permutation $\pi$ and then perform two important steps:

- Use the KPHE key-space homomorphism to transform $\mathsf{ctx}$ from an encryption under $\mathsf{sk}_{\mathsf{e}-1}$ to an encryption under $\pi(\mathsf{sk}_{\mathsf{e}-1})$.
- Use the KPHE message-space homomorphism to transform $\Delta_{\mathsf{e}}$ from an encryption of $\mathsf{sk}_{\mathsf{e}-1}$ to an encryption of $\pi(\mathsf{sk}_{\mathsf{e}-1})$.

The updated ciphertext becomes

$$\mathsf{ct}_{\mathsf{e}} = \Big(\mathsf{KPHE.Enc}_{\overline{\mathsf{pk}}_0}(\mathsf{m}), \mathsf{KPHE.Enc}_{\overline{\mathsf{pk}}_1}(\overline{\mathsf{sk}}_0), \ldots, \mathsf{KPHE.Enc}_{\overline{\mathsf{pk}}_{\mathsf{e}-1}}(\overline{\mathsf{sk}}_{\mathsf{e}-2}), \mathsf{KPHE.Enc}_{\mathsf{pk}_{\mathsf{e}}}(\overline{\mathsf{sk}}_{\mathsf{e}-1})\Big),$$

where $\overline{\mathsf{sk}}_{\mathsf{e}-1} = \pi(\mathsf{sk}_{\mathsf{e}-1})$ (with corresponding public key $\overline{\mathsf{pk}}_{\mathsf{e}-1}$). In our construction, the KPHE secret key is a $2n$-bit string, which is randomly sampled with exactly $n$ bits of 0 and $n$ bits of 1. The affine transformation $\pi$ is a random permutation on the $2n$ bits of the string. By transforming from $\mathsf{sk}_{\mathsf{e}-1}$ to $\overline{\mathsf{sk}}_{\mathsf{e}-1}$ we ensure that a fresh secret key is used for each update operation and hence there is appropriate isolation between all ciphertexts updated in a given epoch. The blinding property of KPHE ensures that re-randomization can be done without knowledge of the underlying secret keys, and that the re-randomized ciphertexts are computationally indistinguishable from freshly generated ciphertexts.

**Use of Balanced KPHE Keys.** The astute reader might have noticed that we use "balanced" secret keys for our KPHE scheme, wherein each secret key is a randomly sampled $2n$-bit string with exactly $n$ bits of 0 and $n$ bits of 1. The restriction is required to offset some leakage that our scheme incurs during the honest re-encryption query phase in the security proofs. Informally, the adversary can use a sequence of honest re-encryption queries to learn some information about the intermediate (re-randomized) secret keys; in particular, it learns the number of 0 and 1 bits in each secret key. Intuitively, we offset this leakage by specifying at setup that all secret keys have an equal number of 0 and 1 entries. As a result, the adversary learns no additional information about these intermediate keys, irrespective of the number of honest re-encryption queries that it issues. We defer a formal treatment to the detailed proofs of security for our constructions.

**Achieving Post-Compromise Secure UE.** We can extend the IND-ENC secure UE construction to achieve post-compromise security. To achieve IND-UPD security, we can modify the update operation to ensure that all the chained ciphertexts are updated (rather than just the last one). In effect what our enhanced construction does is again exploit properties of the KPHE scheme to re-randomize each of the ciphertext components. This ensures that two updated ciphertexts of the same length are computationally indistinguishable. To further achieve the combined IND-UE security, we need to additionally guarantee that a freshly generated ciphertext has the same length as an updated ciphertext in a certain epoch. More details on our UE constructions are given in Section 3.

**Achieving Unidirectional PRE.** We can use the same high-level approach to construct a unidirectional PRE scheme, where a ciphertext consists of a chain of KPHE ciphertexts, and re-encryption exploits the two KPHE homomorphisms to transform each new KPHE ciphertext to a fresh secret key. The crucial subtlety in the PRE case, which makes proving security slightly more involved, is that we no longer consider sequential ciphertext updates but must consider re-encryption between all possible key pairs. As a result we need to further exploit the circular security properties of the KPHE scheme to prove security. This is further detailed in Section 4.

**Connections between UE and PRE.** Generally speaking, unidirectional PRE can be viewed as a stronger primitive than unidirectional UE because UE only allows for sequential updates while PRE allows for re-encryption between every pair of keys. In fact, we observe that if we treat the public-secret key pair of PRE as a secret key for UE, and the PRE re-encryption key as an update token for UE, then IND-HRA secure PRE implies IND-ENC secure UE, and IND-PCS secure PRE implies IND-UPD secure UE. This is also why our constructions for unidirectional UE and PRE follow a very similar framework. On the other hand, since PRE supports re-encryption between (potentially) every pair of keys, our constructions of PRE require stronger security guarantees (in particular, circular security) from the underlying KPHE scheme.

**Efficiency and Feasibility.** We acknowledge that the ciphertext length in our UE/PRE constructions grows linearly with the number of epochs/re-encryption hops, unlike certain existing constructions (e.g. in [Nis21, PWA+16, PRSV17]) where the ciphertext size remains the same. In this context, we emphasize that our paper is the first to achieve backward-leak unidirectional UE and unidirectional PRE from standard assumptions, specifically DDH. It has been a long-standing open problem for over a decade whether obfuscation/FHE is necessary for unidirectional UE/(multi-hop) PRE, and our work closes this assumption gap. As a result, we believe that our results should be viewed with emphasis on the new theoretical insights/understanding into unidirectional-UE/PRE that they enable as opposed to concrete efficiency. Our work opens up the discussion of whether obfuscation/FHE is necessary for achieving unidirectional UE/PRE with "succinctness" in the ciphertext length.

We note that in the UE setting, key rotation may only happen a small number of times in practice. For example, once a year for the lifetime of the ciphertext (say 10years). Thus, taking a similar approach to [BEKS20] (from the ciphertext-dependent setting) we could bound the number of updates and have fixed-length ciphertexts through some form of padding. We also point out that while the size of ciphertexts in our general constructions grow linearly, the secret keys and update tokens/re-encryption keys remain constant-sized. We also note that for the basic versions of our UE/PRE construction (IND-CPA unidirectional UE and IND-HRA secure unidirectional PRE), the work done per update/re-encryption operation is also constant (independent of the number of epochs/update hops).

We note here that a naïve approach to achieving unidirectional UE is the so called "download–decrypt–re-encrypt–upload" approach, where the client downloads the encrypted data (e.g. from the server storing the encrypted data), locally decrypts it, re-encrypts it using the new key, and re-uploads the newly encrypted data to the server. Our UE constructions are non-trivial in the sense that we achieve significantly better properties as compared to this naïve approach. In particular, for applications of UE (e.g. key rotation) where the client outsources encrypted data to the server, this entails constant computational/ communication/storage overheads at the client during key rotation (the client simply generates and sends the update token to the server); the corresponding client-overheads are linear (in database size) in the naïve solution.

**Using Random Oracles.** A possible approach towards achieving practical efficiency is to use random oracles (such as in the single-hop unidirectional threshold PRE scheme Umbral [Nun18]). Our focus is primarily on feasibility results for unidirectional UE/PRE in the standard model, and we consciously avoid the use of random oracles. We also point out that a previous result [AMP19] showed that, even in the symmetric-key setting, unidirectional UE/PRE implies public-key encryption, and so a construction from just a random oracle is unlikely. However, it might be possible to achieve efficiency gains using a random oracle. We leave investigating such a random oracle-based construction of unidirectional UE/PRE as an interesting direction of future research.

### 1.4 Paper Outline

The rest of the paper is organized as follows. Section 2 formally defines a KPHE scheme and its associated security properties. Section 3 presents our constructions of IND-CPA and IND-UPD secure UE from any KPHE scheme (parts of the corresponding proofs of security are detailed in Appendices A and B). Section 4 presents our constructions of IND-HRA secure PRE and IND-PCS secure PRE from any KPHE scheme (the corresponding proofs of security are detailed in Appendices C and D, respectively). Finally, Section 5 describes how to instantiate KHPE from DDH (here, we mostly rely on known results from the literature).

For readers not familiar with the formal definitions of UE and PRE, we present relatively self-contained background material on UE and PRE in Sections 3.1 and 4.1, respectively.

### 1.5 Notations

We summarize here the notations used in the rest of the paper. We write $x \xleftarrow{\$} \chi$ to represent that an element $x$ is sampled randomly from a set/distribution $\mathcal{X}$. The output $x$ of a deterministic (resp. randomized) algorithm $\mathcal{A}$ is denoted by $x = \mathcal{A}$ (resp. $x \xleftarrow{\$} \mathcal{A}$). For $a \in \mathbb{N}$ such that $a \geq 1$, we denote by $[a]$ the set of integers lying between 1 and $a$ (both inclusive). We refer to $\lambda \in \mathbb{N}$ as the security parameter, and denote by $\text{poly}(\lambda)$ and $\text{negl}(\lambda)$ any generic (unspecified) polynomial function and negligible function in $\lambda$, respectively.[1]

## 2 Key and Plaintext Homomorphic Encryption

In this section, we present the definitions for the core building block for our constructions, namely key and plaintext homomorphic encryption (KPHE). Informally, a KPHE scheme has the following features:

- **Keys and Plaintexts:** Each secret key $\mathsf{sk}$ is an $\ell$-bit string for some $\ell = \text{poly}(\lambda)$ ($\lambda$ being the security parameter). Additionally, each plaintext message $\mathsf{m}$ is an $\ell'$-bit string for some $\ell' = \text{poly}(\lambda)$.

- **Key Distribution:** Each secret key is sampled according to some distribution $\mathcal{D}$ over $\{0,1\}^{\ell}$. In particular, for our applications, we assume KPHE schemes where each secret key $\mathsf{sk}$ is a $2n$-bit string with equally many 0 and 1 entries.

- **Key Homomorphism:** Let $T$ be any linear transformation that maps $\ell$-bit strings to $\ell$-bit strings. Then, it is possible to efficiently evaluate the following:

[1] Note that a function $f : \mathbb{N} \rightarrow \mathbb{N}$ is said to be negligible in $\lambda$ if for every positive polynomial $p$, $f(\lambda) < 1/p(\lambda)$ when $\lambda$ is sufficiently large.

- Given a public key $\mathsf{pk}$ corresponding to some secret key $\mathsf{sk} \in \{0,1\}^\ell$, it is possible to efficiently compute a valid public key $\mathsf{pk}'$ corresponding to the transformed secret key $\mathsf{sk}' = T(\mathsf{sk})$, without the knowledge of $\mathsf{sk}$.
- Given a ciphertext $\mathsf{ct}$ encrypting a message $\mathsf{m}$ under some secret key $\mathsf{sk} \in \{0,1\}^\ell$, it is possible to efficiently compute a ciphertext $\mathsf{ct}'$ encrypting the same message $\mathsf{m}$ under the transformed secret key $\mathsf{sk}' = T(\mathsf{sk})$, without the knowledge of $\mathsf{sk}$.

- **Plaintext Homomorphism:** Let $T'$ be any linear transformation that maps $\ell'$-bit strings to $\ell'$-bit strings. Then, given a ciphertext $\mathsf{ct}$ encrypting a message $\mathsf{m} \in \{0,1\}^{\ell'}$ under some secret key $\mathsf{sk}$, it is possible to efficiently compute a ciphertext $\mathsf{ct}''$ encrypting the transformed message $\mathsf{m}' = T'(\mathsf{m})$ under the same secret key $\mathsf{sk}$.

We now summarize these features of KPHE formally below.

**Definition 2.1** (KPHE). *A KPHE scheme is a tuple of PPT algorithms of the form* $\mathsf{KPHE} = (\mathsf{Setup}, \mathsf{SKGen}, \mathsf{PKGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *that are defined as follows:*

- $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*: On input the security parameter* $\lambda$*, the setup algorithm outputs a public parameter* $\mathsf{pp}$*.*

- $\mathsf{sk} \xleftarrow{\$} \mathsf{SKGen}(\mathsf{pp}, \mathcal{D})$*: On input the public parameter* $\mathsf{pp}$ *and a distribution* $\mathcal{D}$ *over* $\{0,1\}^\ell$ *(for* $\ell = \mathrm{poly}(\lambda)$*), the secret key generation algorithm outputs a secret key* $\mathsf{sk} \xleftarrow{\$} \mathcal{D}$*.*

- $\mathsf{pk} \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk})$*: On input the public parameter* $\mathsf{pp}$ *and a secret key* $\mathsf{sk} \in \{0,1\}^\ell$*, the public key generation algorithm outputs a public key* $\mathsf{pk}$*.*

- $\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$*: On input a public key* $\mathsf{pk}$ *and a message* $\mathsf{m} \in \{0,1\}^{\ell'}$ *(for* $\ell' = \mathrm{poly}(\lambda)$*), the encryption algorithm outputs a ciphertext* $\mathsf{ct}$*.*

- $\mathsf{m}/\bot \xleftarrow{\$} \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$*: On input a secret key* $\mathsf{sk} \in \{0,1\}^\ell$ *and a ciphertext* $\mathsf{ct}$*, the decryption algorithm outputs a plaintext message string* $\mathsf{m}$ *or an error symbol* $\bot$*.*

- $(\mathsf{pk}', \mathsf{ct}') \xleftarrow{\$} \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, T, T')$*: On input a public key* $\mathsf{pk}$*, a ciphertext* $\mathsf{ct}$*, and a pair of (linear) transformations* $T : \{0,1\}^\ell \to \{0,1\}^\ell$ *and* $T' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell'}$*, the homomorphic evaluation algorithm outputs a tuple consisting of a transformed public key and a transformed ciphertext* $(\mathsf{pk}', \mathsf{ct}')$*.*

**Correctness.** A KPHE scheme $(\mathsf{Setup}, \mathsf{SKGen}, \mathsf{PKGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is said to be correct with respect to a distribution $\mathcal{D}$ over $\{0,1\}^\ell$ if for any $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$, any $\mathsf{sk} \xleftarrow{\$} \mathsf{SKGen}(\mathsf{pp}, \mathcal{D})$, any $\mathsf{pk} \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk})$, any $\mathsf{m} \in \{0,1\}^{\ell'}$, and any pair of (linear) transformations $T : \{0,1\}^\ell \to \{0,1\}^\ell$ and $T' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell'}$, letting $\mathsf{sk}' = T(\mathsf{sk})$, $\mathsf{m}' = T'(\mathsf{m})$ and

$$\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m}), \quad (\mathsf{pk}', \mathsf{ct}') \xleftarrow{\$} \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, T, T'),$$

*both* of the following hold with overwhelmingly large probability:

**Experiment** $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DSS-KPHE}}(\lambda, \mathcal{A})$:

1. The challenger generates $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$, $\mathsf{sk} \xleftarrow{\$} \mathsf{SKGen}(\mathsf{pp}, \mathcal{D})$, and $\mathsf{pk} \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk})$, and provides the adversary $\mathcal{A}$ with $(\mathsf{pp}, \mathsf{pk})$.
2. The adversary $\mathcal{A}$ issues a challenge encryption query for a pair of messages $(\mathsf{m}_0, \mathsf{m}_1)$.
3. The challenger samples $b \xleftarrow{\$} \{0, 1\}$, creates the challenge ciphertext

$$\mathsf{ct}^* \xleftarrow{\$} \mathsf{Enc}\left(\mathsf{pk}, \mathsf{m}_b\right),$$

   and sends $\mathsf{ct}^*$ to the adversary $\mathcal{A}$.
4. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.
5. Output 1 if $b = b'$ and 0 otherwise.

Figure 1: The $\mathcal{D}$-Semantic Security Experiment for KPHE

- $\mathsf{pk}'$ is a valid public key with respect to $\mathsf{sk}' = T(\mathsf{sk})$, i.e., for any $\bar{\mathsf{m}} \in \{0,1\}^{\ell'}$, it holds that $\mathsf{Dec}(\mathsf{sk}', \mathsf{Enc}(\mathsf{pk}', \bar{\mathsf{m}})) = \bar{\mathsf{m}}$.
- $\mathsf{ct}'$ is a valid encryption of $\mathsf{m}'$ under $(\mathsf{pk}', \mathsf{sk}')$, i.e., $\mathsf{Dec}(\mathsf{sk}', \mathsf{ct}') = \mathsf{m}'$.

**Distributional Semantic Security.** We (informally) say that a KPHE satisfies distributional semantic security with respect to some distribution $\mathcal{D}$ over $\{0,1\}^\ell$ if it remains semantically secure even when the secret key $\mathsf{sk}$ is sampled according to the distribution $\mathcal{D}$. Formally, this is modeled using a semantic security game where the secret key is sampled by the challenger as per the distribution $\mathcal{D}$.

**Definition 2.2** ($\mathcal{D}$-Semantic Security)**.** *A KPHE scheme with $\ell$-bit secret keys is said to be $\mathcal{D}$-semantically secure with respect to a distribution $\mathcal{D}$ over $\{0,1\}^\ell$ if for any security parameter $\lambda \in \mathbb{N}$ and any PPT adversary $\mathcal{A}$, the following holds with overwhelmingly large probability:*

$$\mid \Pr[\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DSS-KPHE}}(\lambda, \mathcal{A}) = 1] - 1/2 \mid < \mathrm{negl}(\lambda),$$

*where the experiment $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DSS-KPHE}}(\lambda, \mathcal{A})$ is as defined in Figure 1.*

**Distributional Circular Security.** We (informally) say that a KPHE satisfies distributional circular security with respect to some distribution $\mathcal{D}$ over $\{0,1\}^\ell$ if it satisfies the standard notion of circular security [CL01, BRS02, BHHO08, ACPS09] even when each secret key is sampled from the distribution $\mathcal{D}$. Formally, this is modeled using a circular security game where the secret keys are sampled by the challenger as per the distribution $\mathcal{D}$.

**Definition 2.3** ($\mathcal{D}$-Circular Security)**.** *A KPHE scheme with $\ell$-bit secret keys and $\ell$-bit messages is said to be $\mathcal{D}$-circular secure with respect to a distribution $\mathcal{D}$ over $\{0,1\}^\ell$ if for any security parameter $\lambda \in \mathbb{N}$ and any PPT adversary $\mathcal{A}$, the following holds with overwhelmingly large probability:*

$$\mid \Pr[\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DCC-KPHE}}(\lambda, \mathcal{A}) = 1] - 1/2 \mid < \mathrm{negl}(\lambda),$$

**Experiment** $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DCC-KPHE}}(\lambda, \mathcal{A})$:

1. The challenger generates $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^{\lambda})$ and provides it to the adversary.
2. The adversary $\mathcal{A}$ outputs $n = \text{poly}(\lambda)$.
3. The challenger samples $\mathsf{sk}_1, \ldots, \mathsf{sk}_n \xleftarrow{\$} \mathsf{SKGen}(\mathsf{pp}, \mathcal{D})$, sets
$$\mathsf{pk}_1 \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk}_1), \ldots, \mathsf{pk}_n \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk}_n),$$
and provides $(\mathsf{pk}_1, \ldots, \mathsf{pk}_n)$ to the adversary $\mathcal{A}$.
4. The challenger also sets the following for each $i, j \in [n]$:
$$\mathsf{ct}_{i,j,0} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, \mathsf{sk}_j), \quad \mathsf{ct}_{i,j,1} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, 0^{|\mathsf{sk}_j|}).$$
5. The challenger finally samples a bit $b \xleftarrow{\$} \{0,1\}$ and provides the adversary $\mathcal{A}$ with the ensemble $\{\mathsf{ct}_{i,j,b}\}_{i,j \in [n]}$.
6. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.
7. Output 1 if $b = b'$ and 0 otherwise.

Figure 2: The $\mathcal{D}$-Circular Security Experiment for KPHE

*where the experiment* $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{DCC-KPHE}}(\lambda, \mathcal{A})$ *is as defined in Figure 2.*

**Blinding.** We (informally) say that a KPHE scheme satisfies public key and ciphertext blinding if the homomorphic evaluation algorithm outputs a public key-ciphertext pair $(\mathsf{pk}', \mathsf{ct}')$ corresponding to the transformed secret key $\mathsf{sk}'$ and the transformed message $\mathsf{m}'$ such that:

- The transformed public key $\mathsf{pk}'$ is computationally indistinguishable from a public key sampled uniformly at random from the set of all valid public keys corresponding to the secret key $\mathsf{sk}'$.
- The transformed ciphertext $\mathsf{ct}'$ is computationally indistinguishable from a ciphertext sampled uniformly at random from the set of all valid ciphertexts corresponding to the transformed message $\mathsf{m}'$ under $\mathsf{pk}'$.

More formally, we define this blinding property as follows.

**Definition 2.4** (Blinding)**.** *A KPHE scheme with $\ell$-bit secret keys and $\ell'$-bit messages is said to satisfy blinding security with respect to a distribution $\mathcal{D}$ over $\{0,1\}^{\ell}$ if for any security parameter $\lambda \in \mathbb{N}$ and any PPT adversary $\mathcal{A}$, the following holds with overwhelmingly large probability:*

$$|\Pr[\mathsf{Expt}_{\mathcal{D}}^{\mathsf{Blind-KPHE}}(\lambda, \mathcal{A}) = 1] - 1/2 | < \text{negl}(\lambda),$$

*where the experiment* $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{Blind-KPHE}}(\lambda, \mathcal{A})$ *is as defined in Figure 3.*

**Experiment** $\mathsf{Expt}_{\mathcal{D}}^{\mathsf{Blind-KPHE}}(\lambda, \mathcal{A})$:

1. The challenger generates $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$, $\mathsf{sk} \xleftarrow{\$} \mathcal{D}$, and $\mathsf{pk} \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk})$, and provides the adversary $\mathcal{A}$ with $(\mathsf{pp}, \mathsf{sk}, \mathsf{pk})$.
2. The adversary $\mathcal{A}$ sends a message $\mathsf{m} \in \{0,1\}^{\ell'}$ to the challenger.
3. The challenger responds to $\mathcal{A}$ with a ciphertext $\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$.
4. The adversary $\mathcal{A}$ then sends a pair of (linear) transformations
$$T : \{0,1\}^\ell \to \{0,1\}^\ell, \quad T' : \{0,1\}^{\ell'} \to \{0,1\}^{\ell'}.$$
5. The challenger sets
$$\mathsf{sk}' = T(\mathsf{sk}), \quad \mathsf{m}' = T'(\mathsf{m}),$$
and computes the following:
$$(\mathsf{pk}_0, \mathsf{ct}_0) \xleftarrow{\$} \mathsf{Eval}(\mathsf{pk}, \mathsf{ct}, T, T'), \quad \mathsf{pk}_1 \xleftarrow{\$} \mathsf{PKGen}(\mathsf{pp}, \mathsf{sk}'), \quad \mathsf{ct}_1 \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_1, \mathsf{m}'),$$
6. The challenger finally samples a bit $b \xleftarrow{\$} \{0,1\}$ and provides the adversary $\mathcal{A}$ with $(\mathsf{pk}_b, \mathsf{ct}_b)$.
7. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.
8. Output 1 if $b = b'$ and 0 otherwise.

Figure 3: The Blinding Experiment for KPHE

**KPHE from DDH.** In Appendix 5, we show the following: assuming that the decisional Diffie-Hellman (DDH) assumption holds, there exists a KPHE scheme that satisfies essentially all of the aforementioned properties. Concretely, we prove the following (informal) theorem:

**Theorem 2.5** (Informal)**.** *Assuming DDH, there exists a KPHE scheme with $2n$-bit secret keys that satisfies distributional semantic security with respect to the distribution $\mathcal{U}_n$, distributional circular security with respect to the distribution $\mathcal{U}_n$, and blinding, as defined above.*

In particular, we rely on known results from [BHHO08, NS12, GHV10] for the DDH-based instantiation of KPHE. See Appendix 5 for details.

**KPHE from LWE.** In this paper, we do not explicitly describe a construction of KPHE from LWE since there already exist constructions of unidirectional UE/PRE from LWE [CCL+14, PWA+16, PRSV17, Nis21]. Our aim in this work is to close the gap between bidirectional and unidirectional constructions of UE/PRE in terms of assumptions, and so we choose to focus on the feasibility results from the DDH assumption.

We note, however, that constructing KPHE from LWE is a very interesting direction of future work. In particular, one needs to be careful during re-encryption, which potentially increases the level of noise in the ciphertext of the LWE-based encryption scheme and could leak extra information. For example, due to increase in the noise level during re-encryption, it is not straightforward to prove the ciphertext blinding property, which requires that a freshly created ciphertext and a re-encrypted ciphertext are distributed in an indistinguishable manner. This issue can be handled using noise flooding techniques, albeit at the cost of a larger ciphertext size.

**KPHE from Other Assumptions.** We also leave it as an interesting open question to construct KPHE from concrete hardness assumptions other than DDH or LWE (e.g., factorization-based assumptions or LPN). Given our results on achieving unidirectional UE/PRE from KPHE, such realizations of KPHE would immediately yield new constructions of unidirectional UE/PRE from these assumptions.

# 3 Unidirectional UE from KPHE

In this section, we show how to construct unidirectional UE satisfying various security notions (IND-ENC, IND-UPD and IND-UE) from any KPHE scheme.

## 3.1 Definition

**Definition 3.1.** *An updatable encryption (UE) scheme for message space $\mathcal{M}$ is a tuple of PPT algorithms* $\mathsf{UE} = (\mathsf{UE.setup}, \mathsf{UE.next}, \mathsf{UE.enc}, \mathsf{UE.upd}, \mathsf{UE.dec})$ *with the following syntax:*

- $\mathtt{k}_0 \xleftarrow{\$} \mathsf{UE.setup}(1^\lambda)$*: On input a security parameter $1^\lambda$, it returns a secret key $\mathtt{k}_\mathsf{e}$ for epoch* $\mathsf{e} = 0$.
- $(\mathtt{k}_{\mathsf{e}+1}, \Delta_{\mathsf{e}+1}) \xleftarrow{\$} \mathsf{UE.next}(\mathtt{k}_\mathsf{e})$*: On input a secret key $\mathtt{k}_\mathsf{e}$ for epoch $\mathsf{e}$, it outputs a new secret key $\mathtt{k}_{\mathsf{e}+1}$ and an update token $\Delta_{\mathsf{e}+1}$ for epoch $\mathsf{e}+1$.*
- $\mathsf{ct}_\mathsf{e} \xleftarrow{\$} \mathsf{UE.enc}(\mathtt{k}_\mathsf{e}, \mathsf{m})$*: On input a secret key $\mathtt{k}_\mathsf{e}$ for epoch $\mathsf{e}$ and a message $\mathsf{m} \in \mathcal{M}$, it outputs a ciphertext $\mathsf{ct}_\mathsf{e}$.*
- $\mathsf{ct}_{\mathsf{e}+1} \xleftarrow{\$} \mathsf{UE.upd}(\Delta_{\mathsf{e}+1}, \mathsf{ct}_\mathsf{e})$*: On input a ciphertext $\mathsf{ct}_\mathsf{e}$ from epoch $\mathsf{e}$ and the update token $\Delta_{\mathsf{e}+1}$, it returns the updated ciphertext $\mathsf{ct}_{\mathsf{e}+1}$.*
- $\mathsf{m}'/\bot \leftarrow \mathsf{UE.dec}(\mathtt{k}_\mathsf{e}, \mathsf{ct}_\mathsf{e})$*: On input a ciphertext $\mathsf{ct}_\mathsf{e}$ and a secret key $\mathtt{k}_\mathsf{e}$ of some epoch $\mathsf{e}$, it returns a message $\mathsf{m}'$ or $\bot$.*

*We stress that* $\mathsf{UE.next}$ *generates a new key along with an update token, which follows from the definition in the work of Lehmann and Tackmann [LT18]. In our constructions, the update token $\Delta_{\mathsf{e}+1}$ can also be generated from $\mathtt{k}_\mathsf{e}$ and $\mathtt{k}_{\mathsf{e}+1}$.*

**Definition 3.2** (Correctness)**.** *Let* $\mathsf{UE} = (\mathsf{UE.setup}, \mathsf{UE.next}, \mathsf{UE.enc}, \mathsf{UE.upd}, \mathsf{UE.dec})$ *be an updatable encryption scheme. We say* $\mathsf{UE}$ *is* correct *if for any $\mathsf{m} \in \mathcal{M}$, any $\mathtt{k}_0 \xleftarrow{\$} \mathsf{UE.setup}(1^\lambda)$, any sequence of $(\mathtt{k}_1, \Delta_1), \ldots, (\mathtt{k}_\mathsf{e}, \Delta_\mathsf{e})$ generated as $(\mathtt{k}_i, \Delta_i) \xleftarrow{\$} \mathsf{UE.next}(\mathtt{k}_{i-1})$ for all $i \in [\mathsf{e}]$, and for any $0 \leq \widehat{\mathsf{e}} \leq \mathsf{e}$, let $\mathsf{ct}_{\widehat{\mathsf{e}}} \xleftarrow{\$} \mathsf{UE.enc}(\mathtt{k}_{\widehat{\mathsf{e}}}, \mathsf{m})$ and $\mathsf{ct}_j \xleftarrow{\$} \mathsf{UE.upd}(\Delta_j, \mathsf{ct}_{j-1})$ for all $j = \widehat{\mathsf{e}}+1, \ldots, \mathsf{e}$, then* $\mathsf{UE.dec}(\mathtt{k}_\mathsf{e}, \mathsf{ct}_\mathsf{e}) = \mathsf{m}$.

```
Setup(1^λ):
  k_0 ←$ UE.setup(1^λ)
  e := 0; phase := 0
  L, L~, K, T, C ←$ ∅

O.enc(m):
  ct ←$ UE.enc(k_e, m)
  L := L ∪ {(e, ct)}
  return ct

O.next:
  e := e + 1
  (k_e, Δ_e) ←$ UE.next(k_{e−1})
  if phase = 1 then
    ct~_e ←$ UE.upd(Δ_e, ct~_{e−1})
    L~ := L~ ∪ {(e, ct~_e)}

O.upd(ct_{e−1}):
  if (e − 1, ct_{e−1}) ∉ L then
    return ⊥
  ct_e ←$ UE.upd(Δ_e, ct_{e−1})
  L := L ∪ {(e, ct_e)}
  return ct_e

O.corr(inp, ê) :
  if ê > e then
    return ⊥
  if inp = key then
    K := K ∪ {ê}
    return k_ê
  if inp = token then
    T := T ∪ {ê}
    return Δ_ê

O.chall-IND-ENC(m̄_0, m̄_1):
  if |m̄_0| ≠ |m̄_1| then
    return ⊥
  phase := 1; ẽ := e
  ct~_ẽ ←$ UE.enc(k_ẽ, m̄_b)
  C := C ∪ {ẽ}
  L~ := L~ ∪ {(ẽ, ct~_ẽ)}
  return ct~_ẽ

O.chall-IND-UPD(ct̄_0, ct̄_1) :
  if (e − 1, ct̄_0) ∉ L or (e − 1, ct̄_1) ∉ L or |ct̄_0| ≠ |ct̄_1| then
    return ⊥
  phase := 1; ẽ := e
  ct~_ẽ ←$ UE.upd(Δ_ẽ, ct̄_b)
  C := C ∪ {ẽ}
  L~ := L~ ∪ {(ẽ, ct~_ẽ)}
  return ct~_ẽ

O.chall-IND-UE(m̄, ct̄) :
  if (e − 1, ct̄) ∉ L then
    return ⊥
  phase := 1; ẽ := e
  if b = 0 then
    ct~_ẽ ←$ UE.enc(k_ẽ, m̄)
  else
    ct~_ẽ ←$ UE.upd(Δ_ẽ, ct̄)
  C := C ∪ {ẽ}
  L~ := L~ ∪ {(ẽ, ct~_ẽ)}
  return ct~_ẽ

O.updC~ :
  if phase = 0 then
    return ⊥
  C := C ∪ {e}
  return ct~_e
```

Figure 4: Oracles in security games for updatable encryption.

**Confidentiality.** The adversary $\mathcal{A}$ has access to the oracles defined in Figure 4. We follow the bookkeeping techniques of [LT18, KLR19, BDGJ20, Jia20], using the following sets to keep track of the generated and updated ciphertexts, and the epochs in which the adversary corrupted a key or a token, or learned a version of the challenge-ciphertext.

- $\mathcal{L}$: Set of non-challenge ciphertexts $(\mathsf{e}, \mathsf{ct}_\mathsf{e})$ produced by calls to the $\mathcal{O}.\mathsf{enc}$ or $\mathcal{O}.\mathsf{upd}$ oracle. $\mathcal{O}.\mathsf{upd}$ only updates ciphertexts obtained in $\mathcal{L}$.
- $\widetilde{\mathcal{L}}$: Set of updated versions of the challenge ciphertexts $(\mathsf{e}, \widetilde{\mathsf{ct}}_\mathsf{e})$. $\widetilde{\mathcal{L}}$ is initiated with the challenge ciphertext $(\tilde{\mathsf{e}}, \widetilde{\mathsf{ct}}_{\tilde{\mathsf{e}}})$. Any call to the $\mathcal{O}.\mathsf{next}$ oracle automatically updates the challenge ciphertext to the new epoch, which the adversary can fetch via a call to $\mathcal{O}.\mathsf{upd}\widetilde{\mathsf{C}}$.
- $\mathcal{K}$: Set of epochs $\mathsf{e}$ in which the adversary corrupted the secret key $\mathtt{k}_\mathsf{e}$ (from $\mathcal{O}.\mathsf{corr}$).

- $\mathcal{T}$: Set of epochs $\mathsf{e}$ in which the adversary corrupted the update token $\Delta_{\mathsf{e}}$ (from $\mathcal{O}.\mathsf{corr}$).
- $\mathcal{C}$: Set of epochs $\mathsf{e}$ in which the adversary learned a version of the challenge ciphertext (from $\mathcal{O}.\mathsf{chall}$ or $\mathcal{O}.\mathsf{upd}\widetilde{\mathsf{C}}$).

We further define the epoch identification sets $\mathcal{C}^*, \mathcal{K}^*, \mathcal{T}^*$ as the extended sets of $\mathcal{C}, \mathcal{K}, \mathcal{T}$ in which the adversary learned or inferred information. We focus on *no-directional* key updates and *uni-directional* ciphertext updates.

$$\begin{aligned}
&\mathcal{K}^* := \mathcal{K}\\
&\mathcal{T}^* := \{\mathsf{e} \in \{0, \ldots, \mathsf{e}_{\mathsf{end}}\} | (\mathsf{e} \in \mathcal{T}) \vee (\mathsf{e}-1 \in \mathcal{K}^* \wedge \mathsf{e} \in \mathcal{K}^*)\}\\
&\mathcal{C}^* := \{\mathsf{e} \in \{0, \ldots, \mathsf{e}_{\mathsf{end}}\} | \mathsf{ChallEq}(\mathsf{e}) = \mathsf{true}\}\\
&\quad \text{where } \mathsf{true} \leftarrow \mathsf{ChallEq}(\mathsf{e}) \Longleftrightarrow (\mathsf{e} \in \mathcal{C}) \vee (\mathsf{ChallEq}(\mathsf{e}-1) \wedge \mathsf{e} \in \mathcal{T}^*)
\end{aligned}$$

**Remark 3.3.** *The constructions we present later will in fact permit* backward-leak key updates*. At first glance the* backward-leak key updates *notion proposed by Nishimaki [Nis21] is seemingly weaker than* no-directionality key updates*. However, as mentioned in the introduction, this notion is essentially equivalent to no-directional key updates because backward-leak derivation of* $\mathtt{k}_{\mathsf{e}-1}$ *does not increase the adversary's advantage in breaking the scheme. In particular, if the adversary obtains a challenge ciphertext* $\widetilde{\mathsf{ct}}_{\mathsf{e}-1}$ *and corrupts* $\Delta_{\mathsf{e}}$ *and* $\mathtt{k}_{\mathsf{e}}$*, then it does* not *matter if the adversary can derive* $\mathtt{k}_{\mathsf{e}-1}$ *or not, as it can always update the ciphertext to* $\widetilde{\mathsf{ct}}_{\mathsf{e}}$ *and decrypt it using* $\mathtt{k}_{\mathsf{e}}$*.*

**Definition 3.4** (IND-ENC, IND-UPD, IND-UE security)**.** *Let* $\mathsf{UE} = (\mathsf{UE.setup}, \mathsf{UE.next}, \mathsf{UE.enc}, \mathsf{UE.upd}, \mathsf{UE.dec})$ *be an updatable encryption scheme. We say* UE *is* notion*-secure for* $\mathsf{notion} \in \{\mathsf{IND\text{-}ENC}, \mathsf{IND\text{-}UPD}, \mathsf{IND\text{-}UE}\}$ *if for all PPT adversary* $\mathcal{A}$ *it holds that*

$$\left| \Pr\left[ \mathsf{Exp}^{\mathsf{notion}}_{\mathcal{A},\mathsf{UE}}(1^\lambda) = 1 \right] - \frac{1}{2} \right| \leq \mathrm{negl}(\lambda)$$

*for some negligible function* $\mathrm{negl}(\cdot)$*.*

**Experiment** $\mathsf{Exp}^{\mathsf{notion}}_{\mathcal{A},\mathsf{UE}}(1^\lambda)$:

Run $\mathsf{Setup}(1^\lambda)$
$(\mathsf{state}, \mathsf{Chall}_0, \mathsf{Chall}_1) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}.\mathsf{enc}, \mathcal{O}.\mathsf{next}, \mathcal{O}.\mathsf{upd}, \mathcal{O}.\mathsf{corr}}(1^\lambda)$
$\mathrm{b} \xleftarrow{\$} \{0,1\}$
$\widetilde{\mathsf{ct}} \xleftarrow{\$} \mathcal{O}.\mathsf{chall\text{-}notion}(\mathsf{Chall}_0, \mathsf{Chall}_1)$
Proceed only if $\widetilde{\mathsf{ct}} \neq \perp$
$\mathrm{b}' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}.\mathsf{enc}, \mathcal{O}.\mathsf{next}, \mathcal{O}.\mathsf{upd}, \mathcal{O}.\mathsf{corr}, \mathcal{O}.\mathsf{upd}\widetilde{\mathsf{C}}}(\mathsf{state}, \widetilde{\mathsf{ct}})$
**return** 1 if $\mathrm{b} = \mathrm{b}'$ and $\mathcal{C}^* \cap \mathcal{K}^* = \emptyset$

### 3.2 IND-ENC Secure Unidirectional UE

We begin by showing that any KPHE scheme with $2n$-bit secret keys that satisfies distributional semantic security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2 implies an IND-ENC secure unidirectional UE scheme.

**Construction.** Given a KHPE scheme of the form

$$\mathsf{KPHE} = (\mathsf{KPHE.Setup}, \mathsf{KPHE.SKGen}, \mathsf{KPHE.PKGen}, \mathsf{KPHE.Enc}, \mathsf{KPHE.Dec}, \mathsf{KPHE.Eval}),$$

with $2n$-bit secret keys, we construct a unidirectional UE scheme

$$\mathsf{UE} = (\mathsf{UE.setup}, \mathsf{UE.next}, \mathsf{UE.enc}, \mathsf{UE.upd}, \mathsf{UE.dec}),$$

with message space $\mathcal{M} = \{0,1\}^{2n}$ as follows:

- $\mathsf{UE.setup}(1^\lambda)$: Generate $\mathsf{pp} \xleftarrow{\$} \mathsf{KPHE.Setup}(1^\lambda)$, $\mathsf{sk}_0 \xleftarrow{\$} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n)$, and output
$$\mathtt{k}_0 = (\mathsf{pp}, \mathsf{sk}_0).$$

- $\mathsf{UE.next}(\mathtt{k}_\mathsf{e})$: Parse $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$. Generate $\mathsf{sk}_{\mathsf{e}+1} \xleftarrow{\$} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n)$ and $\mathsf{pk}_{\mathsf{e}+1} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_{\mathsf{e}+1})$. Output
$$\mathtt{k}_{\mathsf{e}+1} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}+1}), \quad \Delta_{\mathsf{e}+1} = (\mathsf{pk}_{\mathsf{e}+1}, \mathsf{KPHE.Enc}(\mathsf{pk}_{\mathsf{e}+1}, \mathsf{sk}_\mathsf{e})).$$

- $\mathsf{UE.enc}(\mathtt{k}_\mathsf{e}, \mathsf{m})$: Parse $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$. Generate $\mathsf{pk}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ and compute $\mathsf{ctx}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, \mathsf{m})$. Output
$$\mathsf{ct}_\mathsf{e} = (0, (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})).$$

- $\mathsf{UE.upd}(\Delta_{\mathsf{e}+1}, \mathsf{ct}_\mathsf{e})$: Parse the update token and the ciphertext as
$$\Delta_{\mathsf{e}+1} = (\mathsf{pk}_\Delta, \mathsf{ctx}_\Delta), \quad \mathsf{ct}_\mathsf{e} = (t, (\overline{\mathsf{pk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e}))$$
Sample a uniform random permutation $\pi : [2n] \to [2n]$. Also, let $\pi_{\mathsf{id}} : [2n] \to [2n]$ denote the *identity* permutation. Compute
$$(\overline{\mathsf{pk}}_\mathsf{e}, \overline{\mathsf{ctx}}_\mathsf{e}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e}, \pi, \pi_{\mathsf{id}}), \quad (\mathsf{pk}_{\mathsf{e}+1}, \mathsf{ctx}_{\mathsf{e}+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_\Delta, \mathsf{ctx}_\Delta, \pi_{\mathsf{id}}, \pi).$$
and output the updated ciphertext as:
$$\mathsf{ct}_{\mathsf{e}+1} = (t+1, (\overline{\mathsf{pk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}), \ldots, (\overline{\mathsf{pk}}_\mathsf{e}, \overline{\mathsf{ctx}}_\mathsf{e}), (\mathsf{pk}_{\mathsf{e}+1}, \mathsf{ctx}_{\mathsf{e}+1})).$$

- $\mathsf{UE.dec}(\mathtt{k}_\mathsf{e}, \mathsf{ct}_\mathsf{e})$: Parse $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ and the ciphertext as
$$\mathsf{ct}_\mathsf{e} = (t, (\overline{\mathsf{pk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})).$$
If $t = 0$, then output $\mathsf{m} \leftarrow \mathsf{KPHE.Dec}(\mathsf{sk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})$.
Otherwise, compute $\overline{\mathsf{sk}}_{\mathsf{e}-1} \leftarrow \mathsf{KPHE.Dec}(\mathsf{sk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})$. Then for each $j$ from $(\mathsf{e}-1)$ downto $(\mathsf{e}-t+1)$, compute
$$\overline{\mathsf{sk}}_{j-1} \leftarrow \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_j, \overline{\mathsf{ctx}}_j).$$
Finally, output the message $\mathsf{m} \leftarrow \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t})$.

**Correctness.** We first prove the correctness of the UE scheme. For any $\mathsf{m} \in \mathcal{M}$, any $\mathtt{k}_0 \leftarrow \mathsf{UE.setup}(1^\lambda)$, any sequence of $(\mathtt{k}_1, \Delta_1), \ldots, (\mathtt{k}_\mathsf{e}, \Delta_\mathsf{e})$ generated as $(\mathtt{k}_i, \Delta_i) \leftarrow \mathsf{UE.next}(\mathtt{k}_{i-1})$ for all $i \in [\mathsf{e}]$, let $\mathsf{ct}_0 \leftarrow \mathsf{UE.enc}(\mathtt{k}_0, \mathsf{m})$ and $\mathsf{ct}_i \leftarrow \mathsf{UE.upd}(\Delta_i, \mathsf{ct}_{i-1})$ for all $j \in [\mathsf{e}]$, then the final ciphertext is of the form $\mathsf{ct}_\mathsf{e} = (\mathsf{e}, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e}))$. All the secret keys are of the form $\mathtt{k}_0 = (\mathsf{pp}, \mathsf{sk}_0), \ldots, \mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$. Let $\pi_j$ be the random permutation sampled in $\mathsf{UE.upd}(\Delta_{j+1}, \mathsf{ct}_j)$ and let $\overline{\mathsf{sk}}_j = \pi_j(\mathsf{sk}_j)$ for all $j = 0, 1, \ldots, \mathsf{e}-1$. We can prove by induction that $\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_0, \overline{\mathsf{ctx}}_0) = \mathsf{m}, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_1, \overline{\mathsf{ctx}}_1) = \overline{\mathsf{sk}}_0, \ldots, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}) = \overline{\mathsf{sk}}_{\mathsf{e}-2}, \mathsf{KPHE.Dec}(\mathsf{sk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e}) = \overline{\mathsf{sk}}_{\mathsf{e}-1}$. Therefore, $\mathsf{UE.dec}(\mathtt{k}_\mathsf{e}, \mathsf{ct}_\mathsf{e})$ outputs $\mathsf{m}$. This argument is for any ciphertext starting from epoch 0. The same argument holds for any ciphertext starting from any epoch $\widehat{\mathsf{e}}$ where $0 \leq \widehat{\mathsf{e}} \leq \mathsf{e}$.

**Confidentiality.** Next we prove the IND-ENC security of our UE scheme. More formally, we state and prove the following theorem:

**Theorem 3.5** (IND-ENC Security). *Assuming that* KPHE *satisfies distributional security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2, the above* UE *construction is an IND-ENC secure unidirectional UE scheme.*

*Proof.* The proof proceeds via a hybrid argument.

$\mathsf{Hyb}_0$ The challenger plays the real game with the adversary.

$\mathsf{Hyb}_1$ Same as $\mathsf{Hyb}_0$ but for $\mathsf{UE.upd}(\Delta_\mathsf{e}, \mathsf{ct}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{upd}$ and $\mathsf{UE.upd}(\Delta_\mathsf{e}, \widetilde{\mathsf{ct}}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{next}$, do the following:

- Let $\mathtt{k}_{\mathsf{e}-1} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}-1})$ and $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$.
- Parse the ciphertext $\mathsf{ct}_{\mathsf{e}-1}$ or $\widetilde{\mathsf{ct}}_{\mathsf{e}-1}$ as

$$(t, (\overline{\mathsf{pk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-2}, \overline{\mathsf{ctx}}_{\mathsf{e}-2}), (\mathsf{pk}_{\mathsf{e}-1}, \mathsf{ctx}_{\mathsf{e}-1})),$$

  where $\mathsf{ctx}_{\mathsf{e}-1} = \mathsf{KPHE.Enc}(\mathsf{pk}_{\mathsf{e}-1}, x)$. Note that if $t = 0$, then $x = \mathsf{m}$ for some message, otherwise $x = \overline{\mathsf{sk}}_{\mathsf{e}-2}$ that is the KPHE secret key corresponding to $\overline{\mathsf{pk}}_{\mathsf{e}-2}$.
- Sample a uniform random permutation $\pi : [2n] \to [2n]$, let $\overline{\mathsf{sk}}_{\mathsf{e}-1} = \pi(\mathsf{sk}_{\mathsf{e}-1})$, and sample $\overline{\mathsf{pk}}_{\mathsf{e}-1} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_{\mathsf{e}-1})$. Compute $\overline{\mathsf{ctx}}_{\mathsf{e}-1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_{\mathsf{e}-1}, x)$.
- Sample $\mathsf{pk}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ and compute $\mathsf{ctx}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, \overline{\mathsf{sk}}_{\mathsf{e}-1})$.
- Let $\mathsf{ct}_\mathsf{e}$ or $\widetilde{\mathsf{ct}}_\mathsf{e}$ be

$$(t+1, (\overline{\mathsf{pk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})).$$

We prove in Lemma 3.6 that this hybrid is computationally indistinguishable from $\mathsf{Hyb}_0$ to any PPT adversary by the blinding property of KPHE.

$\mathsf{Hyb}_2$ Same as $\mathsf{Hyb}_1$ but for $\mathsf{UE.upd}(\Delta_\mathsf{e}, \mathsf{ct}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{upd}$ and $\mathsf{UE.upd}(\Delta_\mathsf{e}, \widetilde{\mathsf{ct}}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{next}$, instead of letting $\overline{\mathsf{sk}}_{\mathsf{e}-1} = \pi(\mathsf{sk}_{\mathsf{e}-1})$, sample $\overline{\mathsf{sk}}_{\mathsf{e}-1}$ from the distribution $\mathcal{U}_n$. This hybrid is statistically identical to $\mathsf{Hyb}_1$.

$\mathsf{Hyb}_3$ Let $\widetilde{\mathsf{e}}$ be the challenge epoch, and let $\overline{\mathsf{e}}$ be the last epoch where the adversary corrupts continuous update tokens from $\widetilde{\mathsf{e}}$, namely the adversary corrupts $\Delta_{\widetilde{\mathsf{e}}+1}, \Delta_{\widetilde{\mathsf{e}}+2}, \ldots, \Delta_{\overline{\mathsf{e}}}$ but not $\Delta_{\overline{\mathsf{e}}+1}$. This hybrid is the same as $\mathsf{Hyb}_2$ except that the challenger guesses $\widetilde{\mathsf{e}}^\star$ and $\overline{\mathsf{e}}^\star$ at the beginning of the game and aborts the game if guessing incorrectly. Let $E$ be the upper bound on the number of epochs during the game. If the challenger does not abort, then this hybrid is identical to $\mathsf{Hyb}_2$, which happens with probability at least $\frac{1}{E^2}$. In the remaining hybrids, we assume for simplicity that the challenger guesses $\widetilde{\mathsf{e}}$ and $\overline{\mathsf{e}}$ correctly.

$\mathsf{Hyb}_4$ Same as $\mathsf{Hyb}_3$ except that for each $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$, generate a single public key $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$. Then whenever $\mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, x)$ is computed for a freshly generated $\mathsf{pk}_\mathsf{e}$ and some $x$, compute it as $\mathsf{KPHE.Eval}(\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, x), \pi_{\mathsf{id}}, \pi_{\mathsf{id}})$. That is, instead of generating a fresh $\mathsf{pk}_\mathsf{e}$ from $\mathsf{sk}_\mathsf{e}$ every time, use the same $\widehat{\mathsf{pk}}_\mathsf{e}$ to encrypt $x$ and use then $\mathsf{KPHE.Eval}$ to re-randomize it.

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_3$ by the blinding property of $\mathsf{KPHE}$. We omit the detailed reduction here, but it is similar to the reduction in the proof of Lemma 3.6.

$\mathsf{Hyb}_5$ Same as $\mathsf{Hyb}_4$ except that for all $\widetilde{\mathsf{e}}+1 \le \mathsf{e} \le \overline{\mathsf{e}}$, $\mathsf{UE.next}(\mathtt{k}_{\mathsf{e}-1})$ is computed as follows. Generate $\mathsf{sk}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n)$ and let $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ be the single public key for $\mathtt{k}_\mathsf{e}$ (that will be used for every $\mathsf{KPHE.Enc}$). Output

$$\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e}), \quad \Delta_\mathsf{e} = (\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, 0^{2n})).$$

We prove in Lemma 3.7 that this hybrid is computationally indistinguishable from $\mathsf{Hyb}_4$ to any PPT adversary based on the distributional semantic security of $\mathsf{KPHE}$.

$\mathsf{Hyb}_6$ Same as $\mathsf{Hyb}_5$ except that for each $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$, generate a single public key $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ and use $\mathsf{KPHE.Eval}(\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, \cdot), \pi_{\mathsf{id}}, \pi_{\mathsf{id}})$ for all the computation of $\mathsf{KPHE.Enc}(\mathtt{k}_\mathsf{e}, \cdot)$ (including the computation of $\Delta_\mathsf{e}$). The only exception is the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$, which is computed using $\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ without re-randomization, namely

$$\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}} = (0, (\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \overline{\mathsf{m}}_\mathrm{b}))).$$

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_5$ by the blinding property of $\mathsf{KPHE}$. We omit the detailed reduction here, but it is similar to the reduction in the proof of Lemma 3.6.

Finally, we argue that in the final hybrid $\mathsf{Hyb}_6$, any PPT adversary cannot distinguish an encryption of $\overline{\mathsf{m}}_0$ or $\overline{\mathsf{m}}_1$ in the challenge epoch $\widetilde{\mathsf{e}}$, which relies on the distributional semantic security of $\mathsf{KPHE}$, which will conclude our proof.

Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish an encryption of $\overline{\mathsf{m}}_0$ or $\overline{\mathsf{m}}_1$ in the challenge epoch. Then we construct a PPT adversary $\mathcal{B}$ that breaks the distributional semantic security of $\mathsf{KPHE}$. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{pk})$ from the challenger in the semantic security security game. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}_6$. $\mathcal{B}$ uses $\mathsf{pp}$ to generate UE keys and update tokens as in $\mathsf{Hyb}_6$ except that for epoch $\widetilde{\mathsf{e}}$, the UE key $\mathtt{k}_{\widetilde{\mathsf{e}}}$ is unknown. When $\mathcal{B}$ receives the challenge messages $(\overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$ from $\mathcal{A}$ in the UE game, it forwards the two messages to the KPHE challenger and gets back $\mathsf{ctx}$, and then responds to $\mathcal{A}$ with $\mathsf{ct} = (0, (\mathsf{pp}, \mathsf{ctx}))$. Note that $\mathcal{B}$ doesn't need to know $\mathtt{k}_{\widetilde{\mathsf{e}}}$ because it is never used. In particular, $\mathcal{B}$ can use $\mathsf{pk}$ to compute all the $\mathsf{UE.enc}(\mathtt{k}_{\widetilde{\mathsf{e}}}, \cdot)$. Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

If $\mathcal{A}$ can distinguish between encryptions of $\overline{\mathsf{m}}_0$ and $\overline{\mathsf{m}}_1$ with non-negligible probability, then $\mathcal{B}$ can break the distributional semantic security of KPHE with non-negligible probability, which leads to contradiction.

**Lemma 3.6.** $\mathsf{Hyb}_0 \overset{c}{\approx} \mathsf{Hyb}_1$ *in the proof of Theorem 3.5.*

**Lemma 3.7.** $\mathsf{Hyb}_4 \overset{c}{\approx} \mathsf{Hyb}_5$ *in the proof of Theorem 3.5.*

We defer the formal proofs of Lemmas 3.6 and 3.7 to Appendix A. These proofs complete the overall proof of Theorem 3.5.

□

**Remark 3.8.** *In our construction one can derive* $\mathtt{k}_{\mathsf{e}-1}$ *from* $\Delta_{\mathsf{e}}$ *and* $\mathtt{k}_{\mathsf{e}}$. *It is for this reason that our construction permits* backward-leak unidirectional key updates *proposed by Nishimaki [Nis21] where secret keys can be derived in the backward direction but not forward direction. However, as discussed earlier, this notion is essentially equivalent to no-directional key updates (the optimal case) and has no bearing on our security analysis.*

## 3.3 Post-Compromise Secure Unidirectional UE

In this section, we show that any KPHE scheme with $2n$-bit secret keys that satisfies distributional security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2 implies a post-compromise secure unidirectional UE scheme.

### 3.3.1 IND-UPD Secure Unidirectional UE

We first show how to construct an UE scheme that satisfies the IND-UPD security definition as proposed in [LT18]. Given a KHPE scheme of the form

$$\mathsf{KPHE} = (\mathsf{KPHE.Setup}, \mathsf{KPHE.KeyGen}, \mathsf{KPHE.Enc}, \mathsf{KPHE.Dec}, \mathsf{KPHE.TransPK}, \mathsf{KPHE.Eval}),$$

with $2n$-bit secret keys, we construct a unidirectional UE scheme

$$\mathsf{UE} = (\mathsf{UE.setup}, \mathsf{UE.next}, \mathsf{UE.enc}, \mathsf{UE.upd}, \mathsf{UE.dec}),$$

that only differs from the IND-ENC construction in $\mathsf{UE.upd}$:

- $\mathsf{UE.upd}(\Delta_{\mathsf{e}+1}, \mathsf{ct}_{\mathsf{e}})$: Parse the update token and the ciphertext as

  $$\Delta_{\mathsf{e}+1} = (\mathsf{pk}_\Delta, \mathsf{ctx}_\Delta), \quad \mathsf{ct}_{\mathsf{e}} = (t, (\overline{\mathsf{pk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}}))$$

  Sample $(t+1)$ uniform random permutations $\pi_{\mathsf{e}-t}, \ldots, \pi_{\mathsf{e}} : [2n] \to [2n]$. Also, let $\pi_{\mathsf{id}} : [2n] \to [2n]$ denote the identity permutation. For each $i \in \{\mathsf{e}-t+1, \ldots, \mathsf{e}-1\}$, compute

  $$(\widetilde{\mathsf{pk}}_i, \widetilde{\mathsf{ctx}}_i) \overset{\$}{\leftarrow} \mathsf{KPHE.Eval}(\overline{\mathsf{pk}}_i, \overline{\mathsf{ctx}}_i, \pi_i, \pi_{i-1}).$$

Additionally, compute

$$(\widetilde{\mathsf{pk}}_{\mathsf{e}-t}, \widetilde{\mathsf{ctx}}_{\mathsf{e}-t}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\overline{\mathsf{pk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}, \pi_{\mathsf{e}-t}, \pi_{\mathsf{id}}),$$
$$(\overline{\mathsf{pk}}_{\mathsf{e}}, \overline{\mathsf{ctx}}_{\mathsf{e}}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}}, \pi_{\mathsf{e}}, \pi_{\mathsf{e}-1}),$$
$$(\mathsf{pk}_{\mathsf{e}+1}, \mathsf{ctx}_{\mathsf{e}+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_{\Delta}, \mathsf{ctx}_{\Delta}, \pi_{\mathsf{id}}, \pi_{\mathsf{e}}).$$

Output the updated ciphertext as:

$$\mathsf{ct}_{\mathsf{e}+1} = (t+1, (\widetilde{\mathsf{pk}}_{\mathsf{e}-t}, \widetilde{\mathsf{ctx}}_{\mathsf{e}-t}), \ldots, (\widetilde{\mathsf{pk}}_{\mathsf{e}-1}, \widetilde{\mathsf{ctx}}_{\mathsf{e}-1}), (\overline{\mathsf{pk}}_{\mathsf{e}}, \overline{\mathsf{ctx}}_{\mathsf{e}}), (\mathsf{pk}_{\mathsf{e}+1}, \mathsf{ctx}_{\mathsf{e}+1})).$$

**Correctness.** We first prove the correctness of the UE scheme. For any $\mathsf{m} \in \mathcal{M}$, any $\mathtt{k}_0 \leftarrow \mathsf{UE.setup}(1^\lambda)$, any sequence of $(\mathtt{k}_1, \Delta_1), \ldots, (\mathtt{k}_{\mathsf{e}}, \Delta_{\mathsf{e}})$ generated as $(\mathtt{k}_i, \Delta_i) \leftarrow \mathsf{UE.next}(\mathtt{k}_{i-1})$ for all $i \in [\mathsf{e}]$, let $\mathsf{ct}_0 \leftarrow \mathsf{UE.enc}(\mathtt{k}_0, \mathsf{m})$ and $\mathsf{ct}_i \leftarrow \mathsf{UE.upd}(\Delta_i, \mathsf{ct}_{i-1})$ for all $j \in [\mathsf{e}]$, then the final ciphertext is of the form $\mathsf{ct}_{\mathsf{e}} = (\mathsf{e}, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}}))$. All the UE secret keys are of the form $\mathtt{k}_0 = (\mathsf{pp}, \mathsf{sk}_0), \ldots, \mathtt{k}_{\mathsf{e}} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}})$. We can prove by induction that there exist permutations $\pi_0, \pi_1 \ldots, \pi_{\mathsf{e}-1} : [2n] \to [2n]$ such that $\overline{\mathsf{sk}}_i = \pi_i(\mathsf{sk}_i)$ for all $i = 0, 1, \ldots, \mathsf{e}-1$, and that $\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_0, \overline{\mathsf{ctx}}_0) = \mathsf{m}, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_1, \overline{\mathsf{ctx}}_1) = \overline{\mathsf{sk}}_0, \ldots, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}) = \overline{\mathsf{sk}}_{\mathsf{e}-2}, \mathsf{KPHE.Dec}(\mathsf{sk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}}) = \overline{\mathsf{sk}}_{\mathsf{e}-1}$. Therefore, $\mathsf{UE.dec}(\mathtt{k}_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}})$ outputs $\mathsf{m}$. This argument is for any ciphertext starting from epoch 0. The same argument holds for any ciphertext starting from any epoch $\widehat{\mathsf{e}}$ where $0 \leq \widehat{\mathsf{e}} \leq \mathsf{e}$.

**Confidentiality.** Next we prove the IND-UPD security the UE scheme. More formally, we state and prove the following theorem (the proof is provided in Appendix B):

**Theorem 3.9** (IND-UPD Security)**.** *Assuming that* KPHE *satisfies distributional security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2, the above* UE *construction is an IND-UPD secure unidirectional UE scheme.*

### 3.3.2 IND-UE Secure Unidirectional UE

The basic IND-UPD construction allows ciphertexts from the same epoch $\mathsf{e}$ to have different sizes. In particular, a freshly created ciphertext in epoch $\mathsf{e}$ can be trivially distinguished from a ciphertext that was created as an update of a ciphertext from epoch $(\mathsf{e}-1)$. So it cannot satisfy the combined security definition of post-compromise security for UE due to Boyd et al. [BDGJ20].

We showcase here a simple extension of the basic construction wherein we ensure that the size for any ciphertext in epoch $\mathsf{e}$ is the same, irrespective of whether it was freshly created, or created as an update of a ciphertext from epoch $(\mathsf{e}-1)$. The overall construction remains exactly the same; the key alteration is in how we generate fresh ciphertexts. At a high level, a freshly created ciphertext in epoch $\mathsf{e}$ is made to look exactly like a ciphertext that has undergone $\mathsf{e}$ update operations. We do this by having $\mathsf{e}$ "dummy wrapper" layers over and above the core ciphertext generated by the basic construction.

We describe the new encryption algorithm below (we assume here that information about the epoch $\mathsf{e}$ is available as part of $\mathtt{k}_{\mathsf{e}}$):

- $\mathsf{UE.enc}(\mathtt{k}_{\mathsf{e}}, \mathsf{m})$: Parse $\mathtt{k}_{\mathsf{e}} = (\mathsf{e}, \mathsf{pp}, \mathsf{sk}_{\mathsf{e}})$ and generate $\mathsf{pk}_{\mathsf{e}} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_{\mathsf{e}})$.

For each $i \in \{0, \ldots, \mathsf{e}-1\}$, generate

$$\overline{\mathsf{sk}}_i \stackrel{\$}{\leftarrow} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n), \quad \overline{\mathsf{pk}}_i \stackrel{\$}{\leftarrow} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_i).$$

Compute $\overline{\mathsf{ctx}}_0 \stackrel{\$}{\leftarrow} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_0, \mathsf{m})$ and $\mathsf{ctx}_\mathsf{e} \stackrel{\$}{\leftarrow} \mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, \overline{\mathsf{sk}}_{\mathsf{e}-1})$.
For each $i \in [\mathsf{e}-1]$, compute

$$\overline{\mathsf{ctx}}_i \stackrel{\$}{\leftarrow} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_i, \overline{\mathsf{sk}}_{i-1}).$$

Output the ciphertext $\mathsf{ct}_\mathsf{e} = (\mathsf{e}, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e}))$.

The only difference between the above UE construction and the IND-UPD secure UE construction presented in Section 3.3.1 is that all the fresh and updated ciphertexts in epoch $\mathsf{e}$ have the same length. The correctness and confidentiality proofs of the new construction follow exactly the same way as the IND-UPD secure construction. In particular, for the security proof, we can use the same hybrid argument as in Theorem 3.9 and prove that the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ is computationally indistinguishable from a ciphertext freshly generated in $\mathsf{Hyb}_7$ of the form

$$(\widetilde{\mathsf{e}}, (\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0), \ldots, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1}), (\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \mathsf{ctx}_{\widetilde{\mathsf{e}}})),$$

where each $\mathsf{ctx}$ is a $\mathsf{KPHE}$ encryption of $0^{2n}$. We state the theorem below and omit the detailed proof.

**Theorem 3.10** (IND-UE Security). *Assuming that* $\mathsf{KPHE}$ *is a KPHE scheme with $2n$-bit secret keys that satisfies distributional security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2, the above* $\mathsf{UE}$ *construction is an IND-UE secure unidirectional UE scheme.*

# 4 Unidirectional PRE from Circular-Secure KPHE

In this section, we show how to construct unidirectional PRE from any KPHE scheme that satisfies distributional circular security. We present the simpler construction of IND-HRA unidirectional PRE in Section 4.2. Subsequently, in Appendix 4.3, we show how to augment it to achieve the stronger notion of strong post-compromise security (PCS) as introduced in a recent work by Davidson et al. [DDLM19].

## 4.1 Definition

**Definition 4.1** (Unidirectional Proxy Re-Encryption (PRE)). *A unidirectional PRE scheme is a tuple of PPT algorithms of the form*

$$\mathsf{PRE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{ReKeyGen}, \mathsf{ReEnc}, \mathsf{Dec}),$$

*described as follows:*

- $\mathsf{pp} \stackrel{\$}{\leftarrow} \mathsf{Setup}(1^\lambda)$*: On input the security parameter $\lambda$, the setup algorithm outputs some public parameters* $\mathsf{pp}$ *(these parameters are implicit to all other algorithms).*

- $(\mathsf{sk}, \mathsf{pk}) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{pp})$: *On input the public parameters* $\mathsf{pp}$, *the key-generation algorithm outputs a secret key-public key pair,* $(\mathsf{sk}, \mathsf{pk})$.

- $\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}, \mathsf{m})$: *On input a public key* $\mathsf{pk}$ *and a message* $\mathsf{m}$, *the encryption algorithm outputs a ciphertext* $\mathsf{ct}$.

- $\mathsf{rk}_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}((\mathsf{sk}_i, \mathsf{pk}_i), \mathsf{pk}_j)$: *The re-key generation algorithm returns a re-encryption key* $\mathsf{rk}_{i,j}$ *for translation of a ciphertext from a key-pair* $(\mathsf{sk}_i, \mathsf{pk}_i)$ *to a key-pair* $(\mathsf{sk}_j, \mathsf{pk}_j)$. *It takes as input* $(\mathsf{sk}_i, \mathsf{pk}_i)$ *and* $\mathsf{pk}_j$, *and outputs the re-encryption key* $\mathsf{rk}_{i,j}$.[2]

- $\mathsf{ct}_j \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{rk}_{i,j}, \mathsf{ct}_i)$: *On input a re-encryption key* $\mathsf{rk}_{i,j}$ *and a ciphertext* $\mathsf{ct}_i$, *the re-encryption algorithm outputs an updated ciphertext* $\mathsf{ct}_j$.[3]

- $\mathsf{m}/\perp \leftarrow \mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$: *On input a secret key* $\mathsf{sk}$ *and a ciphertext* $\mathsf{ct}$, *the decryption algorithm outputs either a plaintext message or an error symbol.*

**Definition 4.2** (Correctness)**.** *A PRE scheme* $\mathsf{PRE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{ReKeyGen}, \mathsf{ReEnc}, \mathsf{Dec})$ *is said to be correct if for any* $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$, *for any* $\ell \geq 0$, *for any* $(\ell+1)$ *key-pairs* $(\mathsf{pk}_0, \mathsf{sk}_0), \ldots, (\mathsf{pk}_\ell, \mathsf{sk}_\ell) \xleftarrow{\$} \mathsf{KeyGen}(\mathsf{pp})$, *and for any plaintext message* $\mathsf{m}$, *letting* $\mathsf{ct}_0 \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_0, \mathsf{m})$, *and letting for each* $j \in [\ell]$

$$\mathsf{rk}_j \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_{j-1}, \mathsf{pk}_{j-1}, \mathsf{pk}_j), \quad \mathsf{ct}_j \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{rk}_j, \mathsf{ct}_{j-1}),$$

*we have* $\mathsf{Dec}(\mathsf{sk}_\ell, \mathsf{ct}_\ell) = \mathsf{m}$ *(with all but negligible probability).*

**Confidentiality.** We recall the various security notions for unidirectional PRE, namely IND-CPA security, IND-HRA security (introduced by Cohen [Coh19]), and IND-PCS security (introduced by Davidson et al. [DDLM19]). Our definitions are game-based, where the game is played between a challenger and a PPT adversary $\mathcal{A}$. We assume that the adversary $\mathcal{A}$ in a PRE security game has access to (a subset of) the oracles defined in Figure 5. As with our UE definitions, we begin by introducing the following notations for our security definitions:

- $\mathcal{L}$: Set of non-challenge ciphertexts $(i, \mathsf{ct}_i)$ ($i$ being the key-index under which $\mathsf{ct}_i$ is generated) produced by calls to the $\mathcal{O}.\mathsf{enc}$ or $\mathcal{O}.\mathsf{HonReEnc}$ oracle. $\mathcal{O}.\mathsf{HonReEnc}$ only outputs a re-encryption of a non-challenge ciphertext provided that this ciphertext is currently available in $\mathcal{L}$.

- $\widetilde{\mathcal{L}}$: Set of re-encrypted versions of the challenge ciphertext $\widetilde{\mathsf{ct}}$. $\widetilde{\mathcal{L}}$ is initiated with the challenge ciphertext $(i, \widetilde{\mathsf{ct}})$ ($i$ being the key-index under which $\widetilde{\mathsf{ct}}$ is generated). Any call to the $\mathcal{O}.\mathsf{HonReEnc}$ oracle on a ciphertext in $\widetilde{\mathcal{L}}$ results in a re-encrypted challenge ciphertext that gets added to $\widetilde{\mathcal{L}}$.

- $\mathcal{K}_{\texttt{Corrupt}}$: Set of indices $i \in [n]$ for which the adversary has corrupted the secret key $\mathsf{sk}_i$ (using $\mathcal{O}.\mathsf{KeyGen}$).

[2] In a bidirectional PRE scheme, the re-key generation algorithm additionally takes as input the destination secret key $\mathsf{sk}_j$, i.e., it takes as input $(\mathsf{sk}_i, \mathsf{pk}_i)$ and $(\mathsf{sk}_j, \mathsf{pk}_j)$, and outputs the re-encryption key $\mathsf{rk}_{i,j}$.

[3] The re-encryption algorithm could be either deterministic or randomized; in this work, we assume throughout that the re-encryption algorithm is randomized.

$\underline{\mathsf{Setup}(1^\lambda)}$:

$\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$\mathsf{state} := \bot$
$\mathcal{L}, \widetilde{\mathcal{L}}, \mathcal{K}_{\texttt{Corrupt}}, \mathcal{K}_{\texttt{Honest}} := \emptyset$
**return** $\mathsf{pp}$

$\underline{\mathcal{O}.\mathsf{KeyGen}(n, \mathcal{K})}$:

$(\mathsf{sk}_1, \mathsf{pk}_1), \ldots, (\mathsf{sk}_n, \mathsf{pk}_n) \xleftarrow{\$} \mathsf{KeyGen}(1^\lambda)$
$\mathcal{K}_{\texttt{Corrupt}} := \mathcal{K}$
$\mathcal{K}_{\texttt{Honest}} := [n] \setminus \mathcal{K}$
$\mathsf{state} := \{\mathsf{sk}_i, \mathsf{pk}_i\}_{i \in [n]}$
**return** $\left(\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\texttt{Corrupt}}}, \{\mathsf{pk}_i\}_{i \in [n]}\right)$

$\underline{\mathcal{O}.\mathsf{enc}(i, \mathsf{m})}$

$\mathsf{ct} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, \mathsf{m})$
$\mathcal{L} := \mathcal{L} \cup (i, \mathsf{ct})$
**return** $\mathsf{ct}$

$\underline{\mathcal{O}.\mathsf{ReKeyGen}(i, j)}$

**if** $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Corrupt}}$ **then**
  **return** $\bot$
$\mathsf{rk}_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{pk}_j)$
**return** $\mathsf{rk}_{i,j}$

$\underline{\mathcal{O}.\mathsf{ReEnc}(i, j, \mathsf{ct})}$

**if** $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Corrupt}}$ **then**
  **return** $\bot$
$\mathsf{rk}_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{pk}_j)$
$\mathsf{ct}' \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{rk}_{i,j}, \mathsf{ct})$
**return** $\mathsf{ct}'$

$\underline{\mathcal{O}.\mathsf{HonReEnc}(i, j, \mathsf{ct})}$

**if** $(i, \mathsf{ct}) \notin \mathcal{L},$ or $(i, \mathsf{ct}) \notin \widetilde{\mathcal{L}}$ **then**
  **return** $\bot$
**if** $(i, \mathsf{ct}) \in \widetilde{\mathcal{L}}$ and $j \in \mathcal{K}_{\texttt{Corrupt}}$ **then**
  **return** $\bot$
$\mathsf{rk}_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{pk}_j)$
$\mathsf{ct}' \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{rk}_{i,j}, \mathsf{ct})$
**if** $(i, \mathsf{ct}) \in \widetilde{\mathcal{L}}$ **then**
  $\widetilde{\mathcal{L}} := \widetilde{\mathcal{L}} \cup \{(j, \mathsf{ct}')\}$
**else**
  $\mathcal{L} := \mathcal{L} \cup \{(j, \mathsf{ct}')\}$
**return** $\mathsf{ct}'$

$\underline{\mathcal{O}.\mathsf{chall\text{-}PRE}(i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)}$[a]

**if** $i \in \mathcal{K}_{\texttt{Corrupt}}$ or $|\overline{\mathsf{m}}_0| \neq |\overline{\mathsf{m}}_1|$ **then**
  **return** $\bot$
$\widetilde{\mathsf{ct}} \xleftarrow{\$} \mathsf{Enc}(\mathsf{pk}_i, \overline{\mathsf{m}}_\mathrm{b})$
$\widetilde{\mathcal{L}} := \widetilde{\mathcal{L}} \cup \{(i, \widetilde{\mathsf{ct}})\}$
**return** $\widetilde{\mathsf{ct}}$

$\underline{\mathcal{O}.\mathsf{chall\text{-}IND\text{-}PCS}(i, j, \overline{\mathsf{ct}}_0, \overline{\mathsf{ct}}_1)}$

**if** $(i, \overline{\mathsf{ct}}_0) \notin \mathcal{L}$ or $(i, \overline{\mathsf{ct}}_1) \notin \mathcal{L}$ **then**
  **return** $\bot$
**if** $|\overline{\mathsf{ct}}_0| \neq |\overline{\mathsf{ct}}_1|$ **then**
  **return** $\bot$
**if** $j \in \mathcal{K}_{\texttt{Corrupt}}$ **then**
  **return** $\bot$
$\mathsf{rk}_{i,j} \xleftarrow{\$} \mathsf{ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{pk}_j)$
$\widetilde{\mathsf{ct}} \xleftarrow{\$} \mathsf{ReEnc}(\mathsf{rk}_{i,j}, \overline{\mathsf{ct}}_\mathrm{b})$
$\widetilde{\mathcal{L}} := \widetilde{\mathcal{L}} \cup \{(j, \widetilde{\mathsf{ct}})\}$
**return** $\widetilde{\mathsf{ct}}$

[a] This is the challenge oracle that is used in both the IND-CPA and the IND-HRA experiments for PRE.

Figure 5: Oracles in security games for unidirectional PRE.

- $\mathcal{K}_{\texttt{Honest}}$: Set of indices $i \in [n]$ for which the adversary has not corrupted the secret key $\mathsf{sk}_i$.

We now formally define the various security notions of unidirectional PRE. Note that the CPA and HRA proceed almost identically, with the major difference being the restrictions posed by their respective re-encryption oracles, $\mathcal{O}.\mathsf{ReEnc}$ and $\mathcal{O}.\mathsf{HonReEnc}$.

**Definition 4.3** (IND-CPA/IND-HRA Security)**.** *A unidirectional PRE scheme*

$$\mathsf{PRE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{ReKeyGen}, \mathsf{ReEnc}, \mathsf{Dec})$$

*is said to be* IND-CPA*-secure (*IND-HRA*-secure resp.) if for any security parameter* $\lambda$ *and any* *(non-uniform) PPT adversary $\mathcal{A}$, it holds that*

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}CPA/IND\text{-}HRA}}_{\mathcal{A},\mathsf{PRE}}(1^\lambda)=1\right]-\frac{1}{2}\right|\leq \mathrm{negl}(\lambda),$$

*for some negligible function* $\mathrm{negl}(\cdot)$*, where the experiment* $\mathsf{Exp}^{\mathsf{IND\text{-}CPA}}_{\mathcal{A},\mathsf{PRE}}$ $(\mathsf{Exp}^{\mathsf{IND\text{-}HRA}}_{\mathcal{A},\mathsf{PRE}}$ *resp.) is defined as below.*

**Experiment** $\mathsf{Exp}^{\mathsf{IND\text{-}CPA/IND\text{-}HRA}}_{\mathcal{A},\mathsf{PRE}}(1^\lambda)$:

Run $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$(\mathsf{state}, n, \mathcal{K}) \leftarrow \mathcal{A}(\mathsf{pp})$
Run $\left(\{\mathsf{sk}_i\}_{i\in\mathcal{K}_{\mathtt{Corrupt}}}, \{\mathsf{pk}_i\}_{i\in[n]}\right) \xleftarrow{\$} \mathcal{O}.\mathsf{KeyGen}(n,\mathcal{K})$
$(\mathsf{state}, i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1) \leftarrow \mathcal{A}^{\mathcal{O}.\mathsf{enc},\mathcal{O}.\mathsf{ReKeyGen},\mathcal{O}.\mathsf{ReEnc}/\mathcal{O}.\mathsf{HonReEnc}}\left(\mathsf{state}, \{\mathsf{sk}_i\}_{i\in\mathcal{K}_{\mathtt{Corrupt}}}, \{\mathsf{pk}_i\}_{i\in[n]}\right)$
$\mathrm{b} \xleftarrow{\$} \{0,1\}$
$\widetilde{\mathsf{ct}} \leftarrow \mathcal{O}.\mathsf{chall\text{-}PRE}(i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$
Proceed only if $\widetilde{\mathsf{ct}} \neq \bot$
$\mathrm{b}' \leftarrow \mathcal{A}^{\mathcal{O}.\mathsf{enc},\mathcal{O}.\mathsf{ReKeyGen},\mathcal{O}.\mathsf{ReEnc}/\mathcal{O}.\mathsf{HonReEnc}}(\mathsf{state}, \widetilde{\mathsf{ct}})$
**return** 1 if $\mathrm{b} = \mathrm{b}'$

**Definition 4.4** (IND-PCS Security)**.** *A unidirectional PRE scheme*

$$\mathsf{PRE} = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{ReKeyGen}, \mathsf{ReEnc}, \mathsf{Dec})$$

*is said to be* IND-PCS*-secure if for any security parameter $\lambda$ and any (non-uniform) PPT adversary $\mathcal{A}$, it holds that*

$$\left|\Pr\left[\mathsf{Exp}^{\mathsf{IND\text{-}PCS}}_{\mathcal{A},\mathsf{PRE}}(1^\lambda)=1\right]-\frac{1}{2}\right|\leq \mathrm{negl}(\lambda),$$

*for some negligible function* $\mathrm{negl}(\cdot)$*, where the experiment* $\mathsf{Exp}^{\mathsf{IND\text{-}PCS}}_{\mathcal{A},\mathsf{PRE}}$ *is defined as below.*

**Experiment** $\mathsf{Exp}^{\mathsf{IND\text{-}PCS}}_{\mathcal{A},\mathsf{PRE}}(1^\lambda)$:

Run $\mathsf{pp} \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$(\mathsf{state}, n, \mathcal{K}) \leftarrow \mathcal{A}(\mathsf{pp})$
Run $\left(\{\mathsf{sk}_i\}_{i\in\mathcal{K}_{\mathtt{Corrupt}}}, \{\mathsf{pk}_i\}_{i\in[n]}\right) \xleftarrow{\$} \mathcal{O}.\mathsf{KeyGen}(n,\mathcal{K})$
$(\mathsf{state}, i, j, \overline{\mathsf{ct}}_0, \overline{\mathsf{ct}}_1) \leftarrow \mathcal{A}^{\mathcal{O}.\mathsf{enc},\mathcal{O}.\mathsf{ReKeyGen},\mathcal{O}.\mathsf{HonReEnc}}\left(\mathsf{state}, \{\mathsf{sk}_i\}_{i\in\mathcal{K}_{\mathtt{Corrupt}}}, \{\mathsf{pk}_i\}_{i\in[n]}\right)$
$\mathrm{b} \xleftarrow{\$} \{0,1\}$
$\widetilde{\mathsf{ct}} \leftarrow \mathcal{O}.\mathsf{chall\text{-}IND\text{-}PCS}(i, j, \overline{\mathsf{ct}}_0, \overline{\mathsf{ct}}_1)$
Proceed only if $\widetilde{\mathsf{ct}} \neq \bot$
$\mathrm{b}' \leftarrow \mathcal{A}^{\mathcal{O}.\mathsf{enc},\mathcal{O}.\mathsf{ReKeyGen},\mathcal{O}.\mathsf{HonReEnc}}(\mathsf{state}, \widetilde{\mathsf{ct}})$
**return** 1 if $\mathrm{b} = \mathrm{b}'$

## 4.2 HRA-Secure Unidirectional PRE

We show that any KPHE scheme with $2n$-bit secret keys and plaintext messages that satisfies: (a) distributional semantic and *circular* security with respect to the distribution $\mathcal{U}_n$, and (b) blinding, implies the existence of a multi-hop IND-HRA secure unidirectional PRE scheme.

**Construction.** Given a KHPE scheme of the form

$$\mathsf{KPHE} = (\mathsf{KPHE.Setup}, \mathsf{KPHE.SKGen}, \mathsf{KPHE.PKGen}, \mathsf{KPHE.Enc}, \mathsf{KPHE.Dec}, \mathsf{KPHE.Eval}),$$

with $2n$-bit secret keys, we construct a unidirectional PRE scheme

$$\mathsf{PRE} = (\mathsf{PRE.Setup}, \mathsf{PRE.KeyGen}, \mathsf{PRE.Enc}, \mathsf{PRE.ReKeyGen}, \mathsf{PRE.ReEnc}, \mathsf{PRE.Dec}),$$

with message space $\mathcal{M} = \{0,1\}^{2n}$ as follows:

- $\mathsf{PRE.Setup}(1^\lambda)$: Sample $\mathsf{pp} \xleftarrow{\$} \mathsf{KPHE.Setup}(1^\lambda)$ and output $\mathsf{pp}$.
- $\mathsf{PRE.KeyGen}(\mathsf{pp})$: Sample and output $(\mathsf{pk}, \mathsf{sk})$ where

$$\mathsf{sk} \xleftarrow{\$} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n), \quad \mathsf{pk} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}).$$

- $\mathsf{PRE.Enc}(\mathsf{pk}, \mathsf{m})$: Compute $\mathsf{ctx}_0 \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}, \mathsf{m})$ and output

$$\mathsf{ct} = (0, (\mathsf{pk}, \mathsf{ctx}_0)).$$

- $\mathsf{PRE.ReKeyGen}(\mathsf{sk}_i, \mathsf{pk}_i, \mathsf{pk}_j)$: Output $\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{ctx}_\Delta)$, where

$$\mathsf{ctx}_\Delta \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, \mathsf{sk}_i).$$

- $\mathsf{PRE.ReEnc}(\mathsf{rk}_{i,j}, \mathsf{ct})$: Parse the reencryption key and the ciphertext as

$$\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{ctx}_\Delta), \quad \mathsf{ct} = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)),$$

  for some $t \geq 0$. Sample a uniformly random permutation $\pi : [2n] \to [2n]$. Also, let $\pi_{\mathsf{id}} : [2n] \to [2n]$ denote the *identity* permutation. Compute

$$(\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t) \xleftarrow{\$} \mathsf{KPHE.Eval}(\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t, \pi, \pi_{\mathsf{id}}),$$

$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \mathsf{ctx}_\Delta, \pi_{\mathsf{id}}, \pi),$$

  and output the updated ciphertext as:

$$\mathsf{ct}' = (t+1, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

- $\mathsf{PRE.Dec}(\mathsf{sk}, \mathsf{ct})$: Parse the ciphertext as

$$\mathsf{ct} = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)),$$

  for some $t \geq 0$. Compute $\overline{\mathsf{sk}}_{t-1} = \mathsf{KPHE.Dec}(\mathsf{sk}, \widehat{\mathsf{ctx}}_t)$. Next, compute the following for each $\ell$ from $(t-1)$ to 1 in decreasing order:

$$\overline{\mathsf{sk}}_{\ell-1} = \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_\ell, \overline{\mathsf{ctx}}_\ell).$$

  Finally, output the message $\mathsf{m} = \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_0, \overline{\mathsf{ctx}}_0)$.

We prove correctness in the following lemma:

**Lemma 4.5** (Correctness). *If the underlying* $\mathsf{KPHE}$ *scheme is correct, then for any* $\mathsf{pp} \xleftarrow{\$} \mathsf{PRE.Setup}(1^\lambda)$*, for any sequence of* $(n+1)$ *key-pairs (where* $n \geq 0$*):*

$$(\mathsf{pk}_0, \mathsf{sk}_0), (\mathsf{pk}_1, \mathsf{sk}_1), \ldots, (\mathsf{pk}_n, \mathsf{sk}_n) \xleftarrow{\$} \mathsf{PRE.KeyGen}(\mathsf{pp}),$$

*and for any plaintext message* $\mathsf{m}$*, letting* $\mathsf{ct}_0 \xleftarrow{\$} \mathsf{PRE.Enc}(\mathsf{pk}_1, \mathsf{m})$*, and letting for each* $j \in [n]$

$$\mathsf{rk}_j \xleftarrow{\$} \mathsf{PRE.ReKeyGen}(\mathsf{sk}_{j-1}, \mathsf{pk}_{j-1}, \mathsf{pk}_j), \quad \mathsf{ct}_j \xleftarrow{\$} \mathsf{PRE.ReEnc}(\mathsf{rk}_j, \mathsf{ct}_{j-1}),$$

*we must have* $\mathsf{PRE.Dec}(\mathsf{sk}_n, \mathsf{ct}_n) = \mathsf{m}$ *(with all but negligible probability).*

*Proof.* From the description of the re-encryption algorithm, it follows that

$$\mathsf{ct}_n = (n, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), (\overline{\mathsf{pk}}_1, \overline{\mathsf{ctx}}_1), \ldots, (\overline{\mathsf{pk}}_{n-1}, \overline{\mathsf{ctx}}_{n-1}), (\widehat{\mathsf{pk}}_n, \widehat{\mathsf{ctx}}_n)),$$

where for each $j \in [0, n-1]$, letting $\pi_j : [2n] \to [2n]$ be a uniformly random permutation sampled during the execution of $\mathsf{PRE.ReEnc}(\mathsf{rk}_{j+1}, \mathsf{ct}_j)$, and letting $\overline{\mathsf{sk}}_j = \pi_j(\mathsf{sk}_j)$, we have the following (with all but negligible probability) whenever the $\mathsf{KPHE}$ scheme is correct:

$$\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_0, \overline{\mathsf{ctx}}_0) = \mathsf{m}, \quad \{\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_j, \overline{\mathsf{ctx}}_j) = \overline{\mathsf{sk}}_{j-1}\}_{j \in [n-1]},$$

$$\mathsf{KPHE.Dec}(\mathsf{sk}_n, \widehat{\mathsf{ctx}}_n) = \overline{\mathsf{sk}}_{n-1}.$$

It immediately follows that we have (with all but negligible probability)

$$\begin{aligned}
&\mathsf{PRE.Dec}(\mathsf{sk}_t, \mathsf{ct}_t) \\
&= \mathsf{KPHE.Dec}(\ldots(\mathsf{KPHE.Dec}(\boxed{\mathsf{KPHE.Dec}(\mathsf{sk}_n, \widehat{\mathsf{ctx}}_n)}, \overline{\mathsf{ctx}}_{n-1}), \ldots), \overline{\mathsf{ctx}}_0) \\
&= \mathsf{KPHE.Dec}(\ldots(\mathsf{KPHE.Dec}(\boxed{\overline{\mathsf{sk}}_{n-1}}, \overline{\mathsf{ctx}}_{n-1}), \ldots), \overline{\mathsf{ctx}}_0) \\
&\vdots \\
&= \mathsf{KPHE.Dec}(\boxed{\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_1, \overline{\mathsf{ctx}}_1)}, \overline{\mathsf{ctx}}_0) \\
&= \mathsf{KPHE.Dec}(\boxed{\overline{\mathsf{sk}}_0}, \overline{\mathsf{ctx}}_0) \\
&= \mathsf{m}.
\end{aligned}$$

□

**Theorem 4.6** (IND-HRA Security). *Assuming that* $\mathsf{KPHE}$ *satisfies blinding and distributional semantic+circular security with respect to the distribution* $\mathcal{U}_n$*,* $\mathsf{PRE}$ *is a multi-hop IND-HRA secure unidirectional PRE scheme.*

**IND-CPA Security (Warm-Up).** As a warm-up, we begin by outlining the proof of IND-CPA security for our construction, which is significantly simpler than the proof of IND-HRA security. We subsequently present the proof of IND-HRA security (in Appendix C), which is significantly more nuanced since it requires careful simulation of adversarial re-encryption queries on honestly generated ciphertexts.

We assume throughout that the adversary corrupts keys *statically* at the beginning of the IND-CPA game, while the ReKeyGen queries may be issued adaptively. Finally, the adversary may issue the challenge encryption query adaptively on any honest key of its choice at any point during the IND-CPA game.

The proof proceeds through a sequence of hybrids. We use $[N]$ to denote the set of all keys (honest + corrupt), $\mathcal{K}_{\texttt{Honest}} \subset [N]$ to denote the set of honest keys, and $\mathcal{K}_{\texttt{Corrupt}} \subset [N]$ to denote the set of corrupt keys. As mentioned earlier, these sets are fixed (adversarially) at the beginning of the game, before the adversary is allowed to issue re-key generation queries (this remains unchanged in each hybrid).

$\mathsf{Hyb}_0$**:** This hybrid is identical to the real IND-CPA security game between the challenger and the adversary.

$\mathsf{Hyb}_1$**:** This hybrid is identical to $\mathsf{Hyb}_0$ except for the manner in which the challenger answers the ReKeyGen and ReEnc queries issued by the adversary. In particular, at setup, the challenger creates an $(n \times n)$ table $T_{\mathsf{rk}}$ (initially empty) and populates it as follows:

- If $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Corrupt}}$, set $T_{\mathsf{rk}}[i,j] = \perp$.
- Else, set $T_{\mathsf{rk}}[i,j] = (\mathsf{pk}_j, \mathsf{KPHE.Enc}(\mathsf{pk}_j, \mathsf{sk}_i))$.

Given a query of the form $\mathsf{ReKeyGen}(i,j)$, the challenger responds with $T_{\mathsf{rk}}[i,j]$. Additionally, given a query of the form $\mathsf{ReEnc}(i,j,\mathsf{ct})$, the challenger proceeds as follows:

- If $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Corrupt}}$, respond with $\perp$.
- Else, respond with the updated ciphertext $\mathsf{ct}' \xleftarrow{\$} \mathsf{ReEnc}(T_{\mathsf{rk}}[i,j], \mathsf{ct})$.

It is easy to see that $\mathsf{Hyb}_1$ is identical to $\mathsf{Hyb}_0$.

$\mathsf{Hyb}_2$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_1$ except that the challenger pre-populates the table $T_{\mathsf{rk}}$ at setup as follows: if $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Honest}}$, set $T_{\mathsf{rk}}[i,j] = (\mathsf{pk}_j, \mathsf{KPHE.Enc}(\mathsf{pk}_j, 0^{2n}))$.

We argue that $\mathsf{Hyb}_2$ is indistinguishable from $\mathsf{Hyb}_1$ in a straightforward manner under the assumption that KPHE satisfies distributional circular security.

**Remark 4.7.** *Note that in* $\mathsf{Hyb}_2$*, the set of honest secret keys* $\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\texttt{Honest}}}$ *is no longer used by the challenger when answering* ReKeyGen *and* ReEnc *queries. In other words,* $\mathsf{Hyb}_2$ *allows the challenger to "forget" the honest secret keys.*

$\mathsf{Hyb}_3$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_2$ except for the manner in which the challenger answers the challenge encryption query issued by the adversary. In particular, given a query of the form $\mathcal{O}.\mathsf{chall}(i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$, the challenger proceeds as follows:

- If $i \in \mathcal{K}_{\texttt{Corrupt}}$, respond with $\mathsf{ct}^* = \bot$ (this is exactly as in $\mathsf{Hyb}_1$).
- If $i \in \mathcal{K}_{\texttt{Honest}}$, respond with $\mathsf{ct}^* = (0, \mathsf{KPHE.Enc}(\mathsf{pk}_i, 0^{2n}))$.

We argue that $\mathsf{Hyb}_3$ is indistinguishable from $\mathsf{Hyb}_2$ under the assumption that $\mathsf{KPHE}$ satisfies distributional semantic security.

We defer the detailed proof of IND-HRA security to Appendix C.

### 4.3 Strongly Post-Compromise Secure Unidirectional PRE

We augment the construction in Section 4 to achieve a strongly post-compromise secure unidirectional PRE scheme (as per the notion of strong post-compromise security introduced in a recent work by Davidson et al. [DDLM19]), without assuming any additional properties on the underlying KPHE scheme. In particular, we only make the following modification to the re-encryption algorithm $\mathsf{PRE.ReEnc}$ from the IND-HRA construction described above:

$\mathsf{PRE.ReEnc}(\mathsf{rk}_{i,j}, \mathsf{ct})$**:** Parse the update token and the ciphertext as

$$\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{ctx}_\Delta), \quad \mathsf{ct} = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)),$$

for some $t \geq 0$. Sample $(t+1)$ uniform permutations $\pi_0, \ldots, \pi_t : [2n] \to [2n]$. Also, let $\pi_{\mathsf{id}} : [2n] \to [2n]$ denote the *identity* permutation. For each $\ell \in [1, t]$, compute

$$(\widetilde{\mathsf{pk}}_\ell, \widetilde{\mathsf{ctx}}_\ell) \xleftarrow{\$} \mathsf{KPHE.Eval}(\overline{\mathsf{pk}}_\ell, \overline{\mathsf{ctx}}_\ell, \pi_\ell, \pi_{\ell-1}).$$

Additionally compute

$$(\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0) \xleftarrow{\$} \mathsf{KPHE.Eval}(\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0, \pi_0, \pi_{\mathsf{id}}),$$
$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \mathsf{ctx}_\Delta, \pi_{\mathsf{id}}, \pi_t),$$

and output the updated ciphertext as:

$$\mathsf{ct}' = (t+1, (\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0), \ldots, (\widetilde{\mathsf{pk}}_{t-1}, \widetilde{\mathsf{ctx}}_{t-1}), (\widetilde{\mathsf{pk}}_t, \widetilde{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

**Correctness.** The correctness of the IND-PCS secure PRE scheme follows from essentially the same arguments as its IND-HRA counterpart, and is hence not detailed.

**Theorem 4.8** (IND-PCS Security)**.** *Assuming that* $\mathsf{KPHE}$ *is a KPHE scheme with* $2n$*-bit secret keys that satisfies blinding, distributional circular security and distributional semantic security with respect to the distribution* $\mathcal{U}_n$*,* $\mathsf{PRE}$ *is a multi-hop strongly post-compromise secure unidirectional PRE scheme.*

We defer the detailed proof to Appendix D.

# 5 KPHE from the DDH Assumption

In [BHHO08], Boneh *et al.* showed that assuming that the decisional Diffie-Hellman (DDH) assumption holds (over any prime-order group), there exists a KPHE scheme with $2n$-bit secret keys that satisfies circular security with respect to the distribution $\mathcal{U}_n$, as well as public key and ciphertext blinding as described in Section 2. In addition, Naor and Segev [NS12] showed that the scheme in [BHHO08] satisfies distributional semantic security, while Gentry *et al.* [GHV10] pointed out that this scheme additionally satisfies distributional circular security.

For the sake of completeness, we recall the DDH assumption, as well as the main results from [BHHO08, NS12, GHV10].

**The DDH assumption.** Let $\mathbb{G}$ be a cyclic group of prime order $p$, and let $g$ be any uniformly sampled generator for $\mathbb{G}$. The decisional Diffie-Hellman (DDH) assumption is that for all PPT algorithms $\mathcal{A}$, we have

$$\left|\Pr\left[\mathcal{A}\left(g, g^{\alpha}, g^{\beta}, g^{\alpha\cdot\beta}\right)=1\right]-\Pr\left[\mathcal{A}\left(g, g^{\alpha}, g^{\beta}, g^{\gamma}\right)=1\right]\right| \leq \mathsf{negl}(\lambda),$$

where $\alpha, \beta, \gamma \xleftarrow{R} \mathbb{Z}_p^*$.

**KPHE from DDH (Imported from [BHHO08]).** We now recall the construction of DDH-based KPHE from [BHHO08], which is the "expanded version" of the original circular secure PKE scheme in [BHHO08]. We note that the original scheme in [BHHO08] supports affine homomorphic transformations for both the key and the plaintext. In the description below, we present a simplified version of the scheme that allows homomorphically permuting keys and plaintexts (which are $2n$-bit strings), as required by our applications.

We first present some notations before describing the construction of KPHE from DDH. Let $\mathbb{G}$ be a cyclic group of prime order $p$. For any $m \in \mathbb{N}$, we use $\mathsf{GL}_m(\mathbb{Z}_p)$ to denote the set of all full-ranked $m \times m$ matrices with entries in $\mathbb{Z}_p$. For any $m, \ell, \ell' \in \mathbb{N}$ with $m \geq \ell \geq \ell'$, we use $\mathsf{Rk}_{\ell'}\left(\mathbb{G}^{m\times\ell}\right)$ denote the set of all matrices in $\mathbb{G}^{m\times\ell}$ of rank $\ell'$. For any $m \in \mathbb{N}$, any $\mathbf{r} = \begin{bmatrix} r_1 & \ldots & r_m \end{bmatrix}^{\mathbf{t}} \in \mathbb{Z}_p^m$, and any $\mathbf{h} = \begin{bmatrix} h_1 & \ldots & h_m \end{bmatrix}^{\mathbf{t}} \in \mathbb{G}^m$, we define

$$\mathbf{r}^{\mathbf{t}}\mathbf{h} := \prod_{i\in[m]} (h_i)^{r_i}.$$

Similarly, for any $m, n, \ell \in \mathbb{N}$, any $\mathbf{R} \in \mathbb{Z}_p^{n\times m}$, and any $\mathbf{H} \in \mathbb{G}^{m\times\ell}$ such that

$$\mathbf{R} = \begin{bmatrix} (\mathbf{r}_1)^{\mathbf{t}} & \ldots & (\mathbf{r}_n)^{\mathbf{t}} \end{bmatrix}^{\mathbf{t}}, \quad \mathbf{H} = \begin{bmatrix} \mathbf{h}_1 & \ldots & \mathbf{h}_n \end{bmatrix},$$

where $\mathbf{r}_i \in \mathbb{Z}_p^m$ and $\mathbf{h}_j \in \mathbb{G}^m$ represent the $i$-th row and $j$-th column of $\mathbf{R}$ and $\mathbf{H}$, respectively, we define the matrix of group elements $\mathbf{RH} \in \mathbb{G}^{n\times\ell}$ as

$$\mathbf{RH} := \begin{bmatrix} (\mathbf{r}_i)^{\mathbf{t}}\mathbf{h}_j \end{bmatrix}_{i\in[n], j\in[\ell]}.$$

Finally, for any $m \in \mathbb{N}$, given a vector $\mathbf{s} = \begin{bmatrix} s_1 & \ldots & s_m \end{bmatrix}^{\mathbf{t}} \in \{0,1\}^m$ and a permutation $\pi : \{0,1\}^m \to \{0,1\}^m$ we denote by the shorthand $\pi(\mathbf{s})$ the vector

$$\mathbf{s}' = \begin{bmatrix} s_{\pi(1)} & \ldots & s_{\pi(m)} \end{bmatrix}^{\mathbf{t}} \in \{0,1\}^m.$$

We now describe the construction of KPHE from DDH.

- $\mathsf{Setup}(1^\lambda)$: The setup algorithm outputs a public parameter $\mathsf{pp} = (\mathbb{G}, p, g)$, where $\mathbb{G}$ is a cyclic group of prime order $p$ ($p$ being a $\lambda$-bit prime), and $g \leftarrow G$ is a uniformly random generator for the group $\mathbb{G}$. In the description of the algorithms below, we assume that $\mathsf{pp}$ is an implicit input to each algorithm, and we avoid specifying it explicitly.

- $\mathsf{SKGen}(\mathsf{pp}, \mathcal{D} = \mathcal{U}_n)$: The secret key generation algorithm outputs a secret key $\mathsf{sk} = \mathbf{s} \in \{0,1\}^{2n}$ where $\mathbf{s} \xleftarrow{\$} \mathcal{U}_n$.

- $\mathsf{PKGen}(\mathsf{pp}, \mathsf{sk} = \mathbf{s})$: Let $n = \lceil 3 \log q \rceil$. The public key generation algorithm samples a uniformly random matrix of group elements $\mathbf{L} \leftarrow \mathsf{Rk}_{2n}\left(\mathbb{G}^{(2n+1)\times 2n}\right)$, computes the vector of group elements $\mathbf{h} = \mathbf{L}\mathbf{s}$, and outputs the public key $\mathsf{pk} = \mathbf{S} \in \mathbb{G}^{(2n+1)\times(2n+1)}$, where

$$\mathbf{S} = \begin{bmatrix} \mathbf{L} & | & -\mathbf{h} \end{bmatrix}.$$

- $\mathsf{Enc}(\mathsf{pk} = \mathbf{S}, \mathsf{m} \in \{0,1\}^{2n})$: Parse the message vector as $\mathsf{m} = (\mathsf{m}_1, \ldots, \mathsf{m}_{2n})$ where $\mathsf{m}_i \in \{0,1\}$ for each $i \in [n]$. Let

$$\mathbf{m} = \begin{bmatrix} g^{\mathsf{m}_1} & \ldots & g^{\mathsf{m}_{2n}} & g \end{bmatrix}^{\mathbf{t}} \in \mathbb{G}^{(2n+1)}, \quad \mathbf{M} = \begin{bmatrix} \mathbf{0}^{(2n+1)\times 2n} & | & \mathbf{m} \end{bmatrix} \in \mathbb{G}^{(2n+1)\times(2n+1)}.$$

The encryption algorithm samples a uniformly random matrix of field elements $\mathbf{R} \leftarrow \mathbb{Z}_p^{(2n+1)\times(2n+1)}$, and outputs the ciphertext $\mathsf{ct} = \mathbf{W} \in \mathbb{G}^{(2n+1)\times(2n+1)}$, where

$$\mathbf{W} = \mathbf{R}\mathbf{S} + \mathbf{M}.$$

- $\mathsf{Dec}(\mathsf{sk} = \mathbf{s}, \mathsf{ct} = \mathbf{W})$: Let $\mathbf{s}' = \begin{bmatrix} \mathbf{s}^{\mathbf{t}} & | & 1 \end{bmatrix}^{\mathbf{t}} \in \mathbb{G}^{(2n+1)}$. Compute

$$\mathbf{m}' = \begin{bmatrix} g_1 & \ldots & g_{2n} & g_{2n+1} \end{bmatrix}^{\mathbf{t}} = \mathbf{W}\mathbf{s}' \in \mathbb{G}^{(2n+1)},$$

and output $\mathsf{m}' = (\mathsf{m}'_1, \ldots, \mathsf{m}'_{2n})$, where for each $i \in [2n]$, we have $\mathsf{m}'_i = 1$ if $g_i = g$, and $\mathsf{m}'_i = 0$ otherwise.

- $\mathsf{Eval}\left(\mathsf{pk} = \mathbf{S}, \mathsf{ct} = \mathbf{W}, \left(\pi, \pi' : \{0,1\}^{2n} \rightarrow \{0,1\}^{2n}\right)\right)$: The evaluation algorithm proceeds via the following steps:

  - **Step-1: Message Rotation under Permutation $\pi'$.** Parse the ciphertext matrix $\mathbf{W} \in \mathbb{G}^{(2n+1)\times(2n+1)}$ as $\mathbf{W} = \left[(\mathbf{w}_1)^{\mathbf{t}} \quad \ldots \quad (\mathbf{w}_{2n})^{\mathbf{t}} \quad (\mathbf{w}_{2n+1})^{\mathbf{t}}\right]^{\mathbf{t}}$, where $\mathbf{w}_i$ is the $i$-th row of $\mathbf{W}$. Output the *row-permuted* ciphertext matrix $\mathbf{W}' \in \mathbb{G}^{(2n+1)\times(2n+1)}$, where

$$\mathbf{W}' = \left[\left(\mathbf{w}_{\pi'(1)}\right)^{\mathbf{t}} \quad \ldots \quad \left(\mathbf{w}_{\pi'(2n)}\right)^{\mathbf{t}} \quad (\mathbf{w}_{2n+1})^{\mathbf{t}}\right]^{\mathbf{t}}.$$

  Note that the permutation affects only the first $2n$ rows of the ciphertext (since these rows "contain" the $2n$ bits of the message), while the last row remains unchanged.

– **Step-2: Key Rotation (Public Key) under Permutation $\pi$.** Let $\mathbf{F} \in \mathbb{Z}_p^{2n\times 2n}$ be the invertible matrix associated with the permutation $\pi'$, i.e., we have $\mathbf{Fs} = \pi(\mathbf{s})$, and let

$$\mathbf{F}' = \begin{bmatrix} \mathbf{F} & \mathbf{0}^{2n\times 1} \\ \mathbf{0}^{1\times 2n} & 1 \end{bmatrix} \in \mathbb{Z}_p^{(2n+1)\times(2n+1)}.$$

Output the rotated public key $\mathsf{pk}' = \mathbf{S}\left(\mathbf{F}'\right)^{-1}$ as the public key corresponding to the secret key $\mathsf{sk}' = \pi(\mathbf{s})$.

– **Step-3: Key Rotation (Ciphertext) under Permutation $\pi$.** Given the message-rotated ciphertext matrix $\mathbf{W}' \in \mathbb{G}^{(2n+1)\times(2n+1)}$, output the message and key-rotated ciphertext $\mathsf{ct}'' = \mathbf{W}'\left(\mathbf{F}'\right)^{-1}$, where $F'$ is as defined above.

– **Step-4: Public Key Blinding:** Given the rotated public key $\mathsf{pk}' = \mathbf{S}' \in \mathbb{G}^{(2n+1)\times(2n+1)}$, sample $\mathbf{U} \leftarrow \mathsf{GL}_{(2n+1)}(\mathbb{Z}_p)$ and output the blinded public key

$$\widetilde{\mathsf{pk}} = \mathbf{US}'.$$

– **Step-5: Ciphertext Blinding:** Given the message and key-rotated ciphertext $\mathsf{ct}'' = \mathbf{W}'' \in \mathbb{G}^{(2n+1)\times(2n+1)}$, sample $\mathbf{V} \leftarrow \mathsf{GL}_{(2n+1)}(\mathbb{Z}_p)$ and output the blinded ciphertext

$$\widetilde{\mathsf{ct}} = \mathbf{VS}' + \mathbf{W}''.$$

**Correctness of Decryption.** Let $\mathbf{W}$ be a ciphertext encrypting a bit-vector $\mathsf{m} \in \{0,1\}^{2n}$ under a public key-secret key pair of the form $(\mathbf{S}, \mathbf{s})$. Also, let $\mathbf{s}' = \begin{bmatrix}\mathbf{s}^{\mathbf{t}} & | & 1\end{bmatrix}^{\mathbf{t}} \in \mathbb{G}^{(2n+1)}$. Note that we have

$$\mathbf{Ss}' = \mathbf{Ls} - \mathbf{h} = \mathbf{0}^{(2n+1)}, \quad \mathbf{Ms}' = \left(\mathbf{0}^{(2n+1)\times 2n}\right)\mathbf{s} + \mathbf{m} = \mathbf{m}.$$

Hence, we have

$$\mathbf{m}' = \mathbf{Ws}' = \mathbf{RSs}' + \mathbf{Ms}' = \mathbf{m},$$

and hence $\mathsf{m}' = \mathsf{m}$, as desired.

**Correctness of Evaluation.** Let $\mathbf{W}$ be a ciphertext encrypting a bit-vector $\mathsf{m} \in \{0,1\}^{2n}$ under a public key-secret key pair of the form $(\mathbf{S}, \mathbf{s})$. Now, observe the following.

*Correctness of Message Rotation.* Let $\mathbf{W} = \mathbf{RS} + \mathbf{M}$. Then $\mathbf{W}' = \mathbf{R}'\mathbf{S} + \mathbf{M}'$, where $\mathbf{R}'$ and $\mathbf{M}'$ are row-rotated versions of $\mathbf{R}$ and $\mathbf{M}$ under the permutation $\pi'$. Hence, we have $\mathbf{M} = \begin{bmatrix}\mathbf{0}^{(2n+1)\times 2n} & | & \mathbf{m}'\end{bmatrix}$, where $\mathbf{m}' = \pi'(\mathbf{m})$, as desired.

*Correctness of Key Rotation (Public Key).* Let $\mathbf{S} = \begin{bmatrix}\mathbf{L} & | & -\mathbf{Ls}\end{bmatrix}$. Then, we have

$$\mathbf{S}' = \mathbf{S}\left(\mathbf{F}'\right)^{-1} = \begin{bmatrix}\mathbf{L}\left(\mathbf{F}\right)^{-1} & | & -\mathbf{Ls}\end{bmatrix} = \begin{bmatrix}\mathbf{L}' & | & -\mathbf{L}'\mathbf{Fs}\end{bmatrix} = \begin{bmatrix}\mathbf{L}' & | & -\mathbf{L}'\pi(\mathbf{s})\end{bmatrix},$$

as desired (recall that $\mathbf{F}$ is invertible since it represents a permutation).

*Correctness of Key Rotation (Ciphertext).* Let $\mathbf{W}' = \mathbf{R}'\mathbf{S} + \mathbf{M}'$. Then we have

$$\mathbf{W}'' = \mathbf{R}'\mathbf{S}\left(\mathbf{F}'\right)^{-1} + \mathbf{M}'\left(\mathbf{F}'\right)^{-1} = \mathbf{R}'\mathbf{S}' + \left[\mathbf{0}^{(2n+1)\times 2n}\left(\mathbf{F}\right)^{-1} \quad | \quad \mathbf{m}'\right] = \mathbf{R}'\mathbf{S}' + \mathbf{M}',$$

as desired.

*Correctness of Public Key Blinding.* We have

$$\widetilde{\mathsf{pk}} = \mathbf{U}\mathbf{S}' = \left[\mathbf{U}\mathbf{L} \quad | \quad -\mathbf{U}\mathbf{L}\pi(\mathbf{s})\right] = \left[\widetilde{\mathbf{L}} \quad | \quad -\widetilde{\mathbf{L}}\pi(\mathbf{s})\right],$$

as desired.

*Correctness of Ciphertext Blinding.* We have

$$\widetilde{\mathsf{ct}} = \mathbf{V}\mathbf{S}' + \mathbf{W}'' = (\mathbf{V} + \mathbf{R}')\mathbf{S}' + \mathbf{M}' = \widetilde{\mathbf{R}}\mathbf{S}' + \mathbf{M}',$$

as desired. This completes the proof of evaluation correctness.

**Security Properties.** Finally, we state the following theorem. The proof of this theorem follows from prior works [BHHO08, GHV10, NS12], and is hence not explicitly detailed here.

**Theorem 5.1** (Imported from [BHHO08, GHV10, NS12]). *Assuming that the DDH assumption holds over the group $\mathbb{G}$, the above a KPHE scheme with $2n$-bit secret keys that satisfies: (i) distributional semantic security with respect to the distribution $\mathcal{U}_n$, (ii) distributional circular security with respect to the distribution $\mathcal{U}_n$, and (iii) blinding.*

*Proof Overview.* At a high level, the proofs of semantic and circular security follow from the matrix DDH family of assumptions (albeit for high-entropy secrets), which is in turn implied by the DDH assumption, while the proofs of key and ciphertext blinding follow from statistical arguments. We refer to Section 3.2 of [BHHO08] for the proofs of semantic security, circular security and key/ciphertext blinding assuming that the secret keys are uniformly random $2n$-bit strings (the authors of [BHHO08], in fact, prove key-dependent message (KDM) security, which implies both semantic and circular security). We also refer to [GHV10, NS12] for the adaptation of the proof to *high-entropy* secret keys, and more concretely, secret keys samples from the distribution $\mathcal{U}_n$. The adaptation of the proof of [BHHO08] follows, fundamentally, from an application of the *leftover hash lemma* [HILL99, Sho06] for high-entropy sources as opposed to uniformly random sources.

# References


[ACDT20] Joël Alwen, Sandro Coretti, Yevgeniy Dodis, and Yiannis Tselekounis. Security analysis and improvements for the IETF MLS standard for group messaging. In *CRYPTO 2020*, volume 12170, pages 248–277. Springer, 2020.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO 2009*, volume 5677, pages 595–618. Springer, 2009.

[AFGH06] Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved proxy re-encryption schemes with applications to secure distributed storage. *ACM Trans. Inf. Syst. Secur.*, 9(1):1–30, 2006.

[AMP19] Navid Alamati, Hart Montgomery, and Sikhar Patranabis. Symmetric primitives with structured secrets. In *CRYPTO 2019*, volume 11692, pages 650–679. Springer, 2019.

[BBB+12] Elaine Barker, William Barker, William Burr, William Polk, Miles Smid, Patrick D. Gallagher, and Under Secretary For. NIST Special Publication 800-57 Recommendation for Key Management – Part 1: General, 2012.

[BBS98] Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible protocols and atomic proxy cryptography. In *EUROCRYPT 1998*, volume 1403, pages 127–144. Springer, 1998.

[BDGJ20] Colin Boyd, Gareth T. Davies, Kristian Gjøsteen, and Yao Jiang. Fast and secure updatable encryption. In *CRYPTO 2020*, volume 12170, pages 464–493. Springer, 2020.

[BEKS20] Dan Boneh, Saba Eskandarian, Sam Kim, and Maurice Shih. Improving speed and security in updatable encryption schemes. In *ASIACRYPT 2020*, volume 12493, pages 559–589. Springer, 2020.

[BF03] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003.

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6:1–6:48, 2012.

[BHHO08] Dan Boneh, Shai Halevi, Michael Hamburg, and Rafail Ostrovsky. Circular-secure encryption from decision diffie-hellman. In *CRYPTO 2008*, volume 5157, pages 108–125. Springer, 2008.

[BLMR13] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic prfs and their applications. In *CRYPTO 2013*, volume 8042, pages 410–428. Springer, 2013.

[BRS02] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In *SAC 2002*, volume 2595, pages 62–75. Springer, 2002.

[CCL+14] Nishanth Chandran, Melissa Chase, Feng-Hao Liu, Ryo Nishimaki, and Keita Xagawa. Re-encryption, functional re-encryption, and multi-hop re-encryption: A framework for achieving obfuscation-based security and instantiations from lattices. In *PKC 2014*, volume 8383, pages 95–112. Springer, 2014.

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In *EUROCRYPT 2001*, volume 2045, pages 93–118. Springer, 2001.

[Coh19] Aloni Cohen. What about bob? the inadequacy of CPA security for proxy reencryption. In *PKC 2019*, volume 11443, pages 287–316. Springer, 2019.

[DDLM19] Alex Davidson, Amit Deo, Ela Lee, and Keith Martin. Strong post-compromise secure proxy re-encryption. In *ACISP 2019*, volume 11547, pages 58–77. Springer, 2019.

[DKW21] Yevgeniy Dodis, Harish Karthikeyan, and Daniel Wichs. Updatable public key encryption in the standard model. In *TCC 2021*, volume 13044, pages 254–285. Springer, 2021.

[EPRS17] Adam Everspaugh, Kenneth G. Paterson, Thomas Ristenpart, and Samuel Scott. Key rotation for authenticated encryption. In *CRYPTO 2017*, volume 10403, pages 98–129. Springer, 2017.

[FKKP19] Georg Fuchsbauer, Chethan Kamath, Karen Klein, and Krzysztof Pietrzak. Adaptively secure proxy re-encryption. In *PKC 2019*, volume 11443, pages 317–346. Springer, 2019.

[FL17] Xiong Fan and Feng-Hao Liu. Proxy re-encryption and re-signatures from lattices. Cryptology ePrint Archive, Report 2017/456, 2017. https://eprint.iacr.org/2017/456.

[Gam85] Taher El Gamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Trans. Inf. Theory*, 31(4):469–472, 1985.

[Gen09] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *ACM STOC 2009*, pages 169–178, 2009.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. $i$-hop homomorphic encryption and rerandomizable yao circuits. In *CRYPTO 2010*, volume 6223, pages 155–172. Springer, 2010.

[GP23] Yao Jiang Galteland and Jiaxin Pan. Backward-leak uni-directional updatable encryption from public key encryption. In *PKC 2023 (to appear)*, 2023. https://eprint.iacr.org/2022/324.

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

[HILL99] Johan Håstad, Russell Impagliazzo, Leonid A. Levin, and Michael Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[ID03] Anca-Andreea Ivan and Yevgeniy Dodis. Proxy cryptography revisited. In *NDSS 2003*. The Internet Society, 2003.

[Jia20] Yao Jiang. The direction of updatable encryption does not matter much. In *ASIACRYPT 2020*, volume 12493, pages 529–558. Springer, 2020.

[JMM19] Daniel Jost, Ueli Maurer, and Marta Mularczyk. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In *EUROCRYPT 2019*, volume 11476, pages 159–188. Springer, 2019.

[Kir14] Elena Kirshanova. Proxy re-encryption from lattices. In *PKC 2014*, volume 8383, pages 77–94. Springer, 2014.

[KLR19] Michael Klooß, Anja Lehmann, and Andy Rupp. (R)CCA secure updatable encryption with integrity protection. In *EUROCRYPT 2019*, volume 11476, pages 68–99. Springer, 2019.

[LT18] Anja Lehmann and Björn Tackmann. Updatable encryption with post-compromise security. In *EUROCRYPT 2018*, volume 10822, pages 685–716. Springer, 2018.

[LV11] Benoît Libert and Damien Vergnaud. Unidirectional chosen-ciphertext secure proxy re-encryption. *IEEE Trans. Inf. Theory*, 57(3):1786–1802, 2011.

[NAL15] David Nuñez, Isaac Agudo, and Javier López. Ntrureencrypt: An efficient proxy re-encryption scheme based on NTRU. In *ACM ASIA CCS 2015*, 2015.

[Nis21] Ryo Nishimaki. The direction of updatable encryption does matter. Cryptology ePrint Archive, Report 2021/221, 2021. https://eprint.iacr.org/2021/221.

[NS12] Moni Naor and Gil Segev. Public-key cryptosystems resilient to key leakage. *SIAM J. Comput.*, 41(4):772–814, 2012.

[Nun18] DAVID Nunez. Umbral: a threshold proxy re-encryption scheme. *NuCypher Inc and NICS Lab, University of Malaga, Spain*, 2018.

[NX15] Ryo Nishimaki and Keita Xagawa. Key-private proxy re-encryption from lattices, revisited. *IEICE Transactions*, 98-A(1):100–116, 2015.

[Pay18] Payment Card Industry (PCI). Data Security Standard – Version 3.2.1, May 2018.

[PRSV17] Yuriy Polyakov, Kurt Rohloff, Gyana Sahu, and Vinod Vaikuntanathan. Fast proxy re-encryption for publish/subscribe systems. *ACM Trans. Priv. Secur.*, 20(4):14:1–14:31, 2017.

[PWA+16] Le Trieu Phong, Lihua Wang, Yoshinori Aono, Manh Ha Nguyen, and Xavier Boyen. Proxy re-encryption schemes with key privacy from LWE. Cryptology ePrint Archive, Report 2016/327, 2016. https://eprint.iacr.org/2016/327.

[SD19] Vipin Singh Sehrawat and Yvo Desmedt. Bi-homomorphic lattice-based prfs and unidirectional updatable encryption. In *CANS 2019*, volume 11829, pages 3–23. Springer, 2019.

[Sho06] Victor Shoup. *A computational introduction to number theory and algebra*. Cambridge University Press, 2006.

[SS21] Daniel Slamanig and Christoph Striecks. Puncture 'em all: Stronger updatable encryption with no-directional key updates. Cryptology ePrint Archive, Report 2021/268, 2021. https://eprint.iacr.org/2021/268.

# A Hybrid Arguments in Proof of IND-ENC Security (Theorem 3.5)

In this section, we present the proofs for the hybrid arguments in Theorem 3.5, which establishes the IND-ENC security of our UE construction detailed in Section 3.2. In particular, we formally prove Lemmas 3.6 and 3.7.

## A.1 Proof of Lemma 3.6

We first present the detailed proof of Lemma 3.6.

*Proof.* Let $Q$ be the upper bound on the total number of $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}-1})$ and $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \widetilde{\mathsf{ct}}_{\mathsf{e}-1})$ outputs. We construct a series of $Q$ intermediate hybrids $\{\mathsf{Hyb}_{0,q}\}_{q\in[Q]}$ where in each hybrid we only change the $q$-th output of $\mathsf{UE.upd}$ (one by one from the first to the last) as described in $\mathsf{Hyb}_1$. Note that $\mathsf{Hyb}_0 = \mathsf{Hyb}_{0,0}$ and $\mathsf{Hyb}_1 = \mathsf{Hyb}_{0,Q}$. In the following we argue that changing the $q$-th $\mathsf{UE.upd}$ output is computationally indistinguishable to any PPT adversary, namely $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$.

For the $q$-th output of $\mathsf{UE.upd}$, without loss of generality we assume it is $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}-1})$ handled in $\mathcal{O}.\mathsf{upd}$.

**Intermediate hybrid $\mathsf{Hyb}'_{0,q-1}$.** We first construct an intermediate hybrid $\mathsf{Hyb}'_{0,q-1}$ between $\mathsf{Hyb}_{0,q-1}$ and $\mathsf{Hyb}_{0,q}$ that computes $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}-1})$ as follows:

- Let $\mathtt{k}_{\mathsf{e}-1} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}-1})$.
- Parse the ciphertext $\mathsf{ct}_{\mathsf{e}-1}$ as

$$(t, (\overline{\mathsf{pk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-2}, \overline{\mathsf{ctx}}_{\mathsf{e}-2}), (\mathsf{pk}_{\mathsf{e}-1}, \mathsf{ctx}_{\mathsf{e}-1})),$$

  where $\mathsf{ctx}_{\mathsf{e}-1} = \mathsf{KPHE.Enc}(\mathsf{pk}_{\mathsf{e}-1}, x)$. Note that if $t = 0$, then $x = \mathsf{m}$ for some message, otherwise $x = \overline{\mathsf{sk}}_{\mathsf{e}-2}$ that is the $\mathsf{KPHE}$ secret key corresponding to $\overline{\mathsf{pk}}_{\mathsf{e}-2}$.
- Sample a uniform random permutation $\pi : [2n] \to [2n]$, let $\overline{\mathsf{sk}}_{\mathsf{e}-1} = \pi(\mathsf{sk}_{\mathsf{e}-1})$, and sample $\overline{\mathsf{pk}}_{\mathsf{e}-1} \overset{\$}{\leftarrow} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_{\mathsf{e}-1})$. Compute $\overline{\mathsf{ctx}}_{\mathsf{e}-1} \overset{\$}{\leftarrow} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_{\mathsf{e}-1}, x)$.
- $(\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}})$ are computed same as in $\mathsf{Hyb}_{0,q-1}$ using $\pi$ and $\pi_{\mathsf{id}}$. That is, let $\Delta_{\mathsf{e}} = (\mathsf{pk}_{\Delta}, \mathsf{ctx}_{\Delta})$, compute $(\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}}) \overset{\$}{\leftarrow} \mathsf{KPHE.Eval}(\mathsf{pk}_{\Delta}, \mathsf{ctx}_{\Delta}, \pi_{\mathsf{id}}, \pi)$.
- Let $\mathsf{ct}_{\mathsf{e}}$ be

$$(t+1, (\overline{\mathsf{pk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1}), (\mathsf{pk}_{\mathsf{e}}, \mathsf{ctx}_{\mathsf{e}})).$$

**$\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}'_{0,q-1}$.** We first argue $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}'_{0,q-1}$ by the blinding property of $\mathsf{KPHE}$. Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish the two hybrids. Then we construct a PPT adversary $\mathcal{B}$ that breaks the blinding property of $\mathsf{KPHE}$. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{sk}, \mathsf{pk})$ from the challenger in the $\mathsf{KPHE}$ blinding experiment. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}_{0,q-1}$.

Let $E$ be the upper bound on the number of epochs during the UE game. $\mathcal{B}$ randomly guesses $\mathsf{e}^\star$ from $[E]$ as the epoch where the target $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}-1})$ computation occurs. $\mathcal{B}$ uses $\mathsf{pp}$ to generate UE keys and update tokens as in $\mathsf{Hyb}_{0,q-1}$ except that in epoch $(\mathsf{e}^\star - 1)$, it uses $(\mathsf{pp}, \mathsf{sk})$ as $\mathtt{k}_{\mathsf{e}^\star-1}$.

Let $C$ be the upper bound on the number of ciphertexts generated from $\mathsf{UE.enc}$ and $\mathsf{UE.upd}$ during epoch $(\mathsf{e}^\star - 1)$. $\mathcal{B}$ randomly guesses $c$ from $[C]$ as the target ciphertext $\mathsf{ct}_{\mathsf{e}^\star-1}$. In the computation of $\mathsf{ct}_{\mathsf{e}^\star-1}$, first set $\mathsf{pk}_{\mathsf{e}^\star-1} = \mathsf{pk}$ (from the $\mathsf{KPHE}$ blinding experiment). Assume $\mathsf{ctx}_{\mathsf{e}^\star-1}$

should be an encryption of $x$ under $\mathsf{pk}_{\mathsf{e}^\star-1}$ for some $x$. $\mathcal{B}$ sends $\mathsf{m} = y$ to the challenger in the KPHE blinding experiment and gets back a ciphertext $\mathsf{ct}$. Let $\mathsf{ctx}_{\mathsf{e}^\star-1} := \mathsf{ct}$.

If the $q$-th UE.upd computation does not occur in epoch $\mathsf{e}^\star$ or if it is not computed on the above $\mathsf{ct}_{\mathsf{e}^\star-1}$, namely $\mathcal{B}$'s guess is wrong, then $\mathcal{B}$ aborts the UE game and outputs a random bit $b \in \{0,1\}$. If $B$'s guess is correct, then $\mathcal{B}$ computes the target $\mathsf{UE.upd}(\Delta_{\mathsf{e}^\star}, \mathsf{ct}_{\mathsf{e}^\star-1})$ as follows:

- First note that $\mathtt{k}_{\mathsf{e}^\star-1} = (\mathsf{pp}, \mathsf{sk})$ received in the KPHE blinding experiment.
- Parse the ciphertext $\mathsf{ct}_{\mathsf{e}^\star-1}$ as

$$(t, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1-t}), \dots, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-2}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-2}), (\mathsf{pk}_{\mathsf{e}^\star-1}, \mathsf{ctx}_{\mathsf{e}^\star-1})),$$

  where $\mathsf{ctx}_{\mathsf{e}^\star-1} = \mathsf{KPHE.Enc}(\mathsf{pk}_{\mathsf{e}^\star-1}, x)$, which is computed by the challenger in the KPHE blinding experiment.
- Sample a uniform random permutation $\pi : [2n] \to [2n]$. Send $(\pi, \pi_{\mathsf{id}})$ to the challenger in the KPHE blinding experiment and get back $(\overline{\mathsf{pk}}_{\mathsf{e}^\star-1}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1})$.
- $(\mathsf{pk}_{\mathsf{e}^\star}, \mathsf{ctx}_{\mathsf{e}^\star})$ are computed same as in $\mathsf{Hyb}_{0,q-1}$ using $\pi$ and $\pi_{\mathsf{id}}$. That is, let $\Delta_{\mathsf{e}^\star} = (\mathsf{pk}_\Delta, \mathsf{ctx}_\Delta)$, compute $(\mathsf{pk}_{\mathsf{e}^\star}, \mathsf{ctx}_{\mathsf{e}^\star}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_\Delta, \mathsf{ctx}_\Delta, \pi_{\mathsf{id}}, \pi)$.
- Let $\mathsf{ct}_{\mathsf{e}^\star}$ be

$$(t+1, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1-t}), \dots, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1}), (\mathsf{pk}_{\mathsf{e}^\star}, \mathsf{ctx}_{\mathsf{e}^\star})).$$

Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. Note that if the challenger in the KPHE blinding experiment responds to $\mathcal{B}$ with the output of KPHE.Eval, then the game is identical to $\mathsf{Hyb}_{0,q-1}$ to $\mathcal{A}$; otherwise the game is identical to $\mathsf{Hyb}'_{0,q-1}$ to $\mathcal{A}$. If $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, then $\mathcal{B}$ can break the blinding property of KPHE with non-negligible probability, which leads to contradiction.

**$\mathsf{Hyb}'_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$.** We next argue $\mathsf{Hyb}'_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ also by the blinding property of KPHE. Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish between the two hybrids. Then we construct a PPT adversary $\mathcal{B}$ that breaks the blinding property of KPHE. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{sk}, \mathsf{pk})$ from the challenger in the KPHE blinding experiment. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}'_{0,q-1}$.

Let $E$ be the upper bound on the number of epochs during the UE game. $\mathcal{B}$ randomly guesses $\mathsf{e}^\star$ from $[E]$ as the epoch where the target $\mathsf{UE.upd}(\Delta_{\mathsf{e}}, \mathsf{ct}_{\mathsf{e}-1})$ computation occurs. $\mathcal{B}$ uses $\mathsf{pp}$ to generate UE keys and update tokens as in $\mathsf{Hyb}'_{0,q-1}$ except that in epoch $\mathsf{e}^\star$, it uses $(\mathsf{pp}, \mathsf{sk})$ as $\mathtt{k}_{\mathsf{e}^\star}$. Additionally, $\mathcal{B}$ sends $\mathsf{m} = \mathsf{sk}_{\mathsf{e}^\star-1}$ in the KPHE blinding experiment and receives $\mathsf{ctx}$. Then $\mathcal{B}$ sets $\Delta_{\mathsf{e}^\star} = (\mathsf{pk}, \mathsf{ctx})$.

If the $q$-th UE.upd computation does not occur in epoch $\mathsf{e}^\star$, namely $\mathcal{B}$'s guess is wrong, then $\mathcal{B}$ aborts the UE game and outputs a random bit $b \in \{0,1\}$. If $B$'s guess is correct, then $\mathcal{B}$ computes the target $\mathsf{UE.upd}(\Delta_{\mathsf{e}^\star}, \mathsf{ct}_{\mathsf{e}^\star-1})$ as follows:

- Let $\mathtt{k}_{\mathsf{e}^\star-1} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}^\star-1})$. Note that $\Delta_{\mathsf{e}^\star} = (\mathsf{pk}, \mathsf{ct})$, where $\mathsf{ct} = \mathsf{KPHE.Enc}(\mathsf{pk}, \mathsf{sk}_{\mathsf{e}^\star-1})$ computed by the challenger in the KPHE blinding experiment.

- Parse the ciphertext $\mathsf{ct}_{\mathsf{e}^\star-1}$ as

$$(t, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-2}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-2}), (\mathsf{pk}_{\mathsf{e}^\star-1}, \mathsf{ctx}_{\mathsf{e}^\star-1})),$$

  where $\mathsf{ctx}_{\mathsf{e}^\star-1} = \mathsf{KPHE.Enc}(\mathsf{pk}_{\mathsf{e}^\star-1}, x)$. Note that if $t = 0$, then $x = \mathsf{m}$ for some message, otherwise $x = \overline{\mathsf{sk}}_{\mathsf{e}^\star-2}$ that is the KPHE secret key corresponding to $\overline{\mathsf{pk}}_{\mathsf{e}^\star-2}$.

- Sample a uniform random permutation $\pi : [2n] \to [2n]$, let $\overline{\mathsf{sk}}_{\mathsf{e}^\star-1} = \pi(\mathsf{sk}_{\mathsf{e}^\star-1})$, and sample $\overline{\mathsf{pk}}_{\mathsf{e}^\star-1} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_{\mathsf{e}^\star-1})$. Compute $\overline{\mathsf{ctx}}_{\mathsf{e}^\star-1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_{\mathsf{e}^\star-1}, x)$.

- Send $(\pi_{\mathsf{id}}, \pi)$ to the challenger in the KPHE blinding experiment and get back $(\mathsf{pk}_{\mathsf{e}^\star}, \mathsf{ctx}_{\mathsf{e}^\star})$.

- Let $\mathsf{ct}_{\mathsf{e}^\star}$ be

$$(t+1, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1-t}), \ldots, (\overline{\mathsf{pk}}_{\mathsf{e}^\star-1}, \overline{\mathsf{ctx}}_{\mathsf{e}^\star-1}), (\mathsf{pk}_{\mathsf{e}^\star}, \mathsf{ctx}_{\mathsf{e}^\star})).$$

Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. Note that if the challenger in the KPHE blinding experiment responds to $\mathcal{B}$ with the output of KPHE.Eval, then the game is identical to $\mathsf{Hyb}'_{0,q-1}$ to $\mathcal{A}$; otherwise the game is identical to $\mathsf{Hyb}_{0,q}$ to $\mathcal{A}$. If $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, then $\mathcal{B}$ can break the blinding property of KPHE with non-negligible probability, which leads to contradiction. This concludes our proof. □

### A.2 Proof of Lemma 3.7

We next present the detailed proof of Lemma 3.7.

*Proof.* Between $\mathsf{Hyb}_4$ and $\mathsf{Hyb}_5$, we construct a series of intermediate hybrids $\mathsf{Hyb}_{4,\bar{\mathsf{e}}}, \mathsf{Hyb}_{4,\bar{\mathsf{e}}-1}, \ldots, \mathsf{Hyb}_{4,\tilde{\mathsf{e}}+1}$ where in each hybrid we change a single $\Delta_{\mathsf{e}}$ from $(\widehat{\mathsf{pk}}_{\mathsf{e}}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\mathsf{e}}, \mathsf{sk}_{\mathsf{e}-1}))$ to $(\widehat{\mathsf{pk}}_{\mathsf{e}}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\mathsf{e}}, 0^{2n}))$, one by one from $\bar{\mathsf{e}}$ down to $\tilde{\mathsf{e}}+1$. Note that $\mathsf{Hyb}_4 = \mathsf{Hyb}_{4,\bar{\mathsf{e}}+1}$ and $\mathsf{Hyb}_5 = \mathsf{Hyb}_{4,\tilde{\mathsf{e}}+1}$. In the following we argue that changing a single $\Delta_{\mathsf{e}}$ is computationally indistinguishable to any PPT adversary, namely $\mathsf{Hyb}_{4,\mathsf{e}+1} \overset{c}{\approx} \mathsf{Hyb}_{4,\mathsf{e}}$ for all $\bar{\mathsf{e}} \geq \mathsf{e} \geq \tilde{\mathsf{e}}+1$.

Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish between $\mathsf{Hyb}_{4,\mathsf{e}+1}$ and $\mathsf{Hyb}_{4,\mathsf{e}}$ for some $\bar{\mathsf{e}} \geq \mathsf{e} \geq \tilde{\mathsf{e}}+1$. Then we construct a PPT adversary $\mathcal{B}$ that breaks the distributional semantic security of KPHE. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{pk})$ from the challenger in the KPHE semantic security game. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}_{4,\mathsf{e}+1}$. $\mathcal{B}$ uses $\mathsf{pp}$ to generate UE keys and update tokens as in $\mathsf{Hyb}_{4,\mathsf{e}+1}$ except that for epoch $\mathsf{e}$, the UE key $\mathtt{k}_{\mathsf{e}}$ is unknown. In addition, $\mathcal{B}$ sends $(\mathsf{m}_0 = \mathsf{sk}_{\mathsf{e}-1}, \mathsf{m}_1 = 0^{2n})$ to the KPHE challenger and gets back a ciphertext $\mathsf{ctx}$. Then $\mathcal{B}$ sets $\Delta_{\mathsf{e}} = (\mathsf{pk}, \mathsf{ctx})$. Note that $\mathcal{B}$ doesn't need to know $\mathtt{k}_{\mathsf{e}}$ because it is never used in $\mathsf{Hyb}_{4,\mathsf{e}+1}$ or $\mathsf{Hyb}_{4,\mathsf{e}}$. In particular, $\mathcal{B}$ can use $\mathsf{pk}$ to compute all the $\mathsf{KPHE.Enc}(\mathtt{k}_{\mathsf{e}}, \cdot)$. Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

Note that if the challenger in the KPHE leakage resilience experiment responds to $\mathcal{B}$ with an encryption of $\mathsf{m}_0$, then the UE game is identical to $\mathsf{Hyb}_{4,\mathsf{e}+1}$ to $\mathcal{A}$; otherwise the UE game is identical to $\mathsf{Hyb}_{4,\mathsf{e}}$ to $\mathcal{A}$. If $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, then $\mathcal{B}$ can break the distributional semantic security of KPHE with non-negligible probability, which leads to contradiction. This concludes our proof. □

## B Proof of IND-UPD Security (Theorem 3.9)

In this section, we prove Theorem 3.9, which formally establishes the IND-UPD security of our UE construction in Section 3.3.1.

*Proof.* The proof proceeds via a hybrid argument.

$\mathsf{Hyb}_0$ The challenger plays the real game with the adversary.

$\mathsf{Hyb}_1$ Same as $\mathsf{Hyb}_0$ but for $\mathsf{UE.upd}(\Delta_\mathsf{e}, \mathsf{ct}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{upd}$ and $\mathsf{UE.upd}(\Delta_\mathsf{e}, \widetilde{\mathsf{ct}}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{next}$, do the following:

- Let $\mathtt{k}_{\mathsf{e}-1} = (\mathsf{pp}, \mathsf{sk}_{\mathsf{e}-1})$ and $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$.
- Parse the ciphertext $\mathsf{ct}_{\mathsf{e}-1}$ or $\widetilde{\mathsf{ct}}_{\mathsf{e}-1}$ as

$$(t, (\overline{\mathsf{pk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}), \dots, (\overline{\mathsf{pk}}_{\mathsf{e}-2}, \overline{\mathsf{ctx}}_{\mathsf{e}-2}), (\mathsf{pk}_{\mathsf{e}-1}, \mathsf{ctx}_{\mathsf{e}-1})),$$

  where $\mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-1-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-1-t}) = \mathsf{m}, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-t}, \overline{\mathsf{ctx}}_{\mathsf{e}-t}) = \overline{\mathsf{sk}}_{\mathsf{e}-1-t}, \dots, \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{\mathsf{e}-2}, \overline{\mathsf{ctx}}_{\mathsf{e}-2}) = \overline{\mathsf{sk}}_{\mathsf{e}-3}, \mathsf{KPHE.Dec}(\mathsf{sk}_{\mathsf{e}-1}, \mathsf{ctx}_{\mathsf{e}-1}) = \overline{\mathsf{sk}}_{\mathsf{e}-2}$.
- Sample $(t+1)$ uniform random permutations $\pi_{\mathsf{e}-1-t}, \dots, \pi_{\mathsf{e}-1} : [2n] \to [2n]$. Also, let $\pi_{\mathsf{id}} : [2n] \to [2n]$ denote the identity permutation.
- For each $i \in \{\mathsf{e}-1-t, \dots, \mathsf{e}-2\}$, let $\widetilde{\mathsf{sk}}_i = \pi_i(\overline{\mathsf{sk}}_i)$ and sample $\widetilde{\mathsf{pk}}_i \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_i)$. Additionally, let $\overline{\mathsf{sk}}_{\mathsf{e}-1} = \pi_{\mathsf{e}-1}(\mathsf{sk}_{\mathsf{e}-1})$ and sample $\overline{\mathsf{pk}}_{\mathsf{e}-1} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_{\mathsf{e}-1})$. Sample $\mathsf{pk}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$.
- For each $i \in \{\mathsf{e}-t, \dots, \mathsf{e}-2\}$, compute $\widetilde{\mathsf{ctx}}_i \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_i, \widetilde{\mathsf{sk}}_{i-1})$. Additionally, compute

$$\begin{aligned}
\widetilde{\mathsf{ctx}}_{\mathsf{e}-1-t} &\xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_{\mathsf{e}-1-t}, \mathsf{m}),\\
\overline{\mathsf{ctx}}_{\mathsf{e}-1} &\xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_{\mathsf{e}-1}, \widetilde{\mathsf{sk}}_{\mathsf{e}-2}),\\
\mathsf{ctx}_\mathsf{e} &\xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, \overline{\mathsf{sk}}_{\mathsf{e}-1}),
\end{aligned}$$

- Let $\mathsf{ct}_\mathsf{e}$ or $\widetilde{\mathsf{ct}}_\mathsf{e}$ be

$$(t+1, (\widetilde{\mathsf{pk}}_{\mathsf{e}-1-t}, \widetilde{\mathsf{ctx}}_{\mathsf{e}-1-t}), \dots, (\widetilde{\mathsf{pk}}_{\mathsf{e}-2}, \widetilde{\mathsf{ctx}}_{\mathsf{e}-2}), (\overline{\mathsf{pk}}_{\mathsf{e}-1}, \overline{\mathsf{ctx}}_{\mathsf{e}-1})(\mathsf{pk}_\mathsf{e}, \mathsf{ctx}_\mathsf{e})).$$

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_0$ to any PPT adversary by the blinding property of $\mathsf{KPHE}$. We omit the detailed proof here, but it follows similarly to the proof of Lemma 3.6.

$\mathsf{Hyb}_2$ Same as $\mathsf{Hyb}_1$ but for $\mathsf{UE.upd}(\Delta_\mathsf{e}, \mathsf{ct}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{upd}$ and $\mathsf{UE.upd}(\Delta_\mathsf{e}, \widetilde{\mathsf{ct}}_{\mathsf{e}-1})$ in $\mathcal{O}.\mathsf{next}$, sample each $\widetilde{\mathsf{sk}}_{\mathsf{e}-1-t}, \dots, \widetilde{\mathsf{sk}}_{\mathsf{e}-2}, \overline{\mathsf{sk}}_{\mathsf{e}-1}$ from the distribution $\mathcal{U}_n$. This hybrid is statistically identical to $\mathsf{Hyb}_1$.

$\mathsf{Hyb}_3$ Let $\widetilde{\mathsf{e}}$ be the challenge epoch, and let $\overline{\mathsf{e}}$ be the last epoch where the adversary corrupts continuous update tokens from $\widetilde{\mathsf{e}}$, namely the adversary corrupts $\Delta_{\widetilde{\mathsf{e}}+1}, \Delta_{\widetilde{\mathsf{e}}+2}, \ldots, \Delta_{\overline{\mathsf{e}}}$ but not $\Delta_{\overline{\mathsf{e}}+1}$. This hybrid is the same as $\mathsf{Hyb}_3$ except that the challenger guesses $\widetilde{\mathsf{e}}^\star$ and $\overline{\mathsf{e}}^\star$ at the beginning of the game and aborts the game if guessing incorrectly. Let $E$ be the upper bound on the number of epochs during the game. If the challenger does not abort, then this hybrid is identical to $\mathsf{Hyb}_2$, which happens with probability at least $\frac{1}{E^2}$. In the remaining hybrids, we assume for simplicity that the challenger guesses $\widetilde{\mathsf{e}}$ and $\overline{\mathsf{e}}$ correctly.

$\mathsf{Hyb}_4$ Same as $\mathsf{Hyb}_3$ except that for each $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$, generate a single public key $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$. Then whenever $\mathsf{KPHE.Enc}(\mathsf{pk}_\mathsf{e}, x)$ is computed for a freshly generated $\mathsf{pk}_\mathsf{e}$ and some $x$, compute it as $\mathsf{KPHE.Eval}(\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, x), \pi_{\mathsf{id}}, \pi_{\mathsf{id}})$. That is, instead of generating a fresh $\mathsf{pk}_\mathsf{e}$ from $\mathsf{sk}_\mathsf{e}$ every time, use the same $\widehat{\mathsf{pk}}_\mathsf{e}$ to encrypt $x$ and use then $\mathsf{KPHE.Eval}$ to re-randomize it.

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_3$ by the blinding property of $\mathsf{KPHE}$. We omit the detailed reduction here, but it is similar to the reduction in the proof of Lemma 3.6.

$\mathsf{Hyb}_5$ Same as $\mathsf{Hyb}_4$ except that for all $\widetilde{\mathsf{e}}+1 \leq \mathsf{e} \leq \overline{\mathsf{e}}$, $\mathsf{UE.next}(\mathtt{k}_{\mathsf{e}-1})$ is computed as follows. Generate $\mathsf{sk}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.SKGen}(\mathsf{pp}, \mathcal{U}_n)$ and let $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ be the single public key for $\mathtt{k}_\mathsf{e}$ (that will be used for every $\mathsf{KPHE.Enc}$). Output

$$\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e}), \quad \Delta_\mathsf{e} = (\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, 0^{2n})).$$

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_4$ based on the distributional semantic security of $\mathsf{KPHE}$. We omit the detailed proof here, but it follows similarly to the proof of Lemma 3.7.

$\mathsf{Hyb}_6$ Same as $\mathsf{Hyb}_5$ except that for each $\mathtt{k}_\mathsf{e} = (\mathsf{pp}, \mathsf{sk}_\mathsf{e})$, generate a single public key $\widehat{\mathsf{pk}}_\mathsf{e} \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \mathsf{sk}_\mathsf{e})$ and use $\mathsf{KPHE.Eval}(\widehat{\mathsf{pk}}_\mathsf{e}, \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_\mathsf{e}, \cdot), \pi_{\mathsf{id}}, \pi_{\mathsf{id}})$ for all the computation of $\mathsf{KPHE.Enc}(\mathtt{k}_\mathsf{e}, \cdot)$ (including the computation of $\Delta_\mathsf{e}$). The only exception is in the computation of the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}} \xleftarrow{\$} \mathsf{UE.upd}(\Delta_{\widetilde{\mathsf{e}}}, \overline{\mathsf{ct}}_\mathrm{b})$, which is in the form of

$$\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}} = (t+1, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-2}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-2}), (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1})(\mathsf{pk}_{\widetilde{\mathsf{e}}}, \mathsf{ctx}_{\widetilde{\mathsf{e}}})),$$

where the last ciphertext is computed from $\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ directly, namely $\mathsf{pk}_{\widetilde{\mathsf{e}}} = \widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ and $\mathsf{ctx}_{\widetilde{\mathsf{e}}} \xleftarrow{\$} \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \overline{\mathsf{sk}}_{\widetilde{\mathsf{e}}-1})$.

This hybrid is computationally indistinguishable from $\mathsf{Hyb}_5$ by the blinding property of $\mathsf{KPHE}$. We omit the detailed reduction here, but it is similar to the reduction in the proof of Lemma 3.6.

$\mathsf{Hyb}_7$ Same as $\mathsf{Hyb}_6$ but the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ is computed as follows.

- Let $\mathtt{k}_{\widetilde{\mathsf{e}}} = (\mathsf{pp}, \mathsf{sk}_{\widetilde{\mathsf{e}}})$ and let $\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ be its corresponding public key.
- Parse the ciphertext $\overline{\mathsf{ct}}_0$ as

$$(t, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-2}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-2}), (\mathsf{pk}_{\widetilde{\mathsf{e}}-1}, \mathsf{ctx}_{\widetilde{\mathsf{e}}-1})),$$

  Note that $\overline{\mathsf{ct}}_1$ is of the same form because $|\overline{\mathsf{ct}}_0| = |\overline{\mathsf{ct}}_1|$.

- For each $i \in \{\widetilde{\mathsf{e}}-1-t, \ldots, \widetilde{\mathsf{e}}-1\}$, sample $\widetilde{\mathsf{sk}}_i$ from $\mathcal{U}_n$ and sample $\widetilde{\mathsf{pk}}_i \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_i)$.
- For each $i \in \{\widetilde{\mathsf{e}}-1-t, \ldots, \widetilde{\mathsf{e}}-1\}$, compute $\widetilde{\mathsf{ctx}}_i \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_i, 0^{2n})$. Additionally, compute $\mathsf{ctx}_{\widetilde{\mathsf{e}}} \xleftarrow{\$} \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, 0^{2n})$.
- Let the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ be

$$(t+1, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1}), (\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \mathsf{ctx}_{\widetilde{\mathsf{e}}})).$$

We prove in Lemma B.1 that this hybrid is computationally indistinguishable from $\mathsf{Hyb}_6$ based on the distributional semantic security of KPHE.

Notice that in the final hybrid $\mathsf{Hyb}_7$, $\mathsf{UE.upd}(\Delta_{\widetilde{\mathsf{e}}}, \overline{\mathsf{ct}}_{\mathrm{b}})$ is computed in the exact same way for $\mathrm{b} = 0$ and $\mathrm{b} = 1$. This concludes our proof.

**Lemma B.1.** $\mathsf{Hyb}_6 \overset{c}{\approx} \mathsf{Hyb}_7$ *in the proof of Theorem 3.9.*

*Proof.* Between $\mathsf{Hyb}_6$ and $\mathsf{Hyb}_7$, we construct a series of intermediate hybrids $\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}}, \mathsf{Hyb}_{6,\widetilde{\mathsf{e}}-1}, \ldots, \mathsf{Hyb}_{6,\widetilde{\mathsf{e}}-1-t}$ where in each hybrid we change a single ctx (in $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$) to a KPHE encryption of $0^{2n}$, one by one from $\widetilde{\mathsf{e}}$ down to $\widetilde{\mathsf{e}}-1-t$. That is, changing $\mathsf{ctx}_{\widetilde{\mathsf{e}}}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-2}, \ldots, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}$ to encryptions of $0^{2n}$ one by one in each hybrid. Note that $\mathsf{Hyb}_6 = \mathsf{Hyb}_{6,\widetilde{\mathsf{e}}+1}$ and $\mathsf{Hyb}_7 = \mathsf{Hyb}_{6,\widetilde{\mathsf{e}}-1-t}$. In the following we argue that changing a single ctx is computationally indistinguishable to any PPT adversary, namely $\mathsf{Hyb}_{6,\mathsf{e}+1} \overset{c}{\approx} \mathsf{Hyb}_{6,\mathsf{e}}$ for all $\widetilde{\mathsf{e}} \geq \mathsf{e} \geq \widetilde{\mathsf{e}}-1-t$.

$\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}+1} \overset{c}{\approx} \mathsf{Hyb}_{6,\widetilde{\mathsf{e}}}$**.** Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish between $\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}+1} (= \mathsf{Hyb}_6)$ and $\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}}$. Then we construct a PPT adversary $\mathcal{B}$ that breaks the distributional semantic security of KPHE. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{pk})$ from the challenger in the KPHE semantic security game. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}_6$. $\mathcal{B}$ uses pp to generate UE keys and update tokens as in $\mathsf{Hyb}_6$ except that for epoch $\widetilde{\mathsf{e}}$, the UE key $\mathbb{k}_{\widetilde{\mathsf{e}}}$ is unknown, but $\mathcal{B}$ uses pk as $\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ for all the computation of $\mathsf{KPHE.Enc}(\mathbb{k}_{\widetilde{\mathsf{e}}}, \cdot)$. In addition, the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ is computed as follows.

- Parse the ciphertext $\overline{\mathsf{ct}}_0$ as

$$(t, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-2}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-2}), (\mathsf{pk}_{\widetilde{\mathsf{e}}-1}, \mathsf{ctx}_{\widetilde{\mathsf{e}}-1})),$$

- For each $i \in \{\widetilde{\mathsf{e}}-1-t, \ldots, \widetilde{\mathsf{e}}-1\}$, sample $\widetilde{\mathsf{sk}}_i$ from $\mathcal{U}_n$ and sample $\widetilde{\mathsf{pk}}_i \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_i)$.
- For each $i \in \{\widetilde{\mathsf{e}}-1-t, \ldots, \widetilde{\mathsf{e}}-1\}$, compute $\widetilde{\mathsf{ctx}}_i$ same as in $\mathsf{Hyb}_6$.
- Send $(\mathsf{m}_0 = \widetilde{\mathsf{sk}}_{\widetilde{\mathsf{e}}-1}, \mathsf{m}_1 = 0^{2n})$ to the KPHE challenger and get back ctx, and set $\mathsf{ctx}_{\widetilde{\mathsf{e}}} := \mathsf{ctx}$.
- Let the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ be

$$(t+1, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1}), (\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \mathsf{ctx}_{\widetilde{\mathsf{e}}})).$$

Note that $\mathcal{B}$ doesn't need to know $\mathtt{k}_{\widetilde{\mathsf{e}}}$ because it is never used in $\mathsf{Hyb}_6$ or $\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}}$. In particular, $\mathcal{B}$ can use $\mathsf{pk}$ to compute all the $\mathsf{KPHE.Enc}(\mathtt{k}_{\widetilde{\mathsf{e}}}, \cdot)$. Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs.

Note that if the challenger in the KPHE leakage resilience experiment responds to $\mathcal{B}$ with an encryption of $\mathsf{m}_0$, then the UE game is identical to $\mathsf{Hyb}_6$ to $\mathcal{A}$; otherwise the UE game is identical to $\mathsf{Hyb}_{6,\widetilde{\mathsf{e}}}$ to $\mathcal{A}$. If $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, then $\mathcal{B}$ can break the distributional semantic security of KPHE with non-negligible probability, which leads to contradiction.

$\mathsf{Hyb}_{6,\mathsf{e}+1} \overset{c}{\approx} \mathsf{Hyb}_{6,\mathsf{e}}$ **for all** $\widetilde{\mathsf{e}}-1 \geq \mathsf{e} \geq \widetilde{\mathsf{e}}-1-t$ Assume for the purpose of contradiction that there exists a PPT adversary $\mathcal{A}$ that can distinguish between $\mathsf{Hyb}_{6,\mathsf{e}+1}$ and $\mathsf{Hyb}_{6,\mathsf{e}}$ for some $\widetilde{\mathsf{e}} - 1 \geq \mathsf{e} \geq \widetilde{\mathsf{e}}-1-t$. Then we construct a PPT adversary $\mathcal{B}$ that breaks the distributional semantic security of KPHE. The adversary $\mathcal{B}$ first receives $(\mathsf{pp}, \mathsf{pk})$ from the challenger in the KPHE semantic security game. Then $\mathcal{B}$ plays the UE game with $\mathcal{A}$ as a challenger in $\mathsf{Hyb}_{6,\mathsf{e}+1}$. $\mathcal{B}$ uses $\mathsf{pp}$ to generate UE keys and update tokens as in $\mathsf{Hyb}_{6,\mathsf{e}+1}$ except that the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ is computed as follows.

- Let $\mathtt{k}_{\widetilde{\mathsf{e}}} = (\mathsf{pp}, \mathsf{sk}_{\widetilde{\mathsf{e}}})$ and let $\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}$ be its corresponding public key.
- Parse the ciphertext $\overline{\mathsf{ct}}_0$ as
$$(t, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\overline{\mathsf{pk}}_{\widetilde{\mathsf{e}}-2}, \overline{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-2}), (\mathsf{pk}_{\widetilde{\mathsf{e}}-1}, \mathsf{ctx}_{\widetilde{\mathsf{e}}-1})),$$
- For each $i \in \{\widetilde{\mathsf{e}}-1-t, \ldots, \widetilde{\mathsf{e}}-1\} \backslash \{\mathsf{e}\}$, sample $\widetilde{\mathsf{sk}}_i$ from $\mathcal{U}_n$ and sample $\widetilde{\mathsf{pk}}_i \overset{\$}{\leftarrow} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_i)$. Set $\widetilde{\mathsf{pk}}_i := \mathsf{pk}$ (received from the KPHE challenger).
- For each $i \in \{\widetilde{\mathsf{e}} - 1 - t, \ldots, \mathsf{e} - 1\}$, compute $\widetilde{\mathsf{ctx}}_i \overset{\$}{\leftarrow} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_i, \widetilde{\mathsf{sk}}_{i-1})$. For each $i \in \{\mathsf{e} + 1, \ldots, \widetilde{\mathsf{e}} - 1\}$, compute $\widetilde{\mathsf{ctx}}_i \overset{\$}{\leftarrow} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_i, 0^{2n})$. Additionally, compute $\mathsf{ctx}_{\widetilde{\mathsf{e}}} \overset{\$}{\leftarrow} \mathsf{KPHE.Enc}(\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, 0^{2n})$.
- Send $(\mathsf{m}_0 = \widetilde{\mathsf{sk}}_{\mathsf{e}-1}, \mathsf{m}_1 = 0^{2n})$ to the KPHE challenger and get back a ciphertext $\mathsf{ctx}$, and let $\widetilde{\mathsf{ctx}}_{\mathsf{e}} := \mathsf{ctx}$.
- Let the challenge ciphertext $\widetilde{\mathsf{ct}}_{\widetilde{\mathsf{e}}}$ be
$$(t+1, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1-t}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1-t}), \ldots, (\widetilde{\mathsf{pk}}_{\widetilde{\mathsf{e}}-1}, \widetilde{\mathsf{ctx}}_{\widetilde{\mathsf{e}}-1}), (\widehat{\mathsf{pk}}_{\widetilde{\mathsf{e}}}, \mathsf{ctx}_{\widetilde{\mathsf{e}}})).$$

Finally, $\mathcal{B}$ outputs whatever $\mathcal{A}$ outputs. Note that if the challenger in the KPHE leakage resilience experiment responds to $\mathcal{B}$ with an encryption of $\mathsf{m}_0$, then the UE game is identical to $\mathsf{Hyb}_{6,\mathsf{e}+1}$ to $\mathcal{A}$; otherwise the UE game is identical to $\mathsf{Hyb}_{6,\mathsf{e}}$ to $\mathcal{A}$. If $\mathcal{A}$ can distinguish between the two hybrids with non-negligible probability, then $\mathcal{B}$ can break the distributional semantic security of KPHE with non-negligible probability, which leads to contradiction.

□

This concludes the proof of Theorem 3.9. □

## C Proof of IND-HRA Security (Theorem 4.6)

In this section, we prove Theorem 4.6, which establishes the IND-HRA security for our PRE construction in Section 4.

*Proof.* The proof follows a sequence of hybrids as outlined below. We use $[N]$ to denote the set of all keys (honest + corrupt), $\mathcal{K}_{\texttt{Honest}} \subset [N]$ to denote the set of honest keys, and $\mathcal{K}_{\texttt{Corrupt}} \subset [N]$ to denote the set of corrupt keys. These sets are fixed (adversarially) at the beginning of the game, before the adversary is allowed to issue re-key generation queries (this remains unchanged in each hybrid).

$\mathsf{Hyb}_0$**:** This hybrid is identical to the real IND-HRA security game between the challenger and the adversary.

$\mathsf{Hyb}_1$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_0$ except that the challenger locally maintains two additional tables $T_0$ and $T_1$ (initially empty), and does the following:

- For each honest encryption query of the form $\mathsf{Enc}(i, \overline{\mathsf{m}})$, the challenger adds to the local table $T_0$ an entry of the form $(i, \overline{\mathsf{ct}}, \overline{\mathsf{m}})$, where $\overline{\mathsf{ct}}$ is generated as $\overline{\mathsf{ct}} = \mathsf{KPHE.Enc}(\mathsf{pk}_i, \overline{\mathsf{m}})$.
- For each honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

  Suppose $\overline{\mathsf{ct}}_i$ is of the form:
  $$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$
  Recover the plaintext mesage $\overline{\mathsf{m}} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \widehat{\mathsf{ctx}}_0)$. Next, sample a uniform permutation $\pi : [2n] \to [2n]$ and set
  $$\overline{\mathsf{sk}}_0 = \pi(\mathsf{sk}_i), \quad \overline{\mathsf{pk}}_0 = \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_0).$$
  Also set
  $$\overline{\mathsf{ctx}}_0 \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_0, \mathsf{m}), \quad \overline{\mathsf{ctx}}_1 \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, \overline{\mathsf{sk}}_0),$$
  $$(\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \overline{\mathsf{ctx}}_1, \pi_{\mathsf{id}}, \pi_{\mathsf{id}}),$$
  where $\pi_{\mathsf{id}} : [2n] \to [2n]$ is the identity permutation. Output the re-encrypted ciphertext as:
  $$\overline{\mathsf{ct}}_j = (1, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), (\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1).$$
  Also, add to the local table $T_1$ an entry of the form $(j, \overline{\mathsf{ct}}_j, \overline{\mathsf{sk}}_0)$.

  Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):
  $$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$
  Recover the intermediate secret key $\overline{\mathsf{sk}}_{t-1} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \overline{\mathsf{ctx}}_t)$. Next, sample a uniform permutation $\pi_t : [2n] \to [2n]$ and set
  $$\overline{\mathsf{sk}}_t = \pi_t(\mathsf{sk}_i), \quad \overline{\mathsf{pk}}_t \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

Also set

$$\overline{\mathsf{ctx}}_t \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_t, \overline{\mathsf{sk}}_{t-1}), \quad \overline{\mathsf{ctx}}_{t+1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, \overline{\mathsf{sk}}_t),$$

$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \overline{\mathsf{ctx}}_{t+1}, \pi_{\mathsf{id}}, \pi_{\mathsf{id}}),$$

where $\pi_{\mathsf{id}} : [2n] \to [2n]$ is the identity permutation. Output the re-encrypted ciphertext as:

$$\overline{\mathsf{ct}}_j = ((t+1), (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

Also, add to the local table $T_1$ an entry of the form $(j, \overline{\mathsf{ct}}_j, \overline{\mathsf{sk}}_t)$.

The view of the IND-HRA adversary in this hybrid is computationally indistinguishable from that in the hybrid $\mathsf{Hyb}_0$ under the assumption the KPHE scheme satisfies public key and ciphertext blinding. We prove this formally in Lemma C.4.

**$\mathsf{Hyb}_2$:** This hybrid is identical to the hybrid $\mathsf{Hyb}_1$ except that the challenger does the following (it still locally maintains the tables $T_0$ and $T_1$ as in $\mathsf{Hyb}_1$): for each (valid) honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$, suppose $\overline{\mathsf{ct}}_i$ is of the form:

$$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$

Look up the local table $T_0$ for an entry of the form $(i, \overline{\mathsf{ct}}_i, \overline{\mathsf{m}}^*)$. Such an entry is guaranteed to exist. Set $\overline{\mathsf{m}} = \overline{\mathsf{m}}^*$. The rest of the simulation proceeds as in hybrid $\mathsf{Hyb}_1$.

Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):

$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$

Look up the local table $T_1$ for an entry of the form $(i, \overline{\mathsf{ct}}_i, \mathsf{sk}^*)$. Such an entry is guaranteed to exist. Set the intermediate secret key $\overline{\mathsf{sk}}_{t-1} = \mathsf{sk}^*$. The rest of the simulation proceeds as in hybrid $\mathsf{Hyb}_1$.

It is easy to see that, assuming that the KPHE scheme is correct with overwhelmingly large probability, the view of the IND-HRA adversary in this hybrid is statistically indistinguishable from that in the hybrid $\mathsf{Hyb}_1$.

**$\mathsf{Hyb}_3$:** This hybrid is identical to the hybrid $\mathsf{Hyb}_2$ except that the challenger does the following (it still locally maintains the tables $T_0$ and $T_1$ as in $\mathsf{Hyb}_2$): for each (valid) honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$, suppose $\overline{\mathsf{ct}}_i$ is of the form:

$$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$

Look up the local table $T_0$ for an entry of the form $(i, \overline{\mathsf{ct}}_i, \overline{\mathsf{m}}^*)$. Such an entry is guaranteed to exist. Set $\overline{\mathsf{m}} = \overline{\mathsf{m}}^*$. Next, set

$$\overline{\mathsf{sk}}_0 \xleftarrow{\$} \mathcal{U}_n, \quad \overline{\mathsf{pk}}_0 = \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_0).$$

The rest of the simulation proceeds as in hybrid $\mathsf{Hyb}_2$.

Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):

$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$

Look up the local table $T_1$ for an entry of the form $(i, \overline{\mathsf{ct}}_i, \mathsf{sk}^*)$. Such an entry is guaranteed to exist. Set the intermediate secret key $\overline{\mathsf{sk}}_{t-1} = \mathsf{sk}^*$. Next, set

$$\overline{\mathsf{sk}}_t \xleftarrow{\$} \mathcal{U}_n, \quad \overline{\mathsf{pk}}_t = \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

The rest of the simulation proceeds as in hybrid $\mathsf{Hyb}_2$.

It is easy to see that the view of the IND-HRA adversary in this hybrid is identical to that in the hybrid $\mathsf{Hyb}_2$, since the distributions of $\overline{\mathsf{sk}}_0$ and $\overline{\mathsf{pk}}_0$ (resp,. $\overline{\mathsf{sk}}_t$ and $\overline{\mathsf{pk}}_t$) remain unchanged from hybrid $\mathsf{Hyb}_2$.

**Remark C.1.** *Note that in* $\mathsf{Hyb}_3$*, the set of secret keys* $\{\mathsf{sk}_i\}_{i \in [N]}$ *(including both corrupt and honest secret keys) is no longer used by the challenger when answering* ReEnc *queries. In particular, all* ReEnc *queries are answered using only the knowledge of the public keys* $\{\mathsf{pk}_i\}_{i \in [N]}$*, and the local tables* $T_0$ *and* $T_1$ *maintained by the challenger.*

$\mathsf{Hyb}_4$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_3$ except for the manner in which the challenger answers the ReKeyGen queries issued by the adversary. In particular, given a query of the form $\mathsf{ReKeyGen}(i, j)$ such that $i \in \mathcal{K}_{\mathtt{Honest}}$ and $j \in \mathcal{K}_{\mathtt{Honest}}$, the challenger responds with $\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{KPHE.Enc}(\mathsf{pk}_j, 0^{2n}))$.

We argue that $\mathsf{Hyb}_4$ is indistinguishable from $\mathsf{Hyb}_3$ in a straightforward manner under the assumption that KPHE satisfies distributional circular security. This argument relies *crucially* on the fact that, as in $\mathsf{Hyb}_3$, $\mathsf{Hyb}_4$ allows the challenger to answer all ReEnc queries using only the knowledge of the public keys $\{\mathsf{pk}_i\}_{i \in [N]}$, and the local tables $T_0$ and $T_1$ maintained by the challenger. We prove this formally in Lemma C.5.

**Remark C.2.** *Note that in* $\mathsf{Hyb}_4$*, the set of honest secret keys* $\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\mathsf{Honest}}}$ *is no longer used by the challenger when answering* ReKeyGen *queries. Also, as in* $\mathsf{Hyb}_3$*, the set of honest secret keys* $\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\mathsf{Honest}}}$ *are also not used by the challenger when answering* ReEnc *queries. In other words,* $\mathsf{Hyb}_4$ *allows the challenger to entirely "forget" the set of honest secret keys.*

$\mathsf{Hyb}_5$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_4$ except for the manner in which the challenger answers the challenge encryption query issued by the adversary. In particular, given a query of the form $\mathcal{O}.\mathsf{chall}(i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$, the challenger proceeds as follows:

- If $i \in \mathcal{K}_{\mathtt{Corrupt}}$, respond with $\overline{\mathsf{ct}}^* = \perp$ (this is exactly as in $\mathsf{Hyb}_4$).
- If $i \in \mathcal{K}_{\mathtt{Honest}}$, respond with $\overline{\mathsf{ct}}^* = (0, \mathsf{KPHE.Enc}(\mathsf{pk}_i, 0^{2n}))$.

We argue that $\mathsf{Hyb}_5$ is indistinguishable from $\mathsf{Hyb}_4$ under the assumption that KPHE satisfies distributional semantic security. This argument relies *crucially* on the fact that, as in $\mathsf{Hyb}_4$, $\mathsf{Hyb}_5$ allows the challenger to answer all ReKeyGen queries and ReEnc queries using only the knowledge of the corrupt secret keys $\{\mathsf{sk}_j\}_{j \in \mathcal{K}_{\mathsf{Corrupt}}}$, the set of all public keys $\{\mathsf{pk}_i\}_{i \in [N]}$, and the local tables $T_0$ and $T_1$ maintained by the challenger. We prove this formally in Lemma C.6.

**Remark C.3.** *Note that in* $\mathsf{Hyb}_5$*, the challenge ciphertext* $\overline{\mathsf{ct}}^*$ *is independent of* $(\overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$*, and hence the adversary's advantage in winning the IND-HRA game is zero.*

**Lemma C.4.** $\mathsf{Hyb}_0 \overset{c}{\approx} \mathsf{Hyb}_1$ *in the proof of Theorem 4.6.*

**Lemma C.5.** $\mathsf{Hyb}_3 \overset{c}{\approx} \mathsf{Hyb}_4$ *in the proof of Theorem 4.6.*

**Lemma C.6.** $\mathsf{Hyb}_4 \overset{c}{\approx} \mathsf{Hyb}_5$ *in the proof of Theorem 4.6.*

In what follows, we prove these lemmas formally. These proofs complete the overall proof of Theorem 4.6. □

## C.1 Proof of Lemma C.4

We first present the detailed proof of Lemma C.4. The proof is essentially identical to the proof of Lemma 3.6 in the proof of IND-ENC security of our UE construction, with only minor syntactic changes to account for the differences between the UE and the PRE schemes. Let $Q$ be the upper bound on the total number of $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ queries issued by the adversary $\mathcal{A}$ in hybrid $\mathsf{Hyb}_0$. We construct a series of $Q$ intermediate hybrids $\{\mathsf{Hyb}_{0,q}\}_{q\in[Q]}$ where in each hybrid we only change the $q$-th output of $\mathsf{ReEnc}$ (one by one from the first to the last) as described in $\mathsf{Hyb}_1$. Note that $\mathsf{Hyb}_0 = \mathsf{Hyb}_{0,0}$ and $\mathsf{Hyb}_1 = \mathsf{Hyb}_{0,Q}$. In the following we argue that changing the $q$-th $\mathsf{ReEnc}$ output is computationally indistinguishable to any PPT adversary, namely $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$.

**Intermediate Hybrid.** To argue that $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$, we first define an intermediate hybrid $\mathsf{Hyb}'_{0,q-1}$ as follows: this hybrid is identical to $\mathsf{Hyb}_{0,q-1}$, except that the challenger answers the $q$-th $\mathsf{ReEnc}$ query as follows: suppose that the $q$-th honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$. The challenger fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

- Suppose $\overline{\mathsf{ct}}_i$ is of the form:
$$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$
Recover the plaintext mesage $\overline{\mathsf{m}} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \widehat{\mathsf{ctx}}_0)$, sample a uniform permutation $\pi : [2n] \to [2n]$ and do the following:
  - Set $(\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0)$ as in hybrid $\mathsf{Hyb}_{0,q}$.
  - Set $(\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1)$ as in hybrid $\mathsf{Hyb}_{0,q-1}$.
  - Output the re-encrypted ciphertext as:
$$\overline{\mathsf{ct}}_j = (1, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), (\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1).$$
- Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):
$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$
Recover the intermediate secret key $\overline{\mathsf{sk}}_{t-1} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \overline{\mathsf{ctx}}_t)$, sample a uniform permutation $\pi_t : [2n] \to [2n]$ and do the following:
  - Set $(\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t)$ as in hybrid $\mathsf{Hyb}_{0,q}$.
  - Set $(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})$ as in hybrid $\mathsf{Hyb}_{0,q-1}$.
  - Output the re-encrypted ciphertext as:
$$\overline{\mathsf{ct}}_j = ((t+1), (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

$\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}'_{0,q-1}$. We argue that $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}'_{0,q-1}$ for all $q \in [Q]$.

*Proof.* Suppose that there exists some PPT adversary $\mathcal{A}$ that can efficiently distinguish $\mathsf{Hyb}_{0,q-1}$ and $\mathsf{Hyb}'_{0,q-1}$ with non-negligible advantage $\epsilon$. We construct a PPT algorithm $\mathcal{B}$ that breaks the blinding security of the KPHE scheme with non-negligible advantage $\epsilon'$. The algorithm $\mathcal{B}$ receives from the challenger in the blinding security game the public parameters $\mathsf{pp}$, a secret key $\mathsf{sk}^*$, and a public key $\mathsf{pk}^*$, and proceeds as follows.

- $\mathcal{B}$ uses $\mathsf{pp}$ to set up the PRE keys (both honest and corrupt) to be provided to the adversary $\mathcal{A}$ exactly as in $\mathsf{Hyb}_4$ except that it sets $\mathsf{sk}_i := \mathsf{sk}^*$ and $\mathsf{pk}_i := \mathsf{pk}^*$ for some uniform $i \xleftarrow{\$} \mathcal{K}_{\mathtt{Honest}}$. Since $\mathsf{pk}^*$ is uniformly random, the view of the adversary $\mathcal{A}$ with respect to the distribution of the keys (both honest and corrupt) remains identical.
- Suppose that the $q$-th honest re-encryption query of the form $\mathsf{ReEnc}(i', j, \overline{\mathsf{ct}}_{i'})$ to which the challenger does not respond with $\perp$. If $i \neq i'$, $\mathcal{B}$ outputs $\perp$. Otherwise, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.
- Suppose $\overline{\mathsf{ct}}_i$ is of the form:
$$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$
Recover the plaintext mesage $\overline{\mathsf{m}} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \widehat{\mathsf{ctx}}_0)$ and provide to the challenger in the blinding security game the message $\overline{\mathsf{m}}$. Next, sample a uniform permutation $\pi : [2n] \to [2n]$ and provide to the challenger the pair of permutations
$$(T, T') = (\pi, \pi_{\mathsf{id}}).$$
Receive from the challenger in the blinding security game a tuple of the form $(\overline{\mathsf{pk}}^*, \overline{\mathsf{ctx}}^*)$, and set
$$\overline{\mathsf{pk}}_0 = \overline{\mathsf{pk}}^*, \quad \overline{\mathsf{ctx}}_0 = \overline{\mathsf{ctx}}^*.$$
Set $(\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1)$ as in hybrid $\mathsf{Hyb}_{0,q-1}$, and output the re-encrypted ciphertext as:
$$\overline{\mathsf{ct}}_j = (1, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), (\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1).$$
- Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):
$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$
Recover the intermediate secret key $\overline{\mathsf{sk}}_{t-1} = \mathsf{KPHE.Dec}(\mathsf{sk}_i, \overline{\mathsf{ctx}}_t)$ and provide to the challenger in the blinding security game the message $\overline{\mathsf{sk}}_{t-1}$. Next, sample a uniform permutation $\pi_t : [2n] \to [2n]$ and provide to the challenger the pair of permutations
$$(T, T') = (\pi_t, \pi_{\mathsf{id}}).$$
Receive from the challenger in the blinding security game a tuple of the form $(\overline{\mathsf{pk}}^*, \overline{\mathsf{ctx}}^*)$, and set
$$\overline{\mathsf{pk}}_t = \overline{\mathsf{pk}}^*, \quad \overline{\mathsf{ctx}}_t = \overline{\mathsf{ctx}}^*.$$
Set $(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})$ as in hybrid $\mathsf{Hyb}_{0,q-1}$, and output the re-encrypted ciphertext as:
$$\overline{\mathsf{ct}}_j = ((t+1), (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

- Eventually the adversary $\mathcal{A}$ outputs a bit $b'$. $\mathcal{B}$ outputs the same bit $b'$.

It immediately follows $\mathcal{B}$ has the same advantage $\epsilon$ as $\mathcal{A}$ in breaking the blinding security of the KPHE scheme whenever $i = i'$ (i.e., when $\mathcal{B}$ does not abort the simulation). Hence, the overall advantage of $\mathcal{B}$ in breaking the blinding security of the KPHE scheme is

$$\epsilon' \geq \epsilon \cdot \Pr[i = i'] \geq \epsilon/n,$$

which is non-negligible whenever $\epsilon$ is non-negligible, for any $n = \text{poly}(\lambda)$. This leads to a contradiction, and concludes the proof of $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}'_{0,q-1}$ for all $q \in [Q]$.

□

$\mathsf{Hyb}'_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$. We now argue that $\mathsf{Hyb}'_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$.

*Proof.* Suppose that there exists some PPT adversary $\mathcal{A}$ that can efficiently distinguish $\mathsf{Hyb}'_{0,q-1}$ and $\mathsf{Hyb}_{0,q}$ with non-negligible advantage $\epsilon$. We construct a PPT algorithm $\mathcal{B}$ that breaks the blinding security of the KPHE scheme with non-negligible advantage $\epsilon'$. The algorithm $\mathcal{B}$ receives from the challenger in the blinding security game the public parameters $\mathsf{pp}$, a secret key $\mathsf{sk}^*$, and a public key $\mathsf{pk}^*$, and proceeds as follows.

- $\mathcal{B}$ uses $\mathsf{pp}$ to set up the PRE keys (both honest and corrupt) to be provided to the adversary $\mathcal{A}$ exactly as in $\mathsf{Hyb}_4$ except that it sets $\mathsf{sk}_j := \mathsf{sk}^*$ and $\mathsf{pk}_j := \mathsf{pk}^*$ for some uniform $j \xleftarrow{\$} \mathcal{K}_{\texttt{Honest}}$. Since $\mathsf{pk}^*$ is uniformly random, the view of the adversary $\mathcal{A}$ with respect to the distribution of the keys (both honest and corrupt) remains identical.

- Suppose that the $q$-th honest re-encryption query of the form $\mathsf{ReEnc}(i, j', \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$. If $j \neq j'$, $\mathcal{B}$ outputs $\perp$. Otherwise, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

- Suppose $\overline{\mathsf{ct}}_i$ is of the form:
$$\overline{\mathsf{ct}}_i = (0, (\mathsf{pk}_i, \widehat{\mathsf{ctx}}_0)).$$
Provide to the challenger in the blinding security game the message $\mathsf{sk}_i$. Next, sample a uniform permutation $\pi : [2n] \to [2n]$ and provide to the challenger the pair of permutations
$$(T, T') = (\pi_{\mathsf{id}}, \pi).$$
Receive from the challenger in the blinding security game a tuple of the form $(\overline{\mathsf{pk}}^*, \overline{\mathsf{ctx}}^*)$. Set $(\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0)$ as in hybrid $\mathsf{Hyb}_{0,q}$, and
$$\widehat{\mathsf{pk}}_1 = \overline{\mathsf{pk}}^*, \quad \widehat{\mathsf{ctx}}_1 = \overline{\mathsf{ctx}}^*.$$
Output the re-encrypted ciphertext as:
$$\overline{\mathsf{ct}}_j = (1, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), (\widehat{\mathsf{pk}}_1, \widehat{\mathsf{ctx}}_1)).$$

- Alternatively, suppose $\overline{\mathsf{ct}}_i$ is of the form (for some $t > 0$):

$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$

  Provide to the challenger in the blinding security game the message $\mathsf{sk}_i$. Next, sample a uniform permutation $\pi_t : [2n] \to [2n]$ and provide to the challenger the pair of permutations

$$(T, T') = (\pi_{\mathsf{id}}, \pi_t).$$

  Receive from the challenger in the blinding security game a tuple of the form $(\overline{\mathsf{pk}}^*, \overline{\mathsf{ctx}}^*)$. Set $(\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t)$ as in hybrid $\mathsf{Hyb}_{0,q}$, and

$$\widehat{\mathsf{pk}}_{t+1} = \overline{\mathsf{pk}}^*, \quad \widehat{\mathsf{ctx}}_{t+1} = \overline{\mathsf{ctx}}^*.$$

  Output the re-encrypted ciphertext as:

$$\overline{\mathsf{ct}}_j = ((t+1), (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

- Eventually the adversary $\mathcal{A}$ outputs a bit $b'$. $\mathcal{B}$ outputs the same bit $b'$.

It immediately follows $\mathcal{B}$ has the same advantage $\epsilon$ as $\mathcal{A}$ in breaking the blinding security of the KPHE scheme whenever $j = j'$ (i.e., when $\mathcal{B}$ does not abort the simulation). Hence, the overall advantage of $\mathcal{B}$ in breaking the blinding security of the KPHE scheme is

$$\epsilon' \geq \epsilon \cdot \Pr[j = j'] \geq \epsilon / n,$$

which is non-negligible whenever $\epsilon$ is non-negligible, for any $n = \text{poly}(\lambda)$. This leads to a contradiction, and concludes the proof of $\mathsf{Hyb}'_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$.

□

Putting these together, we have $\mathsf{Hyb}_{0,q-1} \overset{c}{\approx} \mathsf{Hyb}_{0,q}$ for all $q \in [Q]$. Finally, a simple hybrid argument over each $q \in [Q]$ completes the proof of Lemma C.4.

## C.2 Proof of Lemma C.5

We now present the detailed proof of Lemma C.5.

*Proof.* Suppose that there exists some PPT adversary $\mathcal{A}$ that can efficiently distinguish $\mathsf{Hyb}_3$ and $\mathsf{Hyb}_4$ in the proof of Theorem 4.6 with non-negligible advantage $\epsilon$. We construct a PPT algorithm $\mathcal{B}$ that breaks the distributional circular security of the KPHE scheme with non-negligible advantage $\epsilon'$. The algorithm $\mathcal{B}$ receives from the challenger in the distributional circular security game the public parameters $\mathsf{pp}$ proceeds as follows.

- Suppose that the adversary $\mathcal{A}$ outputs the honest and corrupt sets $\mathcal{K}_{\mathtt{Honest}}$ and $\mathcal{K}_{\mathtt{Corrupt}}$ of its choice. $\mathcal{B}$ provides to the challenger in the distributional circular security game with $n' = |\mathcal{K}_{\mathtt{Honest}}|$, and receives the set of KPHE public keys, say, $\{\mathsf{pk}_j\}_{j \in \mathcal{K}_{\mathtt{Honest}}}$.
- In its simulation, $\mathcal{B}$ uses the KPHE public keys $\{\mathsf{pk}_j\}_{j \in \mathcal{K}_{\mathtt{Honest}}}$ to simulate the honest PRE keys to be provided to the adversary $\mathcal{A}$. It additionally samples the corrupt keys to be provided to the adversary $\mathcal{A}$ on its own. Note that the view of the adversary $\mathcal{A}$ with respect to the distribution of the keys (both honest and corrupt) remains identical to that in $\mathsf{Hyb}_3$.

- $\mathcal{B}$ simulates the challenger in answering all re-encryption queries (on honestly generated ciphertexts) issued by $\mathcal{A}$ exactly as in $\mathsf{Hyb}_3$. Note that $\mathcal{B}$ can simulate the challenger in $\mathsf{Hyb}_3$ perfectly in answering such re-encryption queries, even without the knowledge of the secret keys corresponding to the honest public keys, since the challenger in $\mathsf{Hyb}_3$ does not require any knowledge of the honest secret keys to answer re-encryption queries on honestly generated ciphertexts.

- $\mathcal{B}$ receives an ensemble of ciphertexts $\{\mathsf{ct}^*_{i,j}\}_{i,j\in\mathcal{K}_{\texttt{Honest}}}$ from the challenger in the distributional circular security game. Upon receipt of a query from $\mathcal{A}$ of the form $\mathsf{ReKeyGen}(i,j)$ such that $i \in \mathcal{K}_{\texttt{Honest}}$ and $j \in \mathcal{K}_{\texttt{Honest}}$, $\mathcal{B}$ responds with $\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{ct}^*_{j,i})$.

- $\mathcal{B}$ simulates the challenger in answering the challenge encryption query issued by $\mathcal{A}$ exactly as in $\mathsf{Hyb}_3$. Note that $\mathcal{B}$ can simulate the challenger in $\mathsf{Hyb}_3$ perfectly in answering the challenge encryption query, even without the knowledge of the secret keys corresponding to the honest public keys, since the challenger in $\mathsf{Hyb}_3$ does not require any knowledge of the honest secret keys to answer the challenge encryption query.

- Eventually the adversary $\mathcal{A}$ outputs a bit $b'$. $\mathcal{B}$ outputs the same bit $b'$.

It immediately follows $\mathcal{B}$ has the same advantage $\epsilon$ as $\mathcal{A}$ in breaking the distributional circular security of the KPHE scheme. This leads to a contradiction, and concludes the proof of Lemma C.5. □

## C.3 Proof of Lemma C.6

Finally, we present the detailed proof of Lemma C.6.

*Proof.* Suppose that there exists some PPT adversary $\mathcal{A}$ that can efficiently distinguish $\mathsf{Hyb}_4$ and $\mathsf{Hyb}_5$ in the proof of Theorem 4.6 with non-negligible advantage $\epsilon$. We construct a PPT algorithm $\mathcal{B}$ that breaks the distributional semantic security of the KPHE scheme with non-negligible advantage $\epsilon'$. The algorithm $\mathcal{B}$ receives from the challenger in the distributional semantic security game the public parameters $\mathsf{pp}$ and a public key $\mathsf{pk}^*$, and proceeds as follows.

- $\mathcal{B}$ uses $\mathsf{pp}$ to set up the PRE keys (both honest and corrupt) to be provided to the adversary $\mathcal{A}$ exactly as in $\mathsf{Hyb}_4$ except that it sets $\mathsf{pk}_{i^*} := \mathsf{pk}^*$ for some uniform $i^* \xleftarrow{\$} \mathcal{K}_{\texttt{Honest}}$. Since $\mathsf{pk}^*$ is uniformly random, the view of the adversary $\mathcal{A}$ with respect to the distribution of the keys (both honest and corrupt) remains identical.

- $\mathcal{B}$ then proceeds to simulate the challenger exactly in $\mathsf{Hyb}_4$. Note that $\mathcal{B}$ can simulate the challenger in $\mathsf{Hyb}_4$ perfectly, without the knowledge of the secret key $\mathsf{sk}^*$ corresponding to $\mathsf{pk}^*$, since the challenger in $\mathsf{Hyb}_4$ does not require any knowledge of the honest secret keys.

- Suppose that in the challenge phase of hybrid $\mathsf{Hyb}_4$, the adversary issues a challenge query of the form $\mathcal{O}.\mathsf{chall}(i, \overline{\mathsf{m}}_0, \overline{\mathsf{m}}_1)$. If $i \neq i^*$, $\mathcal{B}$ outputs $\perp$ and aborts.

- Otherwise, $\mathcal{B}$ samples $b \in \{0,1\}$ and outputs the challenge messages $(\overline{\mathsf{m}}_b, 0^{2n})$ to the challenger in the distributional semantic security game. Upon receipt of the challenge ciphertext $\widehat{\mathsf{ct}}^*$, $\mathcal{B}$ outputs $\overline{\mathsf{ct}}^* = (0, (\mathsf{pk}^*, \widehat{\mathsf{ct}}^*))$ to the adversary $\mathcal{A}$.

- Eventually the adversary $\mathcal{A}$ outputs a bit $b'$. $\mathcal{B}$ outputs the same bit $b'$.

It immediately follows when $i = i^*$ (i.e., when $\mathcal{B}$ does not abort the simulation), $\mathcal{B}$ has the same advantage $\epsilon$ as $\mathcal{A}$ in breaking the distributional semantic security of the KPHE scheme. Hence, the overall advantage of $\mathcal{B}$ in breaking the distributional semantic security of the KPHE scheme is

$$\epsilon' \geq \epsilon \cdot \Pr[i = i^*] \geq \epsilon / n,$$

which is non-negligible whenever $\epsilon$ is non-negligible, for any $n = \text{poly}(\lambda)$. This leads to a contradiction, and concludes the proof of Lemma C.6. □

# D Proof of IND-PCS Security (Theorem 4.8)

In this section, we prove Theorem 4.8, which establishes the IND-PCS security for our PRE construction in Section 4.3.

*Proof.* The proof follows a sequence of hybrids as outlined below. As before, we use $[N]$ to denote the set of all keys (honest + corrupt), $\mathcal{K}_{\texttt{Honest}} \subset [N]$ to denote the set of honest keys, and $\mathcal{K}_{\texttt{Corrupt}} \subset [N]$ to denote the set of corrupt keys. As mentioned earlier, these sets are fixed (adversarially) at the beginning of the game, before the adversary is allowed to issue re-key generation queries (this remains unchanged in each hybrid). We additionally let $\mathcal{T}$ denote the set of all re-encryption keys received by the adversary in response to the (valid) $\mathsf{ReKeyGen}$ queries issued by it during the game (where these queries can be issued adaptively).

**$\mathsf{Hyb}_0$:** This hybrid is identical to the real IND-PCS security game between the challenger and the adversary.

**$\mathsf{Hyb}_1$:** This hybrid is identical to the hybrid $\mathsf{Hyb}_0$ except that the challenger locally maintains an additional table $T$ (initially empty), and does the following:

- For each honest encryption query of the form $\mathsf{Enc}(i, \overline{\mathsf{m}})$, the challenger adds to the local table $T$ an entry of the form $(i, \overline{\mathsf{ct}}, \overline{\mathsf{m}})$, where $\overline{\mathsf{ct}}$ is generated as $\overline{\mathsf{ct}} = \mathsf{KPHE.Enc}(\mathsf{pk}_i, \overline{\mathsf{m}})$.
- For each honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

  Suppose $\overline{\mathsf{ct}}_i$ is of the form:

  $$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$

  for some $t \geq 0$. The challenger recovers the following (in order):

  $$\begin{aligned} \overline{\mathsf{sk}}_{t-1} &= \mathsf{KPHE.Dec}(\mathsf{sk}_i, \widehat{\mathsf{ctx}}_t), \\ \overline{\mathsf{sk}}_{t-2} &= \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), \\ &\vdots \\ \overline{\mathsf{sk}}_0 &= \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_1, \overline{\mathsf{ctx}}_1), \\ \overline{\mathsf{m}} &= \mathsf{KPHE.Dec}(\overline{\mathsf{sk}}_0, \overline{\mathsf{ctx}}_0). \end{aligned}$$

The challenger samples uniform permutations $\pi_0, \ldots, \pi_t : [2n] \to [2n]$ and sets the following:

$$\widetilde{\mathsf{sk}}_\ell = \pi_\ell(\overline{\mathsf{sk}}_\ell), \quad \widetilde{\mathsf{pk}}_\ell \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_\ell) \text{ for each } \ell \in [0, t-1],$$

$$\overline{\mathsf{sk}}_t = \pi_t(\mathsf{sk}_i), \quad \overline{\mathsf{pk}}_t \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

Also set

$$\overline{\mathsf{ctx}}_t \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_t, \overline{\mathsf{sk}}_{t-1}), \quad \overline{\mathsf{ctx}}_{t+1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, \overline{\mathsf{sk}}_t),$$
$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \overline{\mathsf{ctx}}_{t+1}, \pi_{\mathsf{id}}, \pi_{\mathsf{id}}),$$

where $\pi_{\mathsf{id}} : [2n] \to [2n]$ is the identity permutation. Finally, the challenger sets the following:

$$\widetilde{\mathsf{ctx}}_0 \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_0, \overline{\mathsf{m}}),$$
$$\widetilde{\mathsf{ctx}}_\ell \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_\ell, \widetilde{\mathsf{sk}}_{\ell-1}) \text{ for each } \ell \in [1, t-1].$$

Output the re-encrypted ciphertext as:

$$\overline{\mathsf{ct}}_j = ((t+1), (\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0), \ldots, (\widetilde{\mathsf{pk}}_{t-1}, \widetilde{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

Also, add to the local table $T$ an entry of the form $(j, \overline{\mathsf{ct}}_j, \widetilde{\mathsf{sk}}_0, \ldots, \widetilde{\mathsf{sk}}_{t-1}, \overline{\mathsf{sk}}_t)$.

The view of the IND-PCS adversary in this hybrid is computationally indistinguishable from that in the hybrid $\mathsf{Hyb}_0$ under the assumption the KPHE scheme satisfies public key and ciphertext blinding. The proof of this indistinguishability claim follows from a hybrid argument very similar to that used in the proof of Lemma C.4, and is hence not detailed separately.

**$\mathsf{Hyb}_2$:** This hybrid is identical to the hybrid $\mathsf{Hyb}_1$ except that the challenger does the following (it still locally maintains the table $T$ as in $\mathsf{Hyb}_1$): for each honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

Suppose $\overline{\mathsf{ct}}_i$ is of the form:

$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)).$$

for some $t \geq 0$. The challenger looks up the local table $T$ for an entry of the form

$$(i, \overline{\mathsf{ct}}_i, \overline{\mathsf{m}}, \overline{\mathsf{sk}}_0, \ldots, \overline{\mathsf{sk}}_{t-1}),$$

and uses these secret keys for the simulation instead of decrypting and recovering them as in hybrid $\mathsf{Hyb}_1$. Such an entry is guaranteed to exist. The rest of the simulation proceeds exactly as in hybrid $\mathsf{Hyb}_1$.

It is easy to see that view of the IND-PCS adversary in this hybrid is identical to that in the hybrid $\mathsf{Hyb}_1$.

$\mathsf{Hyb}_3$: This hybrid is identical to the hybrid $\mathsf{Hyb}_2$ except that the challenger does the following (it still locally maintains the table $T$ as in $\mathsf{Hyb}_2$): for each honest re-encryption query of the form $\mathsf{ReEnc}(i, j, \overline{\mathsf{ct}}_i)$ to which the challenger does not respond with $\perp$, it fetches the entry $(i, \overline{\mathsf{ct}}_i)$ (from either $\mathcal{L}$ or $\widetilde{\mathcal{L}}$) and proceeds as follows.

Suppose $\overline{\mathsf{ct}}_i$ is of the form:

$$\overline{\mathsf{ct}}_i = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)),$$

for some $t \geq 0$. The challenger looks up the local table $T$ for an entry of the form

$$(i, \overline{\mathsf{ct}}_i, \overline{\mathsf{m}}, \overline{\mathsf{sk}}_0, \ldots, \overline{\mathsf{sk}}_{t-1}),$$

and uses these secret keys for the simulation instead of decrypting and recovering them as in hybrid $\mathsf{Hyb}_1$. Such an entry is guaranteed to exist. Next, the challenger samples the following:

$$\widetilde{\mathsf{sk}}_\ell \xleftarrow{\$} \mathcal{U}_n, \quad \widetilde{\mathsf{pk}}_\ell \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_\ell) \text{ for each } \ell \in [0, t-1].$$
$$\overline{\mathsf{sk}}_t \xleftarrow{\$} \mathcal{U}_n, \quad \overline{\mathsf{pk}}_t \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

The rest of the simulation proceeds exactly as in hybrid $\mathsf{Hyb}_2$.

It is again easy to see that the view of the IND-PCS adversary in this hybrid is identical to that in the hybrid $\mathsf{Hyb}_2$, since the distributions of the intermediate secret keys in the re-encrypted ciphertexts remain identical.

**Remark D.1.** *Note that in* $\mathsf{Hyb}_3$*, the set of secret keys* $\{\mathsf{sk}_i\}_{i \in [N]}$ *(including both corrupt and honest secret keys) is no longer used by the challenger when answering* $\mathsf{ReEnc}$ *queries. In particular, all* $\mathsf{ReEnc}$ *queries are answered using only the knowledge of the public keys* $\{\mathsf{pk}_i\}_{i \in [N]}$*, and the local table* $T$ *maintained by the challenger.*

$\mathsf{Hyb}_4$: This hybrid is identical to the hybrid $\mathsf{Hyb}_3$ except for the manner in which the challenger answers the $\mathsf{ReKeyGen}$ queries issued by the adversary. In particular, given a query of the form $\mathsf{ReKeyGen}(i, j)$ such that $i \in \mathcal{K}_{\mathtt{Honest}}$ and $j \in \mathcal{K}_{\mathtt{Honest}}$, the challenger responds with $\mathsf{rk}_{i,j} = (\mathsf{pk}_j, \mathsf{KPHE.Enc}(\mathsf{pk}_j, 0^{2n}))$.

We argue that $\mathsf{Hyb}_4$ is indistinguishable from $\mathsf{Hyb}_3$ in a straightforward manner under the assumption that $\mathsf{KPHE}$ satisfies distributional circular security. This argument is identical to the argument in the proof of Lemma C.5, and is hence not detailed.

**Remark D.2.** *Note that in* $\mathsf{Hyb}_4$*, the set of honest secret keys* $\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\mathtt{Honest}}}$ *is no longer used by the challenger when answering* $\mathsf{ReKeyGen}$ *queries. Also, as in* $\mathsf{Hyb}_3$*, the set of honest secret keys* $\{\mathsf{sk}_i\}_{i \in \mathcal{K}_{\mathtt{Honest}}}$ *are also not used by the challenger when answering* $\mathsf{ReEnc}$ *queries. In other words,* $\mathsf{Hyb}_4$ *allows the challenger to entirely "forget" the set of honest secret keys.*

$\mathsf{Hyb}_5$: This hybrid is identical to the hybrid $\mathsf{Hyb}_4$ except for the manner in which the challenger answers the challenge re-encryption query issued by the adversary. In particular, given a challenge query of the form $\mathcal{O}.\mathsf{chall\text{-}IND\text{-}PCS}(i, j, \overline{\mathsf{ct}}_0^*, \overline{\mathsf{ct}}_1^*)$, the challenger proceeds as follows:

- If $(i, \overline{\mathsf{ct}}_0) \notin \mathcal{L}$ or $(i, \overline{\mathsf{ct}}_1) \notin \mathcal{L}$, respond with $\overline{\mathsf{ct}}^* = \perp$ (this is exactly as in $\mathsf{Hyb}_4$).

- If $|\overline{\mathsf{ct}}_0^*| \neq |\overline{\mathsf{ct}}_1^*|$ or $j \in \mathcal{K}_{\mathtt{Corrupt}}$, respond with $\overline{\mathsf{ct}}^* = \perp$ (this is exactly as in $\mathsf{Hyb}_4$).
- Otherwise, proceed as follows: let $b$ be the choice bit of the challenger and suppose $\overline{\mathsf{ct}}_b^*$ is of the form:

$$\overline{\mathsf{ct}}_b^* = (t, (\overline{\mathsf{pk}}_0, \overline{\mathsf{ctx}}_0), \ldots, (\overline{\mathsf{pk}}_{t-1}, \overline{\mathsf{ctx}}_{t-1}), (\widehat{\mathsf{pk}}_t, \widehat{\mathsf{ctx}}_t)),$$

for some $t \geq 0$. Look up the local table $T$ for an entry of the form

$$(i, \overline{\mathsf{ct}}_b^*, \overline{\mathsf{m}}, \overline{\mathsf{sk}}_0, \ldots, \overline{\mathsf{sk}}_{t-1}).$$

Such an entry is guaranteed to exist. Next, sample uniform permutations $\pi_0, \ldots, \pi_t : [2n] \to [2n]$ and set the following:

$$\widetilde{\mathsf{sk}}_\ell = \pi_\ell(\overline{\mathsf{sk}}_\ell), \quad \widetilde{\mathsf{pk}}_\ell \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_\ell) \text{ for each } \ell \in [0, t-1],$$

$$\overline{\mathsf{sk}}_t = \pi_t(\mathsf{sk}_i), \quad \overline{\mathsf{pk}}_t \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

Also set

$$\overline{\mathsf{ctx}}_t \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_t, \overline{\mathsf{sk}}_{t-1}), \quad \overline{\mathsf{ctx}}_{t+1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, \overline{\mathsf{sk}}_t),$$

$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \overline{\mathsf{ctx}}_{t+1}, \pi_{\mathsf{id}}, \pi_{\mathsf{id}}),$$

where $\pi_{\mathsf{id}} : [2n] \to [2n]$ is the identity permutation. Finally, the challenger sets the following:

$$\widetilde{\mathsf{ctx}}_0 \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_0, \overline{\mathsf{m}}),$$

$$\widetilde{\mathsf{ctx}}_\ell \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_\ell, \widetilde{\mathsf{sk}}_{\ell-1}) \text{ for each } \ell \in [1, t-1].$$

Output the re-encrypted ciphertext as:

$$\overline{\mathsf{ct}}_j^* = ((t+1), (\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0), \ldots, (\widetilde{\mathsf{pk}}_{t-1}, \widetilde{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

Also, add to the local table $T_1$ an entry of the form $(j, \overline{\mathsf{ct}}_j^*, \widetilde{\mathsf{sk}}_0, \ldots, \widetilde{\mathsf{sk}}_{t-1}, \overline{\mathsf{sk}}_t)$.

We argue that the view of the adversary in $\mathsf{Hyb}_5$ is computationally indistinguishable from that in $\mathsf{Hyb}_4$ under the assumption the KPHE scheme satisfies public key and ciphertext blinding.

**$\mathsf{Hyb}_6$:** This hybrid is identical to the hybrid $\mathsf{Hyb}_5$ except for the manner in which the challenger answers the challenge re-encryption query issued by the adversary. In particular, when responding to the challenge query, the challenger sets:

$$\widetilde{\mathsf{sk}}_\ell \xleftarrow{\$} \mathcal{U}_n, \quad \widetilde{\mathsf{pk}}_\ell \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \widetilde{\mathsf{sk}}_\ell) \text{ for each } \ell \in [0, t-1].$$

$$\overline{\mathsf{sk}}_t \xleftarrow{\$} \mathcal{U}_n, \quad \overline{\mathsf{pk}}_t \xleftarrow{\$} \mathsf{KPHE.PKGen}(\mathsf{pp}, \overline{\mathsf{sk}}_t).$$

The rest of the simulation proceeds exactly as in hybrid $\mathsf{Hyb}_5$.

It is again easy to see that the view of the adversary in $\mathsf{Hyb}_6$ is identical to that in $\mathsf{Hyb}_5$, since the distributions of the intermediate secret keys in the challenge ciphertext do not change.

$\mathsf{Hyb}_7$**:** This hybrid is identical to the hybrid $\mathsf{Hyb}_6$ except for the manner in which the challenger answers the challenge re-encryption query issued by the adversary. In particular, when responding to the challenge query, the challenger sets:

$$\overline{\mathsf{ctx}}_t \xleftarrow{\$} \mathsf{KPHE.Enc}(\overline{\mathsf{pk}}_t, 0^{2n}), \quad \overline{\mathsf{ctx}}_{t+1} \xleftarrow{\$} \mathsf{KPHE.Enc}(\mathsf{pk}_j, 0^{2n}),$$

$$(\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1}) \xleftarrow{\$} \mathsf{KPHE.Eval}(\mathsf{pk}_j, \overline{\mathsf{ctx}}_{t+1}, \pi_{\mathsf{id}}, \pi_{\mathsf{id}}),$$

where $\pi_{\mathsf{id}} : [2n] \to [2n]$ is the identity permutation. Finally, the challenger sets the following:

$$\widetilde{\mathsf{ctx}}_0 \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_0, 0^{2n}),$$

$$\widetilde{\mathsf{ctx}}_\ell \xleftarrow{\$} \mathsf{KPHE.Enc}(\widetilde{\mathsf{pk}}_\ell, 0^{2n}) \text{ for each } \ell \in [1, t-1].$$

The rest of the simulation proceeds exactly as in hybrid $\mathsf{Hyb}_6$, i.e., the challenger outputs the re-encrypted ciphertext as:

$$\overline{\mathsf{ct}}_j^* = ((t+1), (\widetilde{\mathsf{pk}}_0, \widetilde{\mathsf{ctx}}_0), \ldots, (\widetilde{\mathsf{pk}}_{t-1}, \widetilde{\mathsf{ctx}}_{t-1}), (\overline{\mathsf{pk}}_t, \overline{\mathsf{ctx}}_t), (\widehat{\mathsf{pk}}_{t+1}, \widehat{\mathsf{ctx}}_{t+1})).$$

We argue that $\mathsf{Hyb}_7$ is indistinguishable from $\mathsf{Hyb}_6$ under the assumption that $\mathsf{KPHE}$ satisfies distributional semantic security. The argument follows via a hybrid argument over each of the intermediate secret keys in the challenge ciphertext in reverse order, i.e., starting with $\overline{\mathsf{sk}}_t$, and then in order from $\widetilde{\mathsf{sk}}_{t-1}$ through $\widetilde{\mathsf{sk}}_0$, where each hybrid argument is very similar to the argument used in the proof of Lemma C.6. The overall argument is also very similar to the proof of Lemma B.1 in the proof of IND-UPD security for our UE construction in Section 3.3.1. Hence, the argument is not detailed separately.

**Remark D.3.** *Note that in* $\mathsf{Hyb}_7$*, the challenge ciphertext* $\overline{\mathsf{ct}}^*$ *is independent of* $(\overline{\mathsf{ct}}_0^*, \overline{\mathsf{ct}}_1^*)$*, and hence the adversary's advantage in winning the IND-PCS game is zero.*

□