Updatable Public Key Encryption from DCR: Efficient Constructions With Stronger Security

Calvin Abou Haidar, Benoit Libert, Alain Passelègue

Updatable Public Key Encryption from DCR: Efficient Constructions With Stronger Security

Calvin Abou Haidar
ENS de Lyon, Laboratoire LIP
(U. Lyon, CNRS, ENSL, Inria, UCBL)
INRIA
France

Benoît Libert
ENS de Lyon, Laboratoire LIP
(U. Lyon, CNRS, ENSL, Inria, UCBL)
CNRS, Laboratoire LIP
France

Alain Passelègue
ENS de Lyon, Laboratoire LIP
(U. Lyon, CNRS, ENSL, Inria, UCBL)
INRIA
France

ABSTRACT

Forward-secure encryption (FS-PKE) is a key-evolving public-key paradigm that preserves the confidentiality of past encryptions in case of key exposure. Updatable public-key encryption (UPKE) is a natural relaxation of FS-PKE, introduced by Jost *et al.* (Eurocrypt'19), which is motivated by applications to secure messaging. In UPKE, key updates can be triggered by any sender – via special update ciphertexts – willing to enforce the forward secrecy of its encrypted messages. So far, the only truly efficient UPKE candidates (which rely on the random oracle idealization) only provide rather weak security guarantees against passive adversaries as they are malleable. Also, they offer no protection against malicious senders willing to hinder the decryption capability of honest users. A recent work of Dodis *et al.* (TCC'21) described UPKE systems in the standard model that also hedge against maliciously generated update messages in the chosen-ciphertext setting (where adversaries are equipped with a decryption oracle). While important feasibility results, their constructions lag behind random-oracle candidates in terms of efficiency. In this paper, we first provide a drastically more efficient UPKE realization in the standard model using Paillier's Composite Residuosity (DCR) assumption. In the random oracle model, we then extend our initial scheme so as to achieve chosen-ciphertext security, even in a model that accounts for maliciously generated update ciphertexts. Under the DCR and Strong RSA assumptions, we thus obtain the first practical UPKE systems that satisfy the strongest security notions put forth by Dodis *et al.*

CCS CONCEPTS

• **Security and privacy → Public key encryption**.

KEYWORDS

DCR, Paillier, Updatable Public Key Encryption, Forward Security

1 INTRODUCTION

Secure communication aims to guarantee the confidentiality of communication over an insecure channel. In some cases involving long-lasting communication between multiple parties, e.g. in the context of secure group messaging applications, a desirable feature is to guarantee the confidentiality of past communications after a user gets compromised. This can be achieved by having users refresh the communication keys regularly: if an attacker compromises a user, messages sent before the last refresh remain confidential. Rather than resampling fresh keys, one would like an efficient procedure to update user keys while preserving functionality and still guaranteeing confidentiality of pre-update messages.

Forward security [2, 4, 11, 15, 18, 33, 36] was suggested to address this problem: it makes possible to update communication keys while preserving functionality, and guarantees that exposure of the secret key of a user does not compromise past communications. In the symmetric setting [5], it can be achieved via simple solutions: e.g., by using a length-doubling PRG and a shared initial seed s_0 to iteratively generate pairs $G(s_{i-1}) = (k_i, s_i)$ with k_i being a one-time key and s_i the seed for the next update. Yet, in the context of secure group messaging, it should be possible to add (or remove) users to the group at any time without having to engage in a large-scale interactive key-agreement protocol that could take long to run, especially if users are often offline. Hence, one wishes to rely on key-evolving public-key encryption.

Unfortunately, forward security in this setting [15] has proven to be much harder to achieve than in the symmetric setting. Again, having all users refresh their public/secret key pair trivially solves the problem, but it requires every user to be active (and online) between each update. In the context of secure group messaging, some users might have lazy behaviors (e.g., read-only or mostly offline users) and might be incapable of refreshing their key pairs. For these users, key exposures can be devastating and ruin the confidentiality of all prior messages. Forward-secure PKE (FS-PKE) addresses this issue: the initial pair of public/secret key (pk_0, sk_0) allows (independently) deriving a chain of pairs of public/secret keys (pk_i, sk_i). Senders can derive a public key pk_i from pk_{i-1} every time they send a message without needing the recipient to be active, and security of this message is later guaranteed by the one-wayness of the key chain: exposure of sk_i does not compromise the ciphertexts encrypted under pk_j for $j < i$. In some sense, this is a relaxed version of Hierarchical Identity-Based Encryption (HIBE) [28, 32], as the chain of secret keys forms a hierarchy that is easy to explore in only one direction. The seminal paper by Canetti *et al.* [15] showed that HIBE implies an efficient[1] form of FS-PKE where all public keys are fixed (i.e., $pk_i = pk_0$ for all i) and only secret keys are updated. In the last 2 decades, many constructions of HIBE were proposed from a variety of assumptions [7, 8, 13, 16, 22], leading to as many instantiations of FS-PKE. Still, if we want to rely on number theoretic foundations outside the realm of pairing-based cryptography, known realizations of HIBE are still far from reaching the efficiency of simple PKE schemes. In particular, we do not have practical candidates based on long-lived assumption like the Decision Diffie-Hellman assumption or factoring-related assumptions. In fact, achieving HIBE from these assumptions inherently [10, 41] requires (usually inefficient) non-black-box techniques.

[1] By "efficient", we mean that the complexity in all metrics (in particular, key sizes) is at most poly-logarithmic in the number of time periods.

To circumvent the aforementioned barriers while still guaranteeing forward-security, an intermediate notion, termed *updatable PKE* (UPKE), was recently proposed by Jost *et al.* [34] and Alwen *et al.* [1]. This notion was notably motivated by secure group messaging applications. The main difference between UPKE and FS-PKE is that, in the former, new updated keys can be derived at any point by any sender. Specifically, any sender can update the public/secret keys (pk, sk) of a target recipient by computing a public update consisting of a pair (pk', ct'), where pk' is the updated public key of the target recipient and ct' encrypts under pk the update to perform on sk in order to obtain sk'.

Typically, the secret key lives in an additive group $(\mathbb{G}, +)$ and the corresponding public key is derived from sk by a public (one-way) group homomorphism $f : (\mathbb{G}, +) \to (\mathbb{H}, *)$ as $pk = f(sk)$. Updating the key can then be performed by sampling a random element r in $\mathbb{G}$ and revealing $pk' = pk * f(r)$ and an encryption $\mathsf{Enc}(pk, r)$. Forward security is then guaranteed by the fact that f is hard to invert: given a compromised (freshly updated) secret key $sk + r$ and $f(r)$, it is hard to recover sk (as long as r was sampled from a distribution on which f is one-way). Notably, any honest user can produce a safe update that will guarantee the confidentiality of all past messages. For instance, it is possible to update the key of a recipient immediately after having sent a message to it, so that the message remains confidential in case the recipient's secret key gets compromised in the future. In this paper, we provide new *practical* realizations of UPKE schemes that rely on standard assumptions and satisfy the strongest security notions in the literature.

Prior Works. A UPKE scheme is a standard PKE scheme (KGen, Enc, Dec) enhanced with two additional algorithms (UpdatePk, UpdateSk). The UpdatePk algorithm can be run by any sender on the current public key pk_{i-1} to produce a new public key pk_i as well as an update ciphertext up_i. This update ciphertext up_i can then be used by the recipient, together with the current secret key sk_{i-1}, to run the UpdateSk algorithm and obtain the new secret key sk_i corresponding to pk_i. The main security notion that has been considered so far [1, 21, 34] is indistinguishability under chosen-plaintext attacks (IND-CPA). Security notions slightly differ in prior works and we adopt the more general and recent formalization of Dodis *et al.* [21] for the notions of indistinguishability under chosen-randomness chosen-plaintext attacks (IND-CR-CPA). The latter guarantees that, even if an attacker has some control on the sequence of updates by choosing (potentially bad) randomness for these updates, a single honest and non-compromised update (that is, an honest update for fresh, high-entropy randomness) suffices to guarantee that exposure of the secret key does not break the IND-CPA security of ciphertexts encrypted before the last non-compromised update.

Three constructions of IND-CR-CPA secure UPKE are known so far. A first one, based on the CDH assumption in the random oracle model, was proposed in [1, 34]. It simply consists of a hashed Elgamal encryption and is extremely efficient, but it crucially relies on the random oracle to circumvent a circular-security issue [6]. In the standard model, we are only aware of two constructions due to Dodis *et al.* [21]. They achieve IND-CR-CPA security in the standard model under the DDH assumption and under the LWE assumption [43] (with super-polynomial modulus-to-noise rate). The two constructions follow a similar approach: to achieve IND-CR-CPA secure under the DDH (resp. LWE) assumption, they start by constructing a circular-secure, leakage-resilient PKE scheme obtained by tweaking the BHHO [9] (resp. Dual-Regev [27, 43]) cryptosystem. These PKE schemes use a binary secret key $s \in \{0,1\}^\ell$, and can retain IND-CPA security even if the adversary is provided with $s + r \in \{0,1,2\}^\ell$ for a uniformly random $r \in \{0,1\}^\ell$, as well as a (bit-by-bit) encryption of r.[2] This property is dubbed CS+LR security [21], where CS (resp. LR) stands for Circular Security (resp. Leakage-Resilience). These schemes further benefit from a homomorphic key structure as defined above: in the BHHO scheme [9], we have $pk = \prod_{i\in[\ell]} g_i^{s_i}$ for some public group elements g_i while, in the Dual-Regev scheme [27], $pk = \mathbf{A} \cdot s$ for a public matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times\ell}$.

These two schemes were shown [21] amenable to constructing UPKE schemes, where updating simply consists in sampling a random $r \in \{0,1\}^\ell$, encrypting it (bit-by-bit) and using the homomorphic key structure to update pk. IND-CR-CPA security is then achievable from CS+LR security by leveraging this homomorphic key structure. If the adversary does not make a single update, IND-CR-CPA and CS+LR security are actually equivalent: r plays the role of the non-compromised, honest update, while $s + r$ is the resulting secret key. Potential prior updates made by the adversary can be dealt with in this reduction using the homomorphic properties of the schemes.[3]

For the time being, IND-CR-CPA security is guaranteed by a simple and efficient construction under CDH in the ROM and two constructions under DDH and LWE in the standard model. However, the standard model candidates are rather inefficient due to the bit-by-bit encryption procedure of update messages. In the LWE case, the scheme of [21] further relies on noise flooding, leading to a super-polynomial modulus-to-noise rate which eventually leads to very large keys/ciphertexts.

IND-CR-CPA security is a fairly weak notion and presents two noticeable weaknesses. The first one is well-known: it lacks security against adversaries equipped with a decryption oracle. To circumvent this issue, Dodis *et al.* [21] defined IND-CR-CCA security, in which the attacker is allowed to make decryption queries. The second weakness is that nothing prevents a malicious sender from generating updates in which the new public key and the update ciphertext are unrelated. By doing so, a user could maliciously turn a public key into a new public key for which the recipient does no longer have a a secret key, thus preventing the targeted user from decrypting until it publicizes a fresh public key. To prevent this, it is desirable to preserve security even if the adversary can submit malicious updates (and not just malicious randomness for the update as in the chosen-randomness setting). This notion was termed indistinguishability under chosen-update, chosen-ciphertext attacks (IND-CU-CCA) in [21]. Generic transformations from IND-CR-CPA to IND-CR-CCA security and from IND-CR-CCA to IND-CU-CCA security have been proposed in

[2] In [21], the authors actually reveal a bit-by-bit encryption of sk rather than r. We use this variant here and in the paper as its makes the argument for proving IND-CR-CPA easier.

[3] In [21], since CS+LR security reveals an encryption of s rather than r, generating the update ciphertext requires to transform the encryption of s into an encryption of r by exploiting the additional homomorphic properties of the scheme.

[21]. Yet, these transformations are rather theoretical and rely on one-time, strong, true-simulation f-extractable Non-Interactive Zero-Knowledge (NIZK) arguments [20]. While such arguments are instantiable from Groth-Sahai proofs [30], it is not clear how to instantiate them in a truly practical way, let alone from assumptions that do not already imply HIBE. This leaves open the question of designing practical IND-CR-CCA or IND-CU-CCA UPKE systems.

Our Contributions. We propose new and efficient UPKE constructions based on standard assumptions. Our contribution is 4-fold. First, we provide the first IND-CR-CPA UPKE construction relying on the Decision Composite Residuosity (DCR) assumption [40] in the standard model. Our scheme features much shorter ciphertexts than the previous standard-model realizations [21]. In particular, we avoid the expensive bit-wise encryption step in the update mechanism. We then extend our construction to obtain the first practical IND-CR-CCA UPKE construction, under the DCR assumption, in the random oracle model. As a third contribution, we further extend our system to obtain the first practical IND-CU-CCA UPKE construction, which is proven secure under the DCR assumption and the Strong RSA assumption [3] in the ROM. Finally, we provide a proof-of-concept implementation of our constructions to provide evidence of the efficiency of our approach.

1.1 Technical Overview.

The starting point of our constructions is the Elgamal-Paillier PKE scheme from [14]. We consider an RSA modulus $N = PQ$ for safe-primes $P = 2p + 1$ and $Q = 2q + 1$. Like Damgård and Jurik [19], we work over $\mathbb{Z}_{N^{\zeta+1}}$ for $\zeta \geq 1$ (in our constructions, we take $\zeta = 1$ or $\zeta = 2$) and provide a public generator g of the subgroup of (unknown) order pq. In this subgroup, computing discrete logarithms is hard, while it is easy in the subgroup of order N^{ζ} generated by $T = (1 + N)$. A secret key is a random element $x \hookleftarrow \mathcal{U}([0, (N-1)/4])$ and the corresponding public key is $h = g^x$. To encrypt $m \in \mathbb{Z}_{N^{\zeta}}$, one samples $t \hookleftarrow \mathcal{U}([0, (N-1)/4])$ and outputs $(c_0, c_1) = (g^t \bmod N^{\zeta+1}, h^t \cdot T^m \bmod N^{\zeta+1})$. Decryption is done by computing the discrete logarithm of $z = c_1 \cdot (c_0)^{-x} \bmod N^{\zeta+1}$, which is efficient as z lives in the subgroup generated by T. For $\zeta = 1$, one can simply compute $\frac{z-1 \bmod N^2}{N}$ and, for $\zeta \geq 2$, a recursive polynomial-time algorithm is given in [19].

Our DCR-Based IND-CR-CPA Construction in the Standard Model. We show that this PKE construction can be turned into an efficient IND-CR-CPA UPKE construction. To update a public key pk, one can simply sample randomness $r \hookleftarrow \mathcal{U}([0, (N-1)/4])$ and encrypts r under pk while updating pk to $pk \cdot g^r$. Note that a nice feature of the scheme is that the entire update information r can be encrypted in a single ciphertext, in contrast with previous UPKE schemes that were encrypting updates bit-by-bit. To argue about security, we follow the same path as above (which is a variant of the one from [21]), and show that the above construction achieves CS+LR security, where the adversary against the IND-CPA security of the scheme additionally gets $sk + r$ (over $\mathbb{Z}$) as well as an encryption $\mathsf{Enc}(pk, r)$. To prove CS+LR security, we first use the algebraic structure and knowledge of $sk + r$ to transform the secret key underlying the distribution given to the adversary from sk to $-r$. This is done by computing a new public key $g^{-r} = pk \cdot g^{-sk-r}$ and by multiplying the c_1 component of ciphertexts $(c_0, c_1) = (g^t, pk^t \cdot T^m)$ by $(g^t)^{-sk-r}$. By doing so, we remove all information about sk except $sk + r$, and now use $-r$ as secret key. Assuming that sk is (exponentially) larger than r, a flooding argument shows that $sk + r$ over $\mathbb{Z}$ does not reveal any information about r. We are left with showing a standard form of circular-security [6], which is doable [35, 37] from the DCR assumption. This yields a DCR-based CS+LR PKE system based on the Elgamal-Paillier PKE, with the difference that sk is now sampled from $\mathcal{U}([0, 2^{\lambda}(N-1)/4])$ in order to apply the flooding argument.[4] Using the CS+LR to IND-CR-CPA argument described above, we then obtain the first DCR-based IND-CR-CPA UPKE scheme in the standard model. Our construction significantly outperforms prior constructions in the standard model (as evidenced by efficiency comparisons in Section 5.1) by encrypting the whole update information in one shot, without relying on bit-by-bit encryption.

From IND-CR-CPA to IND-CR-CCA Security. As a second step, we aim to upgrade our construction so as to achieve IND-CR-CCA security. To this end, we exploit the algebraic structure of ciphertexts that makes it compatible with efficient zero-knowledge proofs (a property which is lacking in the efficient hashed Elgamal constructions $(g^r, M \oplus H(h^r))$ of [1, 34], where the hash function hinders the use of zero-knowledge proofs). We apply the standard Naor-Yung –double encrypt + NIZK– paradigm [39] to our construction and show that it leads to IND-CR-CCA security. In more details, we modify our construction as follows: we add a second random generator h_d of the subgroup of order pq in the public parameters. KGen remains unchanged, but encrypting a message m to a recipient with public key pk now leads to a ciphertext (ct, ct_d, π), where ct encrypts m under pk; ct_d encrypts m under h_d; and π is a NIZK proof that plaintexts underlying ct, ct_d are equal. UpdatePk, UpdateSk algorithms are unchanged. Note, in particular, that the second key h_d is part of the public parameters and is never updated by the senders. In the proof, the second public key h_d can serve as a backdoor to answer decryption queries and simulation-soundness [44] guarantees that the adversary cannot generate malicious decryption queries, even if the challenge ciphertext contains a simulated proof at some point. It is then easy using standard Naor-Yung gymnastic to argue about IND-CR-CCA security, assuming the IND-CR-CPA security of our first construction. To obtain an efficient simulation-sound NIZK proof for plaintext equality, we first build a standard Σ-protocol for plaintext equality and prove it statistically sound and statistically special honest-verifier zero-knowledge. We also show that it satisfies a notion of quasi-unique responses [23] assuming that factoring is hard (which is implied by the DCR assumption). The Fiat-Shamir heuristic [24] can then compile this Σ-protocol into a simulation-sound NIZK in the random oracle model, by applying a result from [23]. Our Σ-protocol is directly inspired by Schnorr-like protocols in groups of unknown order [14, 26]. As a result, we obtain an efficient IND-CR-CCA UPKE scheme based

[4] We could reverse the argument by picking exponentially-large updates r and relying on the same CS+LR security notion as Dodis *et al.* [21], but as secret keys are sampled once while updates could be sampled every time a message is sent, we opt for large secret keys and small updates. Taking large updates would also prevent us from using $\zeta = 1$ as we need the updates to fit into the plaintext space N^{ζ}.

on the DCR assumption in the random oracle model. In terms of efficiency, this IND-CR-CCA construction only incurs a fairly small overhead w.r.t. its IND-CR-CPA counterpart.

From IND-CR-CCA to IND-CU-CCA Security. We finally enhance the latter IND-CR-CCA construction to achieve IND-CU-CCA security. The main change is to add a NIZK argument in the update message to prove its well-formedness: each update comes with an argument showing that the update randomness r being encrypted is the randomness that allows moving from pk_{i-1} to $pk_i = pk_{i-1} \cdot g^r$. In order to prove CU-CCA security, we also need to apply the Naor-Yung paradigm in update messages. Eventually, an update with randomness r is formed of the updated public key pk_i, as well as $ct, ct'_d, \pi_{NY}, \pi_{WF}$ where (ct, ct'_d, π_{NY}) form a standard Naor-Yung-like double encryption and π_{WF} is a proof that pk_{i-1}, pk_i, ct are consistent. We instantiate the Naor-Yung part exactly as in Enc, except that we use a second public parameters h'_d as a public key for ct'_d. The reason is that we prove IND-CU-CCA security via a reduction from the IND-CR-CCA security of our second scheme. The security proof follows standard Naor-Yung techniques again, with several crucial observations: (i) The challenger only generates an update at the very end of the security game, and therefore even if the NIZK argument of well-formedness in this update is simulated, the adversary cannot take advantage of this to generate proofs for ill-formed updates. In particular, we only need the NIZK argument of update well-formedness to achieve computational soundness, rather than simulation-soundness. (ii) Using a third public key h'_d for the Naor-Yung encryption of updates prevents the adversary from sending the challenge ciphertext as part of an update query. It also serves as a trapdoor to decrypt the randomness used in updates in order to reduce IND-CU-CCA security to IND-CR-CCA, where an adversary can only produce updates based on (possibly malicious) randomness. The rest of the proof follows standard techniques.

To construct the NIZK argument of update well-formedness, we build a Σ-protocol and apply the Fiat-Shamir transform to make it non-interactive. The underlying Σ-protocol bears similarities with Schnorr-like protocols for groups of unknown order [26] based on the Strong RSA assumption. It is a variant of a construction due to Camenisch and Shoup [14] to prove that a Paillier encryption coincides with a discrete logarithm in a prime order group. The main difference is that, in our setting, the encrypted discrete logarithm lives in a cyclic group of order pq (instead of a group of prime order $\rho < 2^\lambda pq$ in [14, Section 5.2]). By enlarging the message space $\mathbb{Z}_{N^\zeta}$ with $\zeta \geq 2$ in the fashion Damgård and Jurik [19], we can adapt the proof of soundness of [14].

Implementation. To complement our work, we provide an implementation of our first two constructions. The key/ciphertext/update size of our schemes makes the computation tractable, especially compared to the DDH-based standard-model construction from [21]. This implementation is heavily optimizable. The running times are a factor away of what one would expect in real world applications, but as UPKEs are meant to achieve forward security in the world of secure messaging, we believed that it was interesting to provide a very simple implementation as a proof of concept.

2 BACKGROUND AND DEFINITIONS

In this preliminary section, we recall formal definitions and security models for UPKE in Section 2.1 and the hardness assumptions on which we rely in Section 2.2. We also remind some useful lemmas in Section 2.3. Additional preliminaries on Σ-protocols, non-interactive zero-knowledge proofs, and the Fiat-Shamir transform are provided in Appendix A for completeness.

2.1 Syntax and Definitions of Updatable Public Key Encryption

We follow the syntax of [21], which is recalled hereunder.

Definition 1. *An updatable public-key encryption (UPKE) scheme is a tuple* (GGen, KGen, Enc, Dec, UpdatePk, UpdateSk) *of efficient algorithm with the following syntax:*

- **Public parameters:** *On input of a security parameters* 1^λ, GGen *generates a set of common public parameters* pp.
- **Key generation:** KGen *takes in public parameters* pp. *It outputs a fresh secret key* sk_0 *and an initial public key* pk_0.
- **Encryption:** Enc *takes in a public key* pk *and a message* m. *It outputs a ciphertext* ct.
- **Decryption:** Dec *inputs a secret key* sk *and a ciphertext* ct *to output a plaintext* m.
- **Update Public Key:** *Given a public key* pk, UpdatePk *produces an update ciphertext* up *and a new public key* pk'.
- **Update Secret Key:** UpdateSk *receives an update ciphertext* up *and secret key* sk. *It outputs a new secret key* sk'.

Correctness*: Let a key pair* $(pk_0, sk_0) \leftarrow \mathsf{KGen}(1^\lambda)$ *and let* $\ell \in \mathbb{N}$. *Define* $(pk_i, up_i) \leftarrow \mathsf{UpdatePk}(pk_{i-1})$ *and* $sk_i \leftarrow \mathsf{UpdateSk}(sk_{i-1}, up_{i-1})$ *for* $j \leq \ell$. *An UPKE scheme provides correctness if, for any* m *in the message space* $\mathcal{M}$ *and all* $i \leq \ell$*:* $\Pr[\mathsf{Dec}(sk_i, \mathsf{Enc}(pk_i, m)) = m] = 1$.

We recall the notion of IND-CR-CPA security, as formalized by Dodis *et al.* [21], which strengthens the original definition of [34].

Definition 2 ([21],IND-CR-CPA security). *Let UPKE =* (GGen, KGen, Enc, Dec, UpdatePk, UpdateSk) *be an UPKE scheme.*

We first define the update oracle $\mathcal{O}_{upd}(\cdot)$*:*

$\mathcal{O}_{upd}(\cdot)$ *: Given a randomness* r_i*, the challenger increments the epoch to* $i+1$. *It then performs the following actions:*

$(up_{i+1}, pk_{i+1}) \leftarrow \mathsf{UpdatePk}(pk_i; r_i); sk_{i+1} \leftarrow \mathsf{UpdateSk}(sk_i, up_{i+1})$.

For any PPT adversary $\mathcal{A}$*, we consider the IND-CR-CPA game:*

1. *Run* $\mathsf{GGen}(1^\lambda)$ *to get the public parameters* pp*, then sample* $(sk_0, pk_0) \leftarrow \mathsf{KGen}(1^\lambda, pp), b \hookleftarrow \mathcal{U}(\{0,1\})$.
2. $(m_0^*, m_1^*, state) \leftarrow \mathcal{A}^{\mathcal{O}_{upd}(\cdot)}(pk_0)$.
3. *Compute* $c^* \leftarrow \mathsf{Enc}(pk_\ell, m_b^*)$ *where* ℓ *is the current epoch.*
4. $state \leftarrow \mathcal{A}^{\mathcal{O}_{upd}(\cdot)}(c^*, state)$
5. *Choose uniformly random* r^* *and then compute*

$$\begin{aligned} (up^*, pk^*) &\leftarrow \mathsf{UpdatePk}(pk_\ell, r^*); \\ sk^* &\leftarrow \mathsf{UpdateSk}(sk_\ell, up^*), \end{aligned}$$

   where ℓ *is the current epoch.*
6. $b' \leftarrow \mathcal{A}(pk^*, sk^*, up^*, state)$.
7. $\mathcal{A}$ *wins the game if* $b = b'$*. The advantage of* $\mathcal{A}$ *in winning the above game is denoted by* $\mathbf{Adv}^{\mathrm{UPKE}}_{cr\text{-}cpa}(\mathcal{A}) = \left|\Pr[b = b'] - \frac{1}{2}\right|$.

An UPKE scheme is IND-CR-CPA -secure if for all PPT attackers $\mathcal{A}$, its advantage $\mathrm{Adv}^{\mathrm{UPKE}}_{cr\text{-}cpa}(\mathcal{A})$ is negligible.

Definition 2 only considers passive adversaries. In the chosen-ciphertext scenario, the following definition considers stronger adversaries that are granted access to a decryption oracle.

Definition 3 ([21], IND-CR-CCA security). *Let UPKE =* (KGen, Enc, Dec, UpdatePk, UpdateSk) *be an UPKE scheme.*

The update oracle is specified as in Definition 2 while the decryption oracle $O_{dec}(\cdot)$ is defined as follows.
$O_{dec}(\cdot)$: Given a ciphertext c, compute and return $m = \mathsf{Dec}(sk_i, c)$ (which may be $\perp$ if c is an invalid ciphertext), where sk_i is the secret key of the current epoch.

For any PPT adversary $\mathcal{A}$, the IND-CR-CCA game goes as follows:

1. *Sample* $(sk_0, pk_0) \leftarrow \mathsf{KGen}(1^\lambda), b \hookleftarrow \mathcal{U}(\{0,1\})$.
2. $(m_0^*, m_1^*, state) \leftarrow \mathcal{A}^{O_{upd}(\cdot), O_{dec}(\cdot)}(pk_0)$
3. *Compute* $c^* \leftarrow \mathsf{Enc}(pk_\ell, m_b^*)$ *where ℓ is the current epoch.*
4. $state \leftarrow \mathcal{A}^{O_{upd}(\cdot), O_{dec}(\cdot)}(c^*, state)$.
   - *No query is allowed to $O_{dec}(\cdot)$ for the challenge c^* until $\mathcal{A}$ makes at least one query to the update oracle $O_{upd}(\cdot)$.*
5. *Choose a uniformly random r^* and then compute*

$$\begin{aligned} (up^*, pk^*) &\leftarrow \mathsf{UpdatePk}(pk_{\ell'}, r^*); \\ sk^* &\leftarrow \mathsf{UpdateSk}(sk_{\ell'}, up^*), \end{aligned}$$

   where ℓ' is the current epoch.
6. $b' \leftarrow \mathcal{A}(pk^*, sk^*, up^*, state)$.
7. *$\mathcal{A}$ wins the game if $b = b'$. The advantage of $\mathcal{A}$ in winning the above game is denoted by* $\mathbf{Adv}^{\mathrm{UPKE}}_{cr\text{-}cca}(\mathcal{A}) = \left|\Pr[b = b'] - \frac{1}{2}\right|$.

While Definition 3 allows running the update oracle on input of adversarially-generated randomness, it still assumes that the update algorithm is honestly executed. Following [21], we introduce an even stronger definition where update messages are *not* assumed to be honestly generated. Instead, we can allow the adversary to come up with update messages of its own (instead of just providing the randomness to be used) that may not be a legitimate output of the UpdatePk algorithm.

In the UPKE syntax, as suggested in [21], we introduce a new algorithm $\mathsf{VerifyUpdate}(pk, up, pk')$. This algorithm performs a verification of the update message (up, pk'). It returns 1 if the update (up, pk') is consistent with the public key pk (that is, it corresponds to the output of $\mathsf{UpdatePk}(pk; r)$ for some valid random coin r) and $\perp$ otherwise.

Definition 4 ([21], IND-CU-CCA security). *Let UPKE =* (KGen, Enc, Dec, UpdatePk, UpdateSk, VerifyUpdate) *be an UPKE scheme. We say that it provides IND-CU-CCA security if no PPT adversary has non-negligible advantage in the same game as in the one of Definition 3, except the update oracle is specified in the following way:*

- *$O_{upd}(\cdot)$: Given an update message (up, pk'), it runs $\tau \leftarrow \mathsf{VerifyUpdate}(pk_i, up, pk')$ (which may be $\perp$ if (up, pk') if an invalid update), where pk_i is the public key of the current epoch. If $\tau \neq \perp$, then it runs $sk' \leftarrow \mathsf{UpdateSk}(sk, up)$, else it returns $\perp$.*

2.2 Hardness Assumptions

We rely on two number theoretic assumptions that imply the hardness of factoring.

Definition 5 ([40]). *Let an RSA modulus $N = PQ$, for primes P, Q, and let an integer $\zeta \geq 1$. The ζ-Decision Composite Residuosity (ζ-DCR) assumption holds if, for any PPT adversary $\mathcal{A}$, we have*

$$\begin{aligned} \mathbf{Adv}^{\mathrm{dcr}}_{\zeta,\mathcal{A}}(\lambda) = \Big|\Pr\left[\mathcal{A}\left(N, r^{N^\zeta} \bmod N^{\zeta+1}\right) = 1 \mid r \hookleftarrow \mathcal{U}(\mathbb{Z}_N^*)\right] \\ - \Pr\left[\mathcal{A}(N, z) = 1 \mid z \hookleftarrow \mathcal{U}(\mathbb{Z}^*_{N^{\zeta+1}})\right]\Big| = \mathsf{negl}(\lambda), \end{aligned}$$

Damgård and Jurik [19] showed that all DCR assumptions are equivalent (up to a polynomial loss) for any $\zeta \in \mathrm{poly}(\lambda)$.

When $N = PQ$ is a product of safe primes $P = 2p + 1$ and $Q = 2q + 1$ (i.e., where p and q are also primes), the DCR assumption can equivalently be defined as follows.

Definition 6 ([31]). *For a safe-prime product $N = PQ$, let $T = 1 + N$. The Decision Composite Residuosity (DCR) assumption asserts that, for any PPT adversary $\mathcal{A}$, we have*

$$\begin{aligned} \mathbf{Adv}^{\mathrm{dcr}}_{\zeta,\mathcal{A}}(\lambda) = \Big|\Pr\left[\mathcal{A}\left(N, g, g^r \bmod N^{\zeta+1}\right) = 1\right] \\ - \Pr\left[\mathcal{A}\left(N, g, T \cdot g^r \bmod N^{\zeta+1}\right) = 1\right]\Big| = \mathsf{negl}(\lambda), \end{aligned}$$

where $r \hookleftarrow \mathcal{U}(\mathbb{Z}_n)$ and $g = \mu^{N^\zeta} \bmod N^{\zeta+1}$, where $\mu \hookleftarrow \mathcal{U}(\mathbb{Z}_N^*)$.

In our IND-CU-CCA secure UPKE construction, we also rely on the Strong RSA assumption to prove the soundness of a non-interactive argument system in the random oracle model.

Definition 7 ([3]). *Let a safe-prime product $N = PQ$ where P, Q are primes of at least $l(\lambda)$ bits for some polynomial $l : \mathbb{N} \to \mathbb{N}$. The Strong RSA assumption states that, for any PPT algorithm $\mathcal{A}$, we have*

$$\begin{aligned} \Pr[y = x^e \bmod N \ \wedge \ e > 1 \mid \\ y \hookleftarrow \mathcal{U}(\mathbb{Z}_N^*),\ (x, e) \leftarrow \mathcal{A}(N, y)] = \mathsf{negl}(\lambda) \end{aligned}$$

Looking ahead, we rely (in the proof of Lemma 6) on the Strong RSA assumption in the subgroup of $2N^\zeta$-th residues in $\mathbb{Z}^*_{N^{\zeta+1}}$. However, an algorithm that computes a non-trivial e-th root of $g = \mu^{2N^\zeta} \in \mathbb{Z}^*_{N^{\zeta+1}}$, for a random $\mu \in \mathbb{Z}_N^*$, can be used to solve the usual Strong RSA problem in the subgroup $\mathbb{QR}_N$ of quadratic residues in $\mathbb{Z}_N^*$ as long as e is restricted to be co-prime to N. Given $v = \mu^2 \bmod N$, for some $\mu \in \mathbb{Z}_N^*$, one can set $g = v^{N^\zeta} \bmod N^{\zeta+1}$. From an adversary that outputs $w \in \mathbb{Z}^*_{N^{\zeta+1}}$ and $e > 1$ such that $\gcd(e, N) = 1$ and $g = w^e \bmod N^{\zeta+1}$, one can apply Shamir's trick to compute $\alpha, \beta \in \mathbb{Z}$ such that $\alpha e + \beta N^\zeta = 1$, which yields $v = (w^\beta \cdot v^\alpha)^e \bmod N^{\zeta+1}$. Then, $w' \triangleq w^\beta \cdot v^\alpha \bmod N$ satisfies $v = w'^e \bmod N$.

2.3 Useful Lemmas

In the next section, we rely on the following lemma, which was proven by Kitagawa *et al.* [35] (based on earlier ideas from [12, 37]) in the context of key-dependent message security [6].

Definition 8. *(Interactive vector game from [35].) Let $\zeta \geq 1$ be an integer and κ be a polynomial of λ. We define the following* $\mathrm{IV}_{\zeta,\kappa}$ *game between a challenger and an adversary $\mathcal{A}$.*

- *The challenger chooses a challenge bit* $b \hookleftarrow \mathcal{U}(\{0,1\})$ *and generates* $(N,P,Q,T,g) \leftarrow \mathsf{GGen}(1^\lambda,\zeta)$. *The challenger generates* $\alpha_i \hookleftarrow \mathcal{U}([0,(N-1)/4])$ *and computes* $g_i \leftarrow g^{\alpha_i} \bmod N^{\zeta+1}$ *for every* $i \in [\kappa]$, *and sends* $N, g,$ *and* $g_1,\ldots,g_\kappa$ *to* $\mathcal{A}$.
- $\mathcal{A}$ *can adaptively make sample queries* $(a_1,\ldots,a_\kappa) \in \mathbb{Z}_{N^\zeta}^\kappa$ *to the challenger. The challenger generates* $r \hookleftarrow \mathcal{U}([0,(N-1)/4])$ *and computes* $e_i \leftarrow T^{b\cdot a_i} \cdot g_i^r \bmod N^{\zeta+1}$ *for every* $i \in [\kappa]$. *The challenger then returns* $(e_1,\ldots,e_\kappa)$ *to* $\mathcal{A}$.
- $\mathcal{A}$ *outputs* $b' \in \{0,1\}$ *and wins if* $b' = b$.

We remind the following result from [37].

Lemma 1. *(DCR Interactive Vector Lemma for* $\kappa = 1, 2$*, adapted from [37]). For* $\kappa = 1, 2$*, no polynomial time adversary can have non-negligible advantage in* $\mathrm{IV}_{\zeta,\kappa}$ *with a polynomial number of queries, under the DCR assumption.*

In the next section, we also rely on the following standard smudging lemma.

Lemma 2 (Smudging lemma). *Let* $B_1, B_2 \in \mathbb{N}$, $a \hookleftarrow \mathcal{U}([-B_1,B_1])$, $b \hookleftarrow \mathcal{U}([-B_2,B_2])$ *with* $B_2/B_1 = \mathsf{negl}(\lambda)$. *Then, the distribution of* $(a+b,a)$ *is statistically indistinguishable from that of* (a,b).

Proof. Let $z \in \mathbb{N}$ with $B_2 - B_1 \le z < B_1 - B_2$ and $c \hookleftarrow \mathcal{U}([-B_1,B_1])$. Then, using the independence of a and b,

$$\begin{aligned}\Pr[a+b=z \wedge b=y] &= \Pr[a+y=z \wedge b=y]\\ &= \Pr[b=y]\cdot\Pr[a=z-y] = \Pr[b=y]\cdot\Pr[a=z-y]\\ &= \Pr[b=y]\cdot\Pr[a=z]\end{aligned}$$

The last equality being valid as $z - y \in [-B_1, B_1]$ by assumption. Also, $\Pr[|a+b| > B_1 - B_2] \le \Pr[|a| \ge B_1 - 2B_2] = (2B_2+1)\frac{1}{2B_1+1} = \mathsf{negl}(\lambda)$ by our assumptions. □

3 A DCR-BASED IND-CR-CPA-SECURE UPKE

In this section, we construct a UPKE scheme that we prove to be IND-CR-CPA secure assuming the DCR assumption. We follow a similar roadmap as in [21] and start by constructing a circular-secure and leakage-resilient (CS+LR) PKE scheme, for a well-chosen leakage function, under the DCR assumption.

We then extend this construction into a UPKE scheme by appending the additional two key update algorithms, and reduce the security of the resulting scheme to the CS+LR security of the underlying PKE scheme.

3.1 A DCR-Based CR+LR Secure PKE

Definition 9 (CS+LR security, adapted from [21]). *Let* (GGen, KGen, Enc, Dec) *be a PKE scheme with message space* $\mathcal{M}$ *and integer secret keys. For* $B \in \mathbb{Z}$, *we say that the scheme is B-CS+LR secure if, for any PPT adversary* $\mathcal{A} = (\mathcal{A}_1,\mathcal{A}_2)$, *we have:*

$$\Big|\Pr[pp \leftarrow \mathsf{GGen}(1^\lambda);\ (pk,sk) \leftarrow \mathsf{KGen}(pp);\ b \hookleftarrow \mathcal{U}(\{0,1\});\ r \hookleftarrow \mathcal{U}([-B,B]);\ (m_0,m_1,st) \leftarrow \mathcal{A}_1(pp,pk) :\ \mathcal{A}_2(st,pk,\mathsf{Enc}(pk,m_b),\mathsf{Enc}(pk,r),sk+r)=b] - \frac{1}{2}\Big| = \mathsf{negl}(\lambda),$$

We refer to the above quantity as the advantage of $\mathcal{A}$ *against CS+LR security, denoted by* $\mathbf{Adv}^{\mathrm{cs+lr}}_{\mathcal{A}}$.

Difference with the Dodis et al. definition. Our notion of CS+LR security defers slightly from the one defined by Dodis *et al.* in the sense that it is actually not an extension of circular-security. Here, the adversary is given a randomized leakage of the secret key $sk + r$ and an encryption of r, whereas the definition of [21] gives of an encryption of (a function of) sk. We could modify our construction to rely on standard CS+LR security, where an encryption of sk is revealed instead of an encryption of sk. We opt to do the opposite as it allows reducing the size of update messages in our UPKE construction. Indeed, in our security proof we use smudging on the sum $sk + r$, in which sk and r play a symmetric part, allowing us to chose which of the two should be exponentially larger than the other.

We now describe our DCR-based B-CS+LR scheme.

- $\mathsf{GGen}(1^\lambda, B)$: Generate two λ-bit safe primes $P = 2p+1$, $Q = 2q+1$, where p, q are also primes, and set $N = PQ$. Let $\zeta \ge 1$, choose $\mu \hookleftarrow \mathcal{U}(\mathbb{Z}_N^*)$, and define a generator g of $\mathbb{Z}^*_{N^{\zeta+1}}$ of the subgroup of order $n = \varphi(N)/4 = pq$ by setting $g = \mu^{2N^\zeta} \bmod N^{\zeta+1}$. Define $pp = (N, g, \zeta, B)$ for the public parameters. We also set $T = 1 + N$.
- $\mathsf{KGen}(pp)$: Sample a random $x \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$. Set
$$pk = h = g^x \bmod N^{\zeta+1}, \qquad sk = x.$$
- $\mathsf{Enc}(pk, m)$: Pick $t \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and return
$$ct = \left(g^t \bmod N^{\zeta+1},\ T^m \cdot h^t \bmod N^{\zeta+1}\right).$$
- $\mathsf{Dec}(sk, ct)$: Parse ct as (c_0, c_1) and return
$$m = \mathsf{DLog}\left(c_1 \cdot (c_0)^{-x} \bmod N^{\zeta+1}\right)$$

When $\zeta > 1$, the decryption algorithm can use the Damgård-Jurik technique [19] to efficiently recover $m \in \mathbb{Z}_{N^\zeta}$ from $(1+N)^m \bmod N^{\zeta+1}$, even when m is arbitrarily large.

Theorem 1. *Under the DCR assumption, the above PKE construction provides B-CS+LR security for any* $B \in \mathbb{N}$ *such that the distributions* $\{u \bmod pq \mid u \hookleftarrow \mathcal{U}([-B,B])\}$ *and* $\mathcal{U}(\mathbb{Z}_{pq})$ *are statistically close. In particular, it holds for* $B = (N-1)/4$.

Proof. We prove the result using a sequence of four games.

Game_0: This is the original CS+LR game obtained by encrypting the message m_b, for some $b \hookleftarrow \mathcal{U}(\{0,1\})$. The adversary is given pk, $\mathsf{Enc}(pk, m_b)$, $\mathsf{Enc}(pk, r)$ and the leakage $L(x; r) = x + r \in \mathbb{Z}$, where $pk = g^x \bmod N^{\zeta+1}$ for $x \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$ and $r \hookleftarrow \mathcal{U}([-B,B])$.

Game_1: We change the generation of $\mathcal{A}_2$'s input

$$pk,\ \mathsf{Enc}(pk, m_b),\ \mathsf{Enc}(pk, r),\ x + r. \tag{1}$$

We first compute

$$pk',\ \mathsf{Enc}(pk', m_b),\ \mathsf{Enc}(pk', r),\ x + r, \tag{2}$$

where $r \hookleftarrow \mathcal{U}([-B,B])$ and $pk' = g^{-r} \bmod N^{\zeta+1}$. Then, we compute $pk = pk' \cdot g^{(x+r)}$, while $\mathsf{Enc}(pk, m_b) = (g^t, T^{m_b} \cdot pk^t)$ is computed from $\mathsf{Enc}(pk', m_b) = (g^t, T^{m_b} \cdot pk'^t)$ as

$$\mathsf{Enc}(pk, m_b) = (g^t, T^{m_b} \cdot pk'^t \cdot (g^t)^{(x+r)}).$$

$\mathsf{Enc}(pk, r)$ can be computed from $\mathsf{Enc}(pk', r)$ in a similar way. While the distribution of (1) is statistically close to that of Game_1 (in

particular, the distribution of pk' is statistically uniform in the subgroup generated by g thanks to the constraint on r), the new distribution does not contain any information about x, except in the leakage $x + r \in \mathbb{Z}$, since (1) can be computed from (2).

Game$_2$: In this game, we replace the leakage $L(x;r) = x + r$ by a random element $u \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$ in (2). Since $g^{-r} \bmod N^{\zeta+1}$, $\mathsf{Enc}(g^{-r}, m_b)$, $\mathsf{Enc}(g^{-r}, r)$ do not depend on x and since

$$\Delta(\mathcal{U}([-2^\lambda B, 2^\lambda B]) * \mathcal{U}([-B, B]), \mathcal{U}([-2^\lambda B, 2^\lambda B])) \leq B/2^\lambda B = \mathsf{negl}(\lambda)$$

by Lemma 2, it follows that Game$_2$ and Game$_1$ are statistically indistinguishable.

Game$_3$: In this game, we replace $\mathsf{Enc}(g^{-r}, r) = (g^t, T^r(g^{-r})^t)$ by a pair $(Tg^t, (g^{-r})^t)$ in (2). The DCR assumption guarantees that $g^t \approx_c Tg^t$, as t is unknown to the adversary. Hence, assuming DCR, we have

$$(g^t, T^r(g^t)^{-r}) \approx_c \left(Tg^t, T^r(Tg^t)^{-r}\right) = \left(Tg^t, (g^{-r})^t\right),$$

and distributions from Game$_3$ and Game$_2$ are computationally indistinguishable. In Game$_3$, $\mathcal{A}_2$'s input is thus generated from a tuple

$$pk' = g^{-r},\ \mathsf{Enc}(g^{-r}, m_b),\ (Tg^t, (g^{-r})^t),\ u, \tag{3}$$

where $r \hookleftarrow \mathcal{U}([-B, B])$, $u \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$.

Game$_4$: We finally replace $\mathsf{Enc}(g^{-r}, m_b) = (g^k, T^{m_b} \cdot (g^{-r})^k)$ and $(Tg^t, (g^{-r})^t)$ from the distribution of (3) by elements $\mathsf{Enc}(g^{-r}, 0) = (g^k, (g^{-r})^k)$ and $(g^t, (g^{-r})^t)$.

Using the Interactive Vector game $\mathrm{IV}_{\zeta,2}$ for $(g_1, g_2) = (g, g^{-r} \bmod N^{\zeta+1})$ with two queries $(0, m_b)$ and $(1, 0)$ (where the IV challenger replies using randomness $k, t \hookleftarrow U(\mathbb{Z}_{(N-1)/4})$), it is straightforward to prove that these two distributions are computationally indistinguishable via Lemma 1, assuming that DCR holds. Note that this step again requires the distribution of $g^{-r} \bmod N^{\zeta+1}$ to be statistically close to the distribution $\{g^\alpha \bmod N^{\zeta+1} \mid \alpha \hookleftarrow \mathcal{U}((N-1)/4)\}$, as guaranteed by the constraint on B in the statement of Theorem 1.

We finally observe that the distribution Game$_4$ is independent of the bit $b \in \{0, 1\}$. Indeed, g, $g^{-r} \bmod N^{\zeta+1}$, $(g^k, (g^{-r})^k)$, and $(g^t, (g^{-r})^t)$ do not carry any information about m_b. This concludes the proof of Theorem 1. □

3.2 A DCR-Based IND-CR-CPA-Secure UPKE

To construct our UPKE, we extend the CS+LR PKE scheme described in Section 3.1 with algorithms (UpdatePk, UpdateSk), defined as follows. We set $B = (N-1)/4$ for our UPKE scheme, which satisfies the requirements of Theorem 1. In the description below, we include the current epoch number in each public key as it simplifies our security proofs in the CCA-secure extensions later on.

- GGen(1^λ): Generate two λ-bit safe primes $P = 2p + 1, Q = 2q + 1$, where p, q are also primes, and set $N = PQ$. Choose $\mu \hookleftarrow \mathcal{U}(\mathbb{Z}_N)$, and obtain a generator g of the subgroup of order $n = \varphi(N)/4 = pq$ by setting $g = \mu^{2N^\zeta} \bmod N^{\zeta+1}$. Define $pp = (N, g)$ as public parameters.
- KGen(pp): Sample a random $x \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$ and compute $h = g^x \bmod N^{\zeta+1}$. Let $\tau = 0$ the current epoch. Set $pk = (\tau, h)$, $sk = (\tau, x)$.
- Enc(pk, m): Pick $t \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and return
$$ct = (g^t \bmod N^{\zeta+1}, (1+N)^m h^t \bmod N^{\zeta+1}).$$
- Dec(sk, ct): Parse ct as (c_0, c_1) and sk as (τ, x). Return
$$m = \mathsf{DLog}\left(c_1 \cdot c_0^{-x} \bmod N^{\zeta+1}\right).$$
- UpdatePk(pk): Parse pk as (τ, h).
  1. Sample $r \hookleftarrow \mathcal{U}([-B, B])$ and compute
  $$h' = h \cdot g^r \bmod N^{\zeta+1}.$$
  2. Compute $up \leftarrow \mathsf{Enc}(pk, r)$ and set $\tau \leftarrow \tau + 1$.
  Return $(up, (\tau, h'))$.
- UpdateSk(sk, up): Given the secret key $sk = (\tau, x)$, compute $r \leftarrow \mathsf{Dec}(sk, up) \in \mathbb{Z}_{N^\zeta}$. Let $\bar{r} = \min(r, N^\zeta - r)$ and let $b_r \in \{0, 1\}$ such that $\bar{r} = (1 - b_r) \cdot r + b_r \cdot (N^\zeta - r) \bmod N^\zeta$. Return $sk' = (\tau + 1, x')$, where $x' = x + (-1)^{b_r} \cdot r \in \mathbb{Z}$.

Theorem 2. *Under the DCR assumption, the above UPKE construction provides IND-CR-CPA security.*

Proof. Let $\mathcal{A}$ be an adversary in the IND-CR-CPA security game of our UPKE. We build an adversary $\mathcal{B}$ against the CS+LR security of the PKE scheme in Section 3.1 with chosen leakage $L(x;r) = x + r \in \mathbb{Z}$ for $r \hookleftarrow \mathcal{U}([-B, B])$ and $x \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$.

- Our adversary $\mathcal{B}$ receives from its challenger a public key $pk = h = g^x \bmod N^{\zeta+1}$ associated with a secret key x.
- $\mathcal{B}$ implicitly sets $x_0 = x$ and sets the epoch τ to 0 by forwarding $pk_0 = (0, h)$ to $\mathcal{A}$.
- When $\mathcal{A}$ queries oracle $\mathcal{O}_{upd}(\delta_i)$, $\mathcal{B}$ just bookkeeps δ_i.
- At some point, $\mathcal{B}$ receives the challenge messages m_0^*, m_1^* from $\mathcal{A}$. Adversary $\mathcal{B}$ forwards m_0^*, m_1^* to its challenger, using them as challenge messages. In response, $\mathcal{B}$ receives a leakage $z = L(x;r) = x + r \in \mathbb{Z}$, and ciphertexts $c = \mathsf{Enc}(pk, m_b^*)$ and $c' = \mathsf{Enc}(pk, r)$. In order to build a challenge ciphertext for $\mathcal{A}$, $\mathcal{B}$ needs to compute $c^* = \mathsf{Enc}(pk_\ell, m_b^*)$, where ℓ is the epoch at which $\mathcal{A}$ sent its challenge messages. We have $pk_\ell = (\ell, g^{x+\sum_{i=1}^\ell \delta_i} \bmod N^{\zeta+1})$. Parsing $c = (c_0, c_1)$ and defining $\delta = \sum_{i=1}^\ell \delta_i$ (computed over $\mathbb{Z}$), $\mathcal{B}$ simply computes
$$c^* = \left(c_0, c_0^\delta \cdot c_1 \bmod N^{\zeta+1}\right)$$
and forwards c^* as challenge ciphertext to $\mathcal{A}$.
- Then, $\mathcal{A}$ resumes its queries of $\mathcal{O}_{upd}(\cdot)$ and $\mathcal{B}$ keeps registering them until the last query $\mathcal{O}_{upd}(\delta_{\ell'})$ is made.
- $\mathcal{B}$ has to send the final secret key $sk^* = (\ell' + 1, x^*)$, which is the result of all update queries made by $\mathcal{A}$ and a last honestly-generated update (of which the randomness is not known to $\mathcal{A}$). Letting $\delta' = \sum_{i=1}^{\ell'} \delta_i \in \mathbb{Z}$, the reduction $\mathcal{B}$ uses the leakage randomness r as the randomness of the final update, which is hidden from $\mathcal{A}$'s view. Namely, $\mathcal{B}$ sets $x^* = z + \delta' = x + r + \delta$ (over $\mathbb{Z}$) and defines the new public for the next epoch to be $pk^* = (\ell'+1, g^{x^*} \bmod N^{\zeta+1})$. To finalize the update message, $\mathcal{B}$ can simply update the public key used

in the encryption of the c' from g^{x_0} to $g^{x_0+\delta}$. This is done using the same technique as for the challenge ciphertext. Namely, given $c' = \mathsf{Enc}(pk, r) = (c'_0, c'_1)$, $\mathcal{B}$ simply sets $up^* = \left(c'_0, {c'_0}^{\delta'} \cdot c'_1 \bmod N^{\zeta+1}\right)$ and then sends (pk^*, sk^*, up^*) to $\mathcal{A}$.
- When $\mathcal{A}$ finally halts with some output $b' \in \{0,1\}$, $\mathcal{B}$ outputs the same bit b'.

By inspection, it is clear that $\mathcal{B}$ perfectly simulates $\mathcal{A}$'s challenger in the IND-CR-CPA security game and succeeds whenever $\mathcal{A}$ does. Theorem 2 follows. □

4 FROM CR-CPA TO CR-CCA/CU-CCA SECURITY IN THE ROM

We now enhance our UPKE scheme so that it becomes IND-CR-CCA secure. To achieve that we need to build Non Interactive Zero Knowledge (NIZK) arguments that allow to perform the Naor-Yung transformation for our IND-CPA-secure PKE. We start by constructing a Σ-protocol for proving plaintext equality in Section 4.1, which we transform into a simulation-sound NIZK arguments via the Fiat-Shamir transform and use to build an IND-CR-CCA secure UPKE scheme in Section 4.2.

We then further upgrade the scheme to achieve IND-CU-CCA security in Section 4.4, by adding NIZK arguments that updates are well-formed. The latter NIZK arguments are obtained in a similar way by building a Σ-protocol in Section 4.3 and applying the Fiat-Shamir transform.

4.1 Proofs of Plaintext Equality

In order to achieve IND-CR-CCA security in the ROM, we apply the Naor-Yung transformation [39] in the random oracle model. Let public parameters $pp = (N, g, \zeta)$, where $N = PQ$ is a safe-prime product and $g \in \mathbb{Z}^*_{N^{\zeta+1}}$ generates the subgroup of $2N^\zeta$-th residues in $\mathbb{Z}^*_{N^{\zeta+1}}$. Let $h = g^x \bmod N^{\zeta+1}$, $h_d = g^{x_d} \bmod N^{\zeta+1}$ be two public keys, for some underlying secret keys $x, x_d \in \mathbb{Z}_{2^{2\lambda}(N-1)/4}$.

We need to prove membership of the language

$$\mathcal{L}_{NY} = \Big\{ (h, h_d, C_0, C_1, D_0, D_1) \in (\mathbb{Z}^*_{N^{\zeta+1}})^6 \;\mid\; \exists t_c, t_d \in \mathbb{Z}_{pq},\ m \in \mathbb{Z}_{N^\zeta} \;:\; C_0^2 = g^{2\cdot t_c} \bmod N^{\zeta+1} \;\wedge\; C_1^2 = (1+N)^{2m} \cdot h^{2\cdot t_c} \bmod N^{\zeta+1} \;\wedge\; D_0^2 = g^{2\cdot t_d} \bmod N^{\zeta+1} \;\wedge\; D_1^2 = (1+N)^{2m} \cdot h_d^{2\cdot t_d} \bmod N^{\zeta+1} \Big\},$$

The Σ-protocol for $\mathcal{L}_{NY}$ is standard and goes as follows.

- $\mathcal{P}$ picks $t'_c, t'_d \hookleftarrow \left[0, 2^{2\lambda} \cdot \frac{N-1}{4}\right]$, $m' \hookleftarrow \mathbb{Z}_{N^\zeta}$ and sends
$$\begin{aligned} C'_0 &= g^{2\cdot t'_c} \bmod N^{\zeta+1}, \\ C'_1 &= (1+N)^{2m'} \cdot h^{2\cdot t'_c} \bmod N^{\zeta+1}, \\ D'_1 &= g^{2\cdot t'_d} \bmod N^{\zeta+1}, \\ D'_1 &= (1+N)^{2m'} \cdot h_d^{2\cdot t'_d} \bmod N^{\zeta+1}. \end{aligned}$$
- $\mathcal{V}$ sends a random challenge $c \hookleftarrow \mathcal{U}([0, 2^\lambda - 1])$.
- $\mathcal{P}$ computes $\tilde{t}_c = t'_c + c \cdot t_c$, $\tilde{t}_d = t'_d + c \cdot t_d$ (over $\mathbb{Z}$) and $\tilde{m} = m' + c \cdot m \bmod N^\zeta$. If $\tilde{t}_c, \tilde{t}_d \in \left[0, 2^{2\lambda} \cdot \frac{N-1}{4}\right]$, it sends $\tilde{t}_c, \tilde{t}_d, \tilde{m}$ to $\mathcal{V}$. Otherwise, it aborts.
- $\mathcal{V}$ first verifies that $\tilde{t}_c, \tilde{t}_d \in \left[0, 2^{2\lambda} \cdot \frac{N-1}{4}\right]$, then that
$$\begin{aligned} C'_0 &= C_0^{-2c} \cdot g^{2\cdot \tilde{t}_c} \bmod N^{\zeta+1}, \\ C'_1 &= C_1^{-2c} \cdot (1+N)^{2\tilde{m}} \cdot h^{2\cdot \tilde{t}_c} \bmod N^{\zeta+1}, \\ D'_0 &= D_0^{-2c} \cdot g^{2\cdot \tilde{t}_d} \bmod N^{\zeta+1}, \\ D'_1 &= D_1^{-2c} \cdot (1+N)^{2\tilde{m}} \cdot h_d^{2\cdot \tilde{t}_d} \bmod N^{\zeta+1}, \end{aligned} \tag{4}$$
$\mathcal{V}$ accepts if all checks succeed, and rejects otherwise.

Lemma 3. *The above Σ-protocol provides statistical soundness.* (The proof is available in Appendix B.1).

Lemma 4. *The Σ-protocol Σ_{NY} is statistically special honest-verifier zero-knowledge.* (The proof is given in Appendix B.2).

The Σ-protocol is used to prove plaintext equalities in order to apply the Naor-Yung paradigm in the random oracle model. In order to obtain the simulation-soundness property by applying Theorem 6, we need the following lemma.

Lemma 5. *Under the assumption that factoring N is hard, the above Σ-protocol has quasi-unique responses.* (The proof is given in Appendix B.3).

4.2 IND-CR-CCA secure UPKE

We now upgrade our IND-CR-CPA scheme in the following way. Since we just aim at IND-CR-CCA (rather than IND-CU-CCA) security in this section, we can set $B = (N-1)/4$ and $\zeta \geq 1$.

- $\mathsf{GGen}(1^\lambda, B)$:
  1. Generate two λ-bit safe primes $P = 2p+1, Q = 2q+1$, where p, q are also primes, and set $N = PQ$. Choose an integer $\zeta \geq 1$.
  2. Choose $\mu, \mu_d \hookleftarrow \mathcal{U}(\mathbb{Z}_N)$ at random. Then, compute generators of the subgroup of order $n = \varphi(N)/4 = pq$ by setting $g = \mu^{2N^\zeta} \bmod N^{\zeta+1}$ and $h_d = \mu_d^{2N^\zeta} \bmod N^{\zeta+1}$.
  3. Choose a hash function $H : \{0,1\}^* \to \{0,1\}^\lambda$ that is modeled as a random oracle in the analysis.

  Define $pp = (N, \zeta, g, h_d, H)$ as public parameters.
- $\mathsf{KGen}(pp)$: Sample a random $x \hookleftarrow \mathcal{U}([-2^\lambda B, 2^\lambda B])$. Define the key pair (sk, pk) by setting $sk = (\tau, x)$ and $pk = (\tau, h)$, where $\tau = 0$ is a counter and $h = g^x \bmod N^{\zeta+1}$.
- $\mathsf{Enc}(pk, m)$: To encrypt $m \in \mathbb{Z}_{N^\zeta}$ under the public key $pk = (\tau, h) \in \mathbb{N} \times \mathbb{Z}^*_{N^{\zeta+1}}$, conduct the following steps.
  1. Pick $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and compute
  $$(C_0, C_1) = \left(g^{t_c} \bmod N^{\zeta+1}, (1+N)^m \cdot h^{t_c} \bmod N^{\zeta+1}\right)$$
  $$(D_0, D_1) = \left(g^{t_d} \bmod N^{\zeta+1}, (1+N)^m \cdot h_d^{t_d} \bmod N^{\zeta+1}\right).$$
  2. Using (t_c, t_d, m), generate a NIZK proof π that
  $$(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY},$$
  where $\mathcal{L}_{NY}$ is the language defined in Section 4.1. This proof is of the form
  $$\pi = (C'_0, C'_1, D'_0, D'_1, c, \tilde{t}_c, \tilde{t}_d, \tilde{m}) \in \{0,1\}^\lambda \times \left[0, 2^{2\lambda} \cdot (N-1)/4\right]^2 \times \mathbb{Z}_{N^\zeta},$$

with $c = H\big(\tau, (h, h_d, C_0, C_1, D_0, D_1), (C_0', C_1', D_0', D_1')\big)$, and where (C_0', C_1', D_0', D_1') satisfy (4).

Output the ciphertext $ct = (C_0, C_1, D_0, D_1, \pi)$.

- Dec(sk, ct): Parse ct as $(C_0, C_1, D_0, D_1, \pi)$ and sk as $(\tau, x) \in \mathbb{N} \times \mathbb{Z}$. Return $\perp$ if π is not a verifying NIZK argument that $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$. Otherwise, return

$$m = \mathsf{DLog}\left(C_1^2 \cdot C_0^{-2x} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^{\zeta}.$$

- UpdatePk(pk): To update $pk = (\tau, h) \in \mathbb{N} \times \mathbb{Z}^*_{N^{\zeta+1}}$,
  1. Choose $r \hookleftarrow \mathcal{U}([-B, B])$ and compute $h' = h \cdot g^r \bmod N^{\zeta+1}$.
  2. Choose $k \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and compute

$$up = (U, V) = \left(g^k \bmod N^{\zeta+1}, (1+N)^r \cdot h^k \bmod N^{\zeta+1}\right)$$

  and set $\tau' = \tau + 1$.

  Return $(up = (U, V), pk' = (\tau', h'))$.

- UpdateSk(sk, up): Given $sk = (\tau, x) \in \mathbb{N} \times \mathbb{Z}$ and an update message $(up = (U, V), pk' = (\tau', h'))$, return $\perp$ if $\tau' \neq \tau + 1$. Otherwise, conduct the following steps.
  1. Return $\perp$ if $(U, V) \notin (\mathbb{Z}^*_{N^{\zeta+1}})^2$ or $h \notin \mathbb{Z}^*_{N^{\zeta+1}}$.
  2. Compute

$$r = \mathsf{DLog}\left(V^2 \cdot U^{-2x} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^{\zeta}.$$

  3. Let $\bar{r} = \min(r, N^{\zeta} - r)$ and let $b_r \in \{0, 1\}$ such that

$$\bar{r} = (1 - b_r) \cdot r + b_r \cdot (N^{\zeta} - r) \bmod N^{\zeta}.$$

  Compute $x' = x + (-1)^{b_r} \cdot r$ over $\mathbb{Z}$.
  4. Return $sk' = (\tau + 1, x') \in \mathbb{N} \times \mathbb{Z}$.

Theorem 3. *The above construction provides IND-CR-CCA security assuming that: (i) The DCR assumption holds; (ii) The NIZK proof for $\mathcal{L}_{NY}$ provides simulation-soundness.* (The proof is given in Appendix B.4).

4.3 Arguments of Well-formedness for Update Ciphertexts

To achieve IND-CU-CCA security, we rely on proofs that the updates are well-formed. We obtain these proofs by using a classical Schnorr-like proof in hidden-order groups.

Given g, h of order pq in $\mathbb{Z}^*_{N^{\zeta+1}}$, where $\zeta \geq 2$. Let $B = (N-1)/4$. We need to prove membership of the language

$$\begin{aligned}\mathcal{L}_{WFU} = \big\{\big((h, h') \in (\mathbb{Z}^*_{N^{\zeta+1}})^2,\ up = (U, V)\big) \mid \exists k \in [0, B],\\ r \in [-B, B] \ :\ U = g^k \bmod N^{\zeta+1}\\ \wedge\ V = (1+N)^r \cdot h^k \bmod N^{\zeta+1}\\ \wedge\ (h/h')^2 = g^r \bmod N^{\zeta+1}\big\},\end{aligned}$$

which is the language of well-formed updates. In the proof of soundness, however, we can only guarantee membership of

$$\begin{aligned}\bar{\mathcal{L}}_{WFU} = \big\{\big((h, h') \in (\mathbb{Z}^*_{N^{\zeta+1}})^2,\ up = (U, V)\big) \mid\\ \exists k \in [-2^{2\lambda+1}B, 2^{2\lambda+1}B],\ r \in [-2^{2\lambda+1}B, 2^{2\lambda+1}B] \ :\\ U^2 = g^{2k} \bmod N^{\zeta+1} \ \wedge\ (h/h')^2 = g^{2r} \bmod N^{\zeta+1}\\ \wedge\ V^2 = (1+N)^{2r} \cdot h^{2k} \bmod N^{\zeta+1}\big\},\end{aligned}$$

so that we have a soundness slack as the actual witnesses (k, r) live in $[0, B] \times [-B, B]$.

We assume that $\zeta \geq 2$, so that the interval $[0, 2^{2\lambda}B]$ fits in the message space $\mathbb{Z}_{N^{\zeta}}$ of the encryption scheme.

We define the following Σ-protocol Σ_{WFU} for $(\mathcal{L}_{WFU}, \bar{\mathcal{L}}_{WFU})$:

- $\mathcal{P}$ picks $r', k' \hookleftarrow \left[-2^{2\lambda}B, 2^{2\lambda}B\right]$, and sends

$$\begin{aligned}U' &= g^{2k'} \bmod N^{\zeta+1},\\ V' &= (1+N)^{2r'} \cdot h^{2k'} \bmod N^{\zeta+1},\\ H &= g^{2r'} \bmod N^{\zeta+1}\end{aligned}$$

  to the verifier.
- $\mathcal{V}$ sends a random challenge $c \hookleftarrow \mathcal{U}([0, 2^{\lambda} - 1])$.
- $\mathcal{P}$ computes responses $\tilde{k} = k' + ck$ and $\tilde{r} = r' + cr$ over $\mathbb{Z}$. If $\tilde{k}, \tilde{r} \in \left[-2^{2\lambda}B, 2^{2\lambda}B\right]$, it sends $\tilde{k}, \tilde{r}$ to $\mathcal{V}$, else it aborts.
- $\mathcal{V}$ first verifies that $\tilde{k}, \tilde{r} \in \left[-2^{2\lambda}B, 2^{2\lambda}B\right]$, then that

$$\begin{aligned}U' &= U^{-2c} \cdot g^{2\tilde{k}} \bmod N^{\zeta+1},\\ V' &= V^{-2c} \cdot (1+N)^{2\tilde{r}} \cdot h^{2\tilde{k}} \bmod N^{\zeta+1},\\ H &= (h/h')^{-2c} \cdot g^{2\tilde{r}} \bmod N^{\zeta+1}\end{aligned} \tag{5}$$

  It accepts if all this checks are correct, and rejects otherwise.

The special-soundness property can be proven under the Strong RSA assumption, by adapting a technique from Camenisch and Shoup [14]. One difference is that, here, the encrypted discrete logarithm lives in $[-(N-1)/4, (N-1)/4]$ (instead of $\mathbb{Z}_{\rho}$ for a public prime order $\rho < 2^{\lambda}pq$ in [14, Section 5.2]). Our use of a larger message space $\mathbb{Z}_{N^{\zeta}}$ with $\zeta \geq 2$ allows the proof to go through by adapting techniques from [14] and [26].

Lemma 6. *The above Σ-protocol provides soundness for the language $\bar{\mathcal{L}}_{WFU}$ under the Strong RSA assumption.*

Proof. Let us assume that a prover can come up with accepting transcripts $((U', V', H), c_1, (\tilde{k}_1, \tilde{r}_1)), ((U', V', H), c_2, (\tilde{k}_2, \tilde{r}_2))$. We show that, unless the Strong RSA assumption is false, it must be that $((h, h'), (U, V)) \in \bar{\mathcal{L}}_{WFU}$.

We define $\Delta c = c_1 - c_2$, $\Delta k = \tilde{k}_1 - \tilde{k}_2$, $\Delta r = \tilde{r}_1 - \tilde{r}_2$ with $\Delta k \in [-2^{2\lambda+1}B, 2^{2\lambda+1}B]$, $\Delta r \in \left[-2^{2\lambda+1}B, 2^{2\lambda+1}B\right]$. From the verification equations (5), we have

$$\begin{aligned}U^{2\Delta c} &= g^{2\Delta k} \bmod N^{\zeta+1},\\ V^{2\Delta c} &= (1+N)^{2\Delta r} \cdot h^{2\Delta k} \bmod N^{\zeta+1},\\ (h/h')^{2\Delta c} &= g^{2\Delta r} \bmod N^{\zeta+1}.\end{aligned} \tag{6}$$

Using the same arguments as [14, Theorem 4], we first show that, unless the prover can compute a non-trivial root of $g \in \mathbb{Z}^*_{N^{\zeta+1}}$, Δc

divides both Δr and Δk. Let us first assume that $d = \gcd(\Delta c, \Delta r) < \Delta c$. Then, there exist efficiently computable $\alpha, \beta \in \mathbb{Z}$ such that $\alpha\Delta c + \beta\Delta r = d$ and the last equation of (6) implies

$$g^{2d} = \left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{2\Delta c} \bmod N^{\zeta+1},$$

which in turn yields

$$g^2 = \psi \cdot \left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{2(\Delta c/d)} \bmod N^{\zeta+1}, \tag{7}$$

for some $\psi \in \mathbb{Z}^*_{N^{\zeta+1}}$ such that $\psi^d = 1 \bmod N^{\zeta+1}$, meaning that $\mathrm{ord}(\psi)|d$. Since $\psi \in \mathbb{Z}^*_{N^{\zeta+1}}$, we have $\mathrm{ord}(\psi)|2N^{\zeta}pq$. Then, since we also have $d|\Delta c$ and $\gcd(\Delta c, N^{\zeta}pq) = 1$, this implies that $\mathrm{ord}(\psi) = 2$. Since we cannot have $\psi = -1$ (as -1 is not a square in $\mathbb{Z}^*_{N^{\zeta+1}}$ when $P = Q = 3 \bmod 4$), we have either $\psi = 1$, or a non-trivial factor of $N = PQ$ is revealed by computing $\gcd(\psi + 1, N)$ (because $(\psi + 1)(\psi - 1) = 0 \bmod N^{\zeta+1}$). Unless the factoring assumption (and thus the Strong RSA assumption) is broken, we thus have

$$g^2 = \left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{2(\Delta c/d)} \bmod N^{\zeta+1}. \tag{8}$$

Then, we distinguish three cases. If $\Delta c/d$ is even, we have

$$g = \left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{\Delta c/d} \bmod N^{\zeta+1}. \tag{9}$$

since squaring is a permutation in the subgroup of squares when $P = Q = 3 \bmod 4$, and $\mu \triangleq g^{\alpha} \cdot (h/h')^{\beta} \bmod N^{\zeta+1}$ is then a non-trivial root of g. If $\Delta c/d$ is odd and

$$g \neq \pm\left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{\Delta c/d} \bmod N^{\zeta+1},$$

we obtain a non-trivial factor of $N = PQ$. Finally, if $\Delta c/d$ is odd and

$$g = \pm\left(g^{\alpha} \cdot (h/h')^{\beta}\right)^{\Delta c/d} \bmod N^{\zeta+1}, \tag{10}$$

then $g = \left(\pm g^{\alpha} \cdot (h/h')^{\beta}\right)^{\Delta c/d} \bmod N^{\zeta+1}$ and we also have a root $\mu \triangleq \pm g^{\alpha} \cdot (h/h')^{\beta} \bmod N^{\zeta+1}$ of g.

We can show in the same way that the Strong RSA assumption is broken if Δc does not divide Δk.

We now assume that $\Delta c|\Delta r$ and $\Delta c|\Delta k$. Since $\Delta c < \min(p, q)$, we necessarily have $\gcd(\Delta c, pq) = 1$ and $\gcd(\Delta c, N) = 1$. Then, if we define $\hat{c} = \Delta c^{-1} \bmod N^{\zeta}pq$, the above arguments imply

$$\begin{aligned} U^2 &= g^{2\Delta k\hat{c}} \bmod N^{\zeta+1} = g^{2(\Delta k/\Delta c)} \bmod N^{\zeta+1}, \\ V^2 &= (1+N)^{2\Delta r\cdot\hat{c}} \cdot h^{2\Delta k\hat{c}} \bmod N^{\zeta+1} \\ &= (1+N)^{2(\Delta r/\Delta c)} \cdot h^{2(\Delta k/\Delta c)} \bmod N^{\zeta+1}, \\ (h/h')^2 &= g^{2(\Delta r\cdot\hat{c})} \bmod N^{\zeta+1} \\ &= g^{2(\Delta r/\Delta c)} \bmod N^{\zeta+1}, \end{aligned}$$

which means that $((h, h'), (U, V)) \in \bar{\mathcal{L}}_{WFU}$ since we have $\Delta r/\Delta c \in [-2^{2\lambda+1}B, 2^{2\lambda+1}B]$. □

Lemma 7. *The Σ-protocol Σ_{WFU} satisfies the special honest verifier zero-knowledge property for the language $\mathcal{L}_{WFU}$.*

Proof. In the special honest verifier setting, the simulator is given the challenge c and has to simulate an accepting transcript with the proper distribution for $h', (U, V) \in \mathcal{L}_{WFU}$. We can uniformly sample $\tilde{k}, \tilde{r} \hookleftarrow \mathcal{U}([-2^{2\lambda}B, 2^{2\lambda}B])$ and then set

$$\begin{aligned} U' &= U^{-2c} \cdot g^{2\tilde{k}} \bmod N^{\zeta+1}, \\ V' &= V^{-2c} \cdot (1+N)^{2\tilde{r}} \cdot h^{2\tilde{k}} \bmod N^{\zeta+1}, \\ H &= (h/h')^{-2c} \cdot g^{2\tilde{r}} \bmod N^{\zeta+1} \end{aligned}$$

We then return $(U', V', H, c, \tilde{k}, \tilde{r})$.

This is a valid transcript as it satisfies (5). Moreover, for any true statement $(h', (U, V)) \in \mathcal{L}_{WFU}$, we know that $\exists k \in [0, B]$, $r \in [-B, B]$ such that $U^2 = g^{2k} \bmod N^{\zeta+1}$, $V^2 = (1+N)^{2r} \cdot h^{2k} \bmod N^{\zeta+1}$, $(h/h')^2 = g^{2r} \bmod N^{\zeta+1}$. We thus have:

$$\begin{aligned} U' &= g^{2\tilde{k}-2cr} \bmod N^{\zeta+1}, \\ V' &= (1+N)^{2\tilde{r}-2cr} \cdot h^{2\tilde{k}-2ck} \bmod N^{\zeta+1}, \\ H &= g^{2\tilde{r}-2cr} \bmod N^{\zeta+1} \end{aligned}$$

Using Lemma 2, we know that the distributions of $\tilde{k}, \tilde{r}$ in a real interaction are statistically uniform in $\left[-2^{2\lambda}B, 2^{2\lambda}B\right]$ as $|c \cdot r|, |c \cdot k| < 2^{\lambda} \cdot B$ and $(2^{\lambda} \cdot B)/(2^{2\lambda} \cdot B) = \mathrm{negl}(\lambda)$. Since (U', V', H) is uniquely determined by the statement and the triple $(c, \tilde{k}, \tilde{r})$ in the verification equations (5), our simulated transcript is statistically indistinguishable from a real transcript. □

4.4 IND-CU-CCA-secure UPKE

Our IND-CU-CCA construction goes as follows. The main differences with our IND-CR-CCA construction of Section 4.2 are that: (i) Each update message *up* comes with a non-interactive argument showing that it was properly generated; (ii) secret keys are chosen in a larger interval over the integers. In addition, ciphertexts are computing using a somewhat larger modulus as we need $\zeta \geq 2$ in order to ensure the soundness of our Σ-protocol in Section 4.3.

- $\mathrm{GGen}(1^{\lambda})$:
  1. Choose λ-bit safe primes $P = 2p + 1, Q = 2q + 1$ and set $N = PQ$. Choose an integer $\zeta \geq 2$.
  2. Choose $\mu, \mu_d, \mu'_d \hookleftarrow \mathcal{U}(\mathbb{Z}_N)$ at random. Then, compute generators of the subgroup of order $n = \varphi(N)/4 = pq$ by setting $g = \mu^{2N^{\zeta}} \bmod N^{\zeta+1}$, $h_d = \mu_d^{2N^{\zeta}} \bmod N^{\zeta+1}$, and $h'_d = {\mu'_d}^{2N^{\zeta}} \bmod N^{\zeta+1}$.
  3. Choose hash functions $H, H' : \{0,1\}^* \to \{0,1\}^{\lambda}$ that will be modeled as random oracles in the analysis.

  Define $pp = (N, \zeta, g, h_d, h'_d, H, H')$ as public parameters.
- $\mathrm{KGen}(pp)$: Sample a random $x \hookleftarrow U(\{-2^{\lambda}B, 2^{\lambda}B\})$. Define the key pair (sk, pk) by setting $sk = (\tau, x)$ and $pk = (\tau, h)$ where $\tau = 0$ and $h = g^x \bmod N^{\zeta+1}$.
- $\mathrm{Enc}(pk, m)$: To encrypt $m \in \mathbb{Z}_{N^{\zeta}}$ under $pk = (\tau, h) \in \mathbb{N} \times \mathbb{Z}^*_{N^{\zeta+1}}$, conduct the following steps.
  1. Pick $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and compute

$$\begin{aligned} (C_0, C_1) &= \left(g^{t_c} \bmod N^{\zeta+1}, (1+N)^m \cdot h^{t_c} \bmod N^{\zeta+1}\right) \\ (D_0, D_1) &= \left(g^{t_d} \bmod N^{\zeta+1}, (1+N)^m \cdot h_d^{t_d} \bmod N^{\zeta+1}\right). \end{aligned}$$

	Secret key	Public Key	Ciphertext	Update	Security	ROM	Assumption
Jost *et al.*	256b	512b	512b	512b	IND-CR-CPA	Yes	DDH
Dodis *et al.*	1.3kb	328kb	328kb	419Mb	IND-CR-CPA	No	DDH
This work	4.6kb	6.1kb	12kb	12kb	IND-CR-CPA	No	DCR ($\zeta = 1$)
	4.6kb	6.1kb	66kb	12kb	IND-CR-CCA	Yes	DCR ($\zeta = 1$)
	4.6kb	9.2kb	91kb	18kb	IND-CR-CCA	Yes	DCR ($\zeta = 2$)
	4.6kb	9.2kb	91kb	105kb	IND-CU-CCA	Yes	DCR ($\zeta = 2$)

Table 1: Comparison of key/ciphertext/update sizes for existing UPKE schemes with 128-bit strength security

2. Using witnesses (t_c, t_d, m), generate a NIZK proof π that $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$, where $\mathcal{L}_{NY}$ is the language defined in Section 4.1. This proof π is of the form

$$(c, \tilde{t}_c, \tilde{t}_d, \tilde{m}) \in \{0,1\}^\lambda \times \left[0, 2^{2\lambda} \cdot (N-1)/4\right]^2 \times \mathbb{Z}_{N^\zeta},$$

with $c = H\big(\tau, (h, h_d, C_0, C_1, D_0, D_1), (C'_0, C'_1, D'_0, D'_1)\big)$, and where (C'_0, C'_1, D'_0, D'_1) satisfy (4).

Output the ciphertext $ct = (C_0, C_1, D_0, D_1, \pi)$.

- Dec(sk, ct): Parse ct as $(C_0, C_1, D_0, D_1, \pi)$ and sk as $(\tau, x) \in \mathbb{N} \times \mathbb{Z}$. Return $\perp$ if π is not a verifying NIZK argument that $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$. Otherwise, return

$$m = \mathsf{DLog}\left(C_1^2 \cdot C_0^{-2x} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^\zeta.$$

- UpdatePk(pk): To update $pk = (\tau, h) \in \mathbb{Z}^*_{N^{\zeta+1}}$,

1. Pick $r \hookleftarrow \mathcal{U}(\{-B, \ldots, B\})$, set $h' = h \cdot g^r \bmod N^{\zeta+1}$.
2. Compute $(U_0, V_0, U_1, V_1, \pi)$ as

$$\begin{aligned}(U_0, V_0) &= \left(g^{t_c} \bmod N^{\zeta+1}, (1+N)^r \cdot h^{t_c} \bmod N^{\zeta+1}\right)\\ (U_1, V_1) &= \left(g^{t_d} \bmod N^{\zeta+1}, (1+N)^r \cdot h_d'^{\,t_d} \bmod N^{\zeta+1}\right),\end{aligned}$$

for random $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$, and π a proof that $(h, h'_d, U_0, V_0, U_1, V_1) \in \mathcal{L}_{NY}$ computed using (t_c, t_d, r), as described in Enc. Note however that here the public key h'_d replaces h_d in the Enc algorithm.

3. Using witness (t_c, r), generate a NIZK argument π_{up} that $(h, h', U_0, V_0) \in \mathcal{L}_{WFU}$, where $\mathcal{L}_{WFU}$ is the language of Section 4.3. This argument is of the form $\pi_{up} = (c_{up}, \tilde{k}, \tilde{r}) \in \{0,1\}^\lambda \times \left[-2^{2\lambda}B, 2^{2\lambda}B\right]^2$, with

$$c_{up} = H'\big(\tau, (h, h', U_0, V_0), (U', V', H)\big),$$

and where (U', V', H) satisfy (5).

Return $\big(up = (U_0, V_0, U_1, V_1, \pi, \pi_{up}), pk' = (\tau + 1, h')\big)$.

- UpdateSk(sk, up): Given $sk = (\tau, x) \in \mathbb{N} \times \mathbb{Z}$ and

$$\big(up = (U_0, V_0, U_1, V_1, \pi, \pi_{up}), pk' = (\tau', h')\big),$$

return $\perp$ if $\tau' \neq \tau + 1$. Otherwise,

1. Return $\perp$ if $(U_0, V_0, U_1, V_1) \notin (\mathbb{Z}^*_{N^{\zeta+1}})^4$ or $h' \notin \mathbb{Z}^*_{N^{\zeta+1}}$.
2. Return $\perp$ if $\pi = (c, \tilde{t}_c, \tilde{t}_d, \tilde{m}) \in \{0,1\}^\lambda \times \left[0, 2^{2\lambda} \cdot \frac{N-1}{4}\right]^2 \times \mathbb{Z}_{N^\zeta}$ does not parse properly or if it is not a valid NIZK argument that $(h, h'_d, U_0, V_0, U_1, V_1) \in \mathcal{L}_{NY}$.
3. Return $\perp$ if $\pi_{up} = (c_{up}, \tilde{k}, \tilde{r}) \in \{0,1\}^\lambda \times [-2^{2\lambda}B, 2^{2\lambda}B]^2$ does not parse properly or if it is not a valid NIZK argument that $(h, h', U_0, V_0) \in \mathcal{L}_{WFU}$.
4. Compute

$$r = \mathsf{DLog}\left(V^2 \cdot U^{-2x} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^\zeta.$$

5. Let $\bar{r} = \min(r, N^\zeta - r)$ and let $b_r \in \{0,1\}$ such that

$$\bar{r} = (1 - b_r) \cdot r + b_r \cdot (N^\zeta - r) \bmod N^\zeta.$$

Compute $x' = x + (-1)^{b_r} \cdot r$ over $\mathbb{Z}$.

Return $sk' = (\tau + 1, x') \in \mathbb{N} \times \mathbb{Z}$.

- VerifyUpdate$(h = pk, up, pk')$: Let $up = (U_0, V_0, U_1, V_1, \pi, \pi_{up})$. Return $\perp$ if π is not a valid NIZK proof for the statement $(h, h_d, (U_0, V_0, U_1, V_1)) \in \mathcal{L}_{NY}$. Then, verify that π_{up} is a valid NIZK argument that $(h, pk', U_0, V_0) \in \mathcal{L}_{WFU}$, in which case return 1, else return $\perp$.

We now prove that the scheme provides IND-CU-CCA-security in the ROM assuming that the Strong RSA assumption holds and that the scheme of Section 4.2 provides IND-CR-CCA security.

Theorem 4. *The above construction provides IND-CU-CCA security assuming that: (i) The DCR assumption holds; (ii) The NIZK proof for $\mathcal{L}_{NY}$ provides simulation-soundness; (iii) The NIZK argument for $\mathcal{L}_{WFU}$ provides computational soundness*

Proof. The proof uses with a sequence of games. For each i, W_i denotes the event that the adversary $\mathcal{A}$ outputs 0 in Game$_i$.

Game$_0$: This is the original IND-CU-CCA game where the challenger's bit is $b = 0$.

Game$_1$: We replace the proof π^*_{up} in the final update message $up^* = (U_0^*, V_0^*, U_1^*, V_1^*, \pi^*, \pi^*_{up})$ by a simulated NIZK proof obtained by programming H'. We have $|\Pr[W_1] - \Pr[W_0]| \leq 2^{-\lambda}$ as the distribution of π^*_{up} is statistically unchanged. We note that $\mathcal{A}$ can only see this simulated proof at the very end of the game, *after* having submitted all its update queries. Therefore, $\mathcal{A}$ never gets to generate a proof for $\mathcal{L}_{WFU}$ after having seen this simulated proof.

Game$_2$: In this game, we replace the proof π^* of plaintext equality in the final update $up^* = (U_0^*, V_0^*, U_1^*, V_1^*, \pi^*, \pi^*_{up})$ by a simulated proof, which is obtained by programming the random oracle H. By Lemma 4, we have $|\Pr[W_2] - \Pr[W_1]| \leq 2^{-\lambda}$.

Game$_3$: We now change the (U_1^*, V_1^*) components of up^* to be an encryption of 0 under the public key h_d. Thanks to the standard IND-CPA security of the Elgamal-Paillier PKE scheme, these two games are indistinguishable under the DCR assumption, and we have $|\Pr[W_3] - \Pr[W_2]| \leq \mathbf{Adv}^{\mathrm{DCR}}(\lambda)$.

Game$_4$: We modify the generation of public parameters and now choose h'_d as $h'_d = g^{x'_d} \bmod N^{\zeta+1}$, where $x'_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$. Since h'_d remains statistically uniform in the subgroup of $2N^\zeta$-th

residues, the distribution of pp is statistically close to that of Game_0. We have $|\Pr[W_4] - \Pr[W_3]| \leq 2^{-\lambda}$.

Game_5: We introduce the following check: When the adversary makes an update query $(U_0, V_0, U_1, V_1, \pi, \pi_{up}, pk' = ((i+1), h'))$, the challenger uses the secret key sk_i to decrypt (U_0, V_0) and check that the underlying plaintext $r \in \mathbb{Z}$ (as decrypted at step 4 of UpdateSk) is such that $h' = h_i \cdot g^{(-1)^{b_r} \cdot r} \bmod N^{\zeta+1}$, where $h_i \in \mathbb{Z}^*_{N^{\zeta+1}}$ is part of pk_i. It halts if it is not the case. We denote by E_5 the event that it halts. Game_5 and Game_4 are identical until event E_6 occurs and we have $|\Pr[W_5] - \Pr[W_4]| \leq \Pr[E_5]$. The computational soundness of our Σ-protocol for $\bar{\mathcal{L}}_{WFU}$ (together with the soundness of Fiat-Shamir in the ROM) guarantees that $\Pr[E_5] \leq \mathbf{Adv}^{\mathrm{SRSA}}(\lambda)$. Here, note that the fact that the simulated proof π^*_{up} is revealed only at the very end of the game plays a crucial role and allows relying on the standard soundness rather than on simulation-soundness.

Game_6: This game is identical to Game_5 except that, at each query to $\mathcal{O}_{upd}(\cdot)$, we use x'_d instead of the actual secret sk_i of the current epoch i. Namely, when the adversary makes an update query $(U_0, V_0, U_1, V_1, \pi, \pi_{up}, pk')$ that passes VerifyUpdate, the challenger does no longer use sk_i to decrypt (U_0, V_0) at step 4 of UpdateSk. Instead, it uses x'_d to decrypt (U_1, V_1). Game_6 proceeds identically to Game_5 until the event E_6 that an $\mathcal{O}_{upd}(\cdot)$-query involves pairs (U_0, V_0), (U_1, V_1) that decrypt to distinct values although π verifies. The statistical soundness of π (which is guaranteed in the ROM when Fiat-Shamir is applied to our Σ-protocol in Section 4.1) ensures that $\Pr[E_6] \leq 2^{-\lambda}$. We insist that we do not rely on simulation-soundness here since E_6 occurs before $\mathcal{A}$ is allowed to see up^* (and thus before it gets to see a simulated proof).

Game_7: This final game is identical to the previous game except that the challenger encrypts m_1 instead of m_0. Lemma 8 shows that these two games are indistinguishable under the IND-CR-CCA security of our construction from Section 4.2.

We have now switched the challenge ciphertext to be an encryption of m^*_1 instead of m^*_0, and the claim follows by using the same sequence of hybrid games backwards to go back to the IND-CU-CCA security game where the challenger's bit is $b = 1$. □

Lemma 8. Game_6 *and* Game_7 *are computationally indistinguishable if the scheme of Section 4.2 provides IND-CR-CCA security.*

Proof. Assuming that there exists a PPT adversary $\mathcal{A}$ that can distinguish between Game_6 and Game_7, we build a PPT adversary $\mathcal{B}$ against the IND-CR-CCA security of the construction in Section 4.2. Algorithm $\mathcal{B}$ receives as input public parameters pp and pk_0 from its challenger. It appends h'_d to the public parameters pp, where it computes h'_d as $h'_d = g^{x'_d} \bmod N^{\zeta+1}$ by sampling $x'_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$, and forwards $(pp' \triangleq pp \cup \{h'_d\}, pk_0)$ to $\mathcal{A}$. $\mathcal{B}$ has access to a random oracle H'' while $\mathcal{A}$ has access to two random oracles H, H'. Except at the end of this reduction, $\mathcal{B}$ uses H'' to answer oracle H-queries while H' is honestly simulated.

During stages 2 and 4, when $\mathcal{A}$ makes a decryption query, $\mathcal{B}$ submits the same query to its decryption oracle and relays the result to $\mathcal{A}$. When $\mathcal{A}$ makes an update query $(U_0, V_0, U_1, V_1, \pi, \pi_{up}, pk')$, $\mathcal{B}$ uses its knowledge of x'_d to decrypt (U_1, V_1). Thanks to the two tweaks introduced in Game_5 and Game_6, $\mathcal{B}$ is guaranteed to recover the correct randomness r used by $\mathcal{A}$ to generate its update Then, $\mathcal{B}$ submits this r to its own update oracle in the IND-CR-CCA game.

At stage 3, when $\mathcal{A}$ submits a pair of challenge messages (m^*_0, m^*_1), $\mathcal{B}$ submits the same pair to its IND-CR-CCA challenger. It then relays the latter's challenge ciphertext c^* to $\mathcal{A}$.

Finally, at stage 6 of the IND-CR-CCA game, $\mathcal{B}$ obtains a tuple $(pk^* = (\ell^*, h^*), sk^*, up^*)$, where $up^* = (U^*_0, V^*_0)$. It then generates an encryption (U^*_1, V^*_1) of 0 under key h'_d. Let $h_{\ell'}$ the public key at the last epoch ℓ' before moving to pk^*. $\mathcal{B}$ generates a simulated proof π^* for the statement $(h_{\ell'}, h'_d, U^*_0, V^*_0, U^*_1, V^*_1) \in \mathcal{L}_{NY}$ by programming the random oracle H. Note that H must be programmed on an input which is unpredictable to $\mathcal{A}$ (as it is chosen by $\mathcal{B}$'s challenger), so that a collision on H occurs with negligible probability $\leq Q_H \cdot 2^{-\lambda}$. Finally, $\mathcal{B}$ generates a simulated proof π^*_{up} for the statement $(h_{\ell'}, h^*, U^*_0, V^*_0)$ by programming the random oracle H'. It then sends $\mathcal{A}$ the tuple $(pk^*, sk^*, (U^*_0, V^*_0, U^*_1, V^*_1, \pi^*, \pi^*_{up}))$. When $\mathcal{A}$ halts and outputs $b' \in \{0, 1\}$, $\mathcal{B}$ outputs the same b'.

The above reduction shows $|\Pr[W_7] - \Pr[W_6]| \leq \mathbf{Adv}^{\text{cr-cca}}(\lambda) + Q_H \cdot 2^{-\lambda}$, where $\mathbf{Adv}^{\text{cr-cca}}(\lambda)$ denotes the advantage in the proof of Theorem 3. □

5 IMPLEMENTATION AND PERFORMANCES

The main advantage of our scheme resides in that its parameters make it close to practical use, in terms of key size, ciphertext size, and running time. In this section, we compare the key, ciphertext and updates sizes to the previous UPKE constructed in the standard model by Dodis *et al.* in [21] and to the one constructed in the ROM by Jost *et al.* in [34]. For the sake of completeness, we also present the running times of our "proof of concept" implementation.

5.1 Key/Ciphertext/Update Sizes

Table 1 compares the different key/ciphertext/update sizes achieved by the existing UPKE construction and their security assumptions for a 128-bit strength security. One can remark that we avoid the quadratic blowup suffered by the Dodis *et al.* scheme in the update sizes thanks to DCR. This allows us to construct the first IND-CR-CPA secure UPKE with both reasonable key, ciphertext and update sizes in the standard model.

Jost *et at.*'s scheme remains the most efficient to date. However, its IND-CR-CPA security fully relies on the ROM and it does not extend to provide the stronger security notions that we are able to achieve with DCR.

	Encryption	Decryption	Update
IND-CR-CPA (112 bits)	0.021 s	0.020 s	0.042 s
IND-CR-CPA (128 bits)	0.062 s	0.062 s	0.127 s
IND-CR-CCA (112 bits)	0.045 s	0.042 s	0.085 s
IND-CR-CCA (128 bits)	0.092 s	0.091 s	0.202 s

Table 2: Benchmarks on our implementation of our IND-CR-CPA and IND-CR-CCA schemes

5.2 Running Time

We implemented our IND-CR-CPA and IND-CR-CCA schemes in C/C++ as a proof of concept. Our implementation relies on the

GMP[5] library to handle computations on big integers. One should note that, being only a proof of concept, the implementation is heavily optimizable. No parallelization is used and a few choices were made to ease the implementation and rather harm the performances.

Our benchmarks were made on an Apple M1 CPU with 8 cores running at 3Ghz, under macOS 12. They are presented in table 2. The code was compiled with clang (clang-1205.0.22.9) with the optimization flag -O3. The running time of each function was estimated by taking the mean running time of 1000 evaluations. For the IND-CR-CPA version, encryptions, decryptions and updates sensibly have the same running time. Simply because an update consists of an encryption and a decryption, and that decryptions require the same operations as encryptions.

The first gap appears when introducing NIZKs for Naor-Yung. As updates do not require proofs, they are a bit faster than encryptions. IND-CR-CCA security comes at the cost of losing a factor 20 in efficiency in our benchmarks. We believe that those results are encouraging for future real-world implementations of UPKEs.

ACKNOWLEDGEMENTS

We thank the reviewers for constructive comments. Part of this research was funded by the French ANR ALAMBIC project (ANR-16-CE39-0006). The first author was funded by the Direction Générale de l'Armement (Pôle de Recherche CYBER).

REFERENCES

[1] J. Alwen, S. Coretti, Y. Dodis, and Y. Tselekounis. 2020. Security analysis and improvements for the IETF MLS standard for group messaging. In *Crypto*.
[2] R. Anderson. 1997. Two remarks on public-key cryptology. In *ACM-CCS*. Invited lecture.
[3] N. Barić and B. Pfitzmann. 1997. Collision-free accumulators and fail-stop signature schemes without trees. In *Eurocrypt*.
[4] M. Bellare and S. Miner. 1999. A forward-secure digital signature scheme. In *Crypto*.
[5] M. Bellare and B. Yee. 2003. Forward-security in private-key cryptography.. In *CT-RSA*.
[6] J. Black, P. Rogaway, and T. Shrimpton. 2002. Encryption-Scheme Security in the Presence of Key-Dependent Messages. In *SAC*.
[7] D. Boneh and X. Boyen. 2004. Efficient selective-ID secure identity based encryption without random oracles. In *Eurocrypt*.
[8] D. Boneh, X. Boyen, and E.-J. Goh. 2005. Hierarchical identity based encryption with constant size ciphertext. In *Eurocrypt*.
[9] D. Boneh, S. Halevi, M. Hamburg, and R. Ostrovsky. 2008. Circular-Secure Encryption from Decision Diffie-Hellman. In *Crypto*.
[10] D. Boneh, P. Papakonstantinou, C. Rackoff, Y. Vahlis, and B. Waters. 2008. On The Impossibility of Basing Identity Based Encryption on Trapdoor Permutations. In *FOCS*.
[11] X. Boyen, H. Shacham, E. Shen, and B. Waters. 2006. Forward-Secure Signatures with Untrusted Update. In *ACM-CCS*.
[12] Z. Brakerski and S. Goldwasser. 2010. Circular and Leakage Resilient Public-Key Encryption Under Subgroup Indistinguishability (or: Quadratic Residuosity Strikes Back). In *Crypto*.
[13] Z. Brakerski, A. Lombardi, G. Segev, and V. Vaikunthanathan. 2018. Anonymous IBE, leakage resilience and circular security from new assumptions. In *Eurocrypt*.
[14] J. Camenisch and V. Shoup. 2003. Practical verifiable encryption and decryption of discrete logarithms. In *Crypto*.
[15] R. Canetti, S. Halevi, and J. Katz. 2003. A forward-secure public-key encryption scheme. In *Eurocrypt*.
[16] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. 2012. Bonsai trees, or how to delegate a lattice basis. In *Journal of Cryptology*.
[17] R. Cramer, I. Damgård, and B. Schoenmakers. 1994. Proofs of partial knowledge and simplified design of witness hiding protocols. In *Crypto*.
[18] E. Cronin, S. Jamin, T. Malkin, and P. McDaniel. 2003. On the performance, feasibility, and use of forward-secure signatures. In *ACM-CCS*.
[19] I. Damgård and M. Jurik. 2001. A generalisation, a simplification and some applications of Paillier's probabilistic public-key system. In *PKC*. Springer, 119–136.
[20] Y. Dodis, K. Haralambiev, A. Lopez-Alt, and D. Wichs. 2010. Efficient Public-Key Cryptography in the Presence of Key Leakage. In *Asiacrypt*.
[21] Y. Dodis, H. Karthikeyan, and D. Wichs. 2021. Updatable Public Key Encryption in the Standard Model. In *TCC*. Springer, 254–285.
[22] N. Döttling and S. Garg. 2017. Identity-based encryption from the Diffie-Hellman assumption. In *Crypto*.
[23] S. Faust, M. Kohlweiss, G.-A. Marson, and D. Venturi. 2012. On the non-malleability of the Fiat-Shamir transform. In *Indocrypt*. Springer, 60–79.
[24] A. Fiat and A. Shamir. 1986. How to prove yourself: Practical solutions to identification and signature problems. In *Crypto*.
[25] P.-A. Fouque and D. Pointcheval. 2001. Threshold Cryptosystems Secure against Chosen-Ciphertext Attacks. In *Asiacrypt*.
[26] E. Fujisaki and T. Okamoto. 1997. Statistical Zero Knowledge Protocols to Prove Modular Polynomial Relations. In *Crypto*.
[27] C. Gentry, C. Peikert, and V. Vaikunthanathan. 2008. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*.
[28] C. Gentry and A. Silverberg. 2002. Hierarchical ID-based cryptography. In *Asiacrypt*.
[29] S. Goldwasser, S. Micali, and C. Rackoff. 1989. The knowledge complexity of interactive proof systems. *SIAM J. Comput.* (1989).
[30] J. Groth and A. Sahai. 2008. Efficient Non-interactive Proof Systems for Bilinear Groups. In *Eurocrypt*.
[31] D. Hofheinz. 2013. Circular Chosen-Ciphertext Security with Compact Ciphertexts. In *Eurocrypt*.
[32] J. Horwitz and B. Lynn. 2002. Toward hierarchical identity-based encryption. In *Eurocrypt*.
[33] G. Itkis and L. Reyzin. 2001. Forward-Secure Signatures with Optimal Signing and Verifying. In *Crypto*.
[34] D. Jost, U. Maurer, and M. Mularczyk. 2019. Efficient ratcheting: Almost-optimal guarantees for secure messaging. In *Eurocrypt*.
[35] F. Kitagawa, T. Matsuda, and K. Tanaka. 2019. Simple and efficient KDM-CCA secure public key encryption. In *Asiacrypt*.
[36] T. Malkin, D. Micciancio, and S. Miner. 2002. Efficient Generic Forward-Secure Signatures with an Unbounded Number Of Time Periods. In *Eurocrypt*.
[37] T. Malkin, I. Teranishi, and M. Yung. 2011. Efficient circuit-size independent public key encryption with KDM security. In *Eurocrypt*.
[38] G. Miller. 1976. Riemann's hypothesis and tests for primality. *Journal of computer and system sciences* 13, 3 (1976), 300–317.
[39] M. Naor and M. Yung. 1990. Public-key cryptosystems provably secure against chosen ciphertext attacks. In *STOC*.
[40] P. Paillier. 1999. Public-Key Cryptosystems Based on Composite Degree Residuosity Classes. In *Eurocrypt*.
[41] P. Papakonstantinou, C. Rackoff, and Y. Vahlis. 2012. How powerful are the DDH hard groups?. In *Cryptology ePrint Archive: Report 2012/653*.
[42] D. Pointcheval and J. Stern. 1996. Security Proofs for Signature Schemes. In *Eurocrypt*.
[43] O. Regev. 2005. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*.
[44] A. Sahai. 1999. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security. In *FOCS*.

A COMPLEMENTARY DEFINITIONS: ZERO-KNOWLEDGE PROOFS AND Σ-PROTOCOLS

We recall standard definitions of zero-knowledge proofs [29].

Interactive proof systems. An interactive proof system (IPS) for a language $\mathcal{L}$ is a two-party protocol between a prover $\mathcal{P}$ and a verifier $\mathcal{V}$, where the former wants to convince the latter that a statement x belongs to $\mathcal{L}$. An IPS is *zero-knowledge* when this can be achieved without having $\mathcal{P}$ leak any information beyond the fact that $x \in \mathcal{L}$. This is formalized by requiring the existence of a zero-knowledge simulator $\mathcal{S}$ that produces transcripts indistinguishable from real conversations between $\mathcal{P}$ and $\mathcal{V}$. An IPS provides *soundness* when even an unbounded $\mathcal{P}$ cannot trick $\mathcal{V}$ into accepting a proof of a false statement. When the soundness condition only holds for polynomial-time cheating provers, an IPS is called an *argument*. A zero-knowledge proof/argument system is

[5] GMP https://gmplib.org

non-interactive (NIZK) when a proof/argument consists of a single message from $\mathcal{P}$ to $\mathcal{V}$.

We now recall the definition of Σ-protocols [17].

Definition 10. *Let an NP language $\mathcal{L}$ associated with a relation $\mathcal{R}$. A 3-move interactive proof system $\Pi = (\mathcal{P}, \mathcal{V})$ is a Σ-protocol for $\mathcal{L}$ if it satisfies the following:*

- **3-Move Form:** *$\mathcal{P}$ and $\mathcal{V}$ proceed as follows: (i) $\mathcal{P}$ inputs $(x, w) \in \mathcal{R}$, computes $(a, st) \leftarrow \mathcal{P}(x, w)$ and sends a to $\mathcal{V}$; (ii) $\mathcal{V}$ sends back a random challenge c; (iii) $\mathcal{P}$ sends a response $z = \mathcal{P}(x, w, a, c, st)$ to $\mathcal{V}$; (iv) On input of (a, c, z), $\mathcal{V}$ outputs 1 or 0.*
- **Completeness:** *If $(x, w) \in \mathcal{R}$ and $\mathcal{P}$ honestly computes (a, z) for a challenge c, then $\mathcal{V}(x, (a, c, z))$ outputs 1 with probability $1 - \mathsf{negl}(\lambda)$.*
- **Special soundness:** *There exists a PPT knowledge extractor $\mathcal{E}$ that, for any statement x, on input of two accepting transcripts (a, c, z) and (a, c', z') with $c \neq c'$, outputs a witness w' such that $(x, w') \in \mathcal{R}$.*
- **Special honest-verifier zero-knowledge:** *There is a PPT simulator $\mathcal{S}$ that inputs $x \in \mathcal{L}$ and a challenge $c \in \mathcal{C}$. It outputs $(a, z) \leftarrow \mathcal{S}(x, c)$ such that (a, c, z) is indistinguishable from a real transcript (for $(x, w) \in \mathcal{R}$) with challenge c.*

The special soundness property can be relaxed to hold in the computational sense. In this case, we require that no PPT cheating prover be able to find two transcripts (a, c, z), (a, c', z'), with $c \neq c'$, for which the extractor fails to compute a witness.

In order to prove plaintext equalities via the Naor-Yung paradigm, we require a Σ-protocol satisfying the following property.

Definition 11 ([23],Quasi-unique responses). *Let $\lambda \in \mathbb{N}$ a security parameter. A Σ-protocol $(\mathcal{P}, \mathcal{V})$ has* quasi-unique responses *if, for any PPT algorithm $\mathcal{A}$, it holds that*

$$\Pr[\mathcal{V}(x, a, c, z) = \mathcal{V}(x, a, c, z') = 1 \wedge z \neq z' \mid (x, a, c, z, z') \leftarrow \mathcal{A}(1^\lambda)] = \mathsf{negl}(\lambda).$$

Removing Interaction. In the random oracle model, the Fiat-Shamir heuristic [24] provides an efficient way to build NIZK proofs by collapsing interactive Σ-protocols. The interaction between $\mathcal{P}$ and $\mathcal{V}$ is removed by replacing the latter's challenge with a hash value $H(x, a)$ computed by the prover, where H is a hash function modeled as a random oracle. The resulting NIZK protocol is denoted by $(\mathcal{P}^H, \mathcal{V}^H)$. In the zero-knowledge simulation, the simulator $\mathcal{S}$ is allowed to program the random oracle at arbitrary points. As in [23], we model it as a stateful algorithm that operates in two modes: a mode $(h_i, st) \leftarrow \mathcal{S}(1, st, q_i)$ that handles random oracle queries (typically via lazy sampling); and a second mode $(\pi, st) \leftarrow \mathcal{S}(2, st, x)$ that simulates actual proofs. The different calls to $\mathcal{S}(1, \cdot, \cdot)$ and $\mathcal{S}(2, \cdot, \cdot)$ share a common state st which is updated after each operation.

Definition 12 (NIZK in the ROM). *Let an NP language $\mathcal{L}$ associated with a relation $\mathcal{R}$. Let $(\mathcal{S}_1, \mathcal{S}_2)$ denote oracles such that $\mathcal{S}_1(q_i)$ returns the first output of $(h_i, st) \leftarrow \mathcal{S}(1, st, q_i)$ and $\mathcal{S}_2(x, w)$ returns the first output of $(\pi, st) \leftarrow \mathcal{S}(2, st, x)$ if $(x, w) \in \mathcal{R}$. A non-interactive protocol $(\mathcal{P}^H, \mathcal{V}^H)$ is zero-knowledge for $\mathcal{L}$ in the random oracle model if there exists a PPT simulator $\mathcal{S}$ such that, for any PPT distinguisher $\mathcal{D}$, we have*

$$|\Pr[\mathcal{D}^{H(\cdot), \mathcal{P}^H(\cdot,\cdot)}(1^\lambda) = 1] - \Pr[\mathcal{D}^{\mathcal{S}_1(\cdot), \mathcal{S}_2(\cdot,\cdot)}(1^\lambda) = 1]| = \mathsf{negl}(\lambda),$$

where oracles $\mathcal{P}$ and $\mathcal{S}_2$ both output $\perp$ on input of $(x, w) \notin \mathcal{R}$.

It is known that, in the random oracle model, the Fiat-Shamir transform compiles Σ-protocols into NIZK protocols.

Theorem 5. *Let λ be a security parameter. Consider a Σ-protocol $\Pi = (\mathcal{P}, \mathcal{V})$ for a language $\mathcal{L}$ in NP. Let H be a function that ranges over the challenge space of Π. In the random oracle model, the proof system $\Pi^{FS} = (\mathcal{P}^H, \mathcal{V}^H)$ derived from Π by applying the Fiat-Shamir transform is a NIZK argument system for $\mathcal{L}$ if the special soundness of Π holds (in the statistical or computational sense). Moreover, if the special honest-verifier zero-knowledge property of Π holds in the statistical sense (resp. computational), then Π^{FS} provides statistical (resp. computational) ZK.*

Simulation-soundness. The soundness property of a NIZK proof ensures that no cheating prover $\mathcal{P}^*$ can come up with a convincing proof for a false statement. The notion of *simulation-soundness* [44] requires that soundness be preserved even if a cheating prover is allowed to see simulated proofs (generated by the NIZK simulator) for possibly false statements chosen by $\mathcal{P}^*$. In the random oracle model, we use a definition of simulation-soundness given in [23].

Definition 13 (Unbounded simulation-soundness). *Let $\mathcal{L}$ an NP language. Consider a proof system $(\mathcal{P}^H, \mathcal{V}^H)$ for $\mathcal{L}$ where the zero-knowledge simulator is denoted by $\mathcal{S}$. Denote by $(\mathcal{S}_1, \mathcal{S}_2)$ the oracles such that $\mathcal{S}_1(q_i)$ returns the first output of $(h_i, st) \leftarrow \mathcal{S}(1, st, q_i)$ and $\mathcal{S}_2(x)$ returns the first output of $\mathcal{S}(2, st, x)$ (possibly with $x \notin \mathcal{L}$). A protocol $(\mathcal{P}^H, \mathcal{V}^H)$ is (unbounded) simulation-sound with respect to $\mathcal{S}$ in the random oracle model if, for any PPT adversary $\mathcal{A}$, we have*

$$\Pr[(x^\star, \pi^\star) \leftarrow \mathcal{A}^{\mathcal{S}_1(\cdot), \mathcal{S}_2(\cdot)}(1^\lambda) : (x^\star, \pi^\star) \notin \mathcal{T} \ \wedge \ x^\star \notin \mathcal{L} \ \wedge \ \mathcal{V}^{\mathcal{S}_1}(x^\star, \pi^\star) = 1] = \mathsf{negl}(\lambda),$$

where $\mathcal{T}$ is the list of pairs (x_i, π_i) such that x_i was queried to the simulator and π_i was the latter's response.

Fouque and Pointcheval [25] showed that, in the random oracle model, certain Σ-protocols are turned into simulation-sound NIZK proofs by applying the Fiat-Shamir heuristic. Their proof applies to a restricted family of IPS. In [23], Faust *et al.* gave a more general result that is used in our CCA-secure constructions.

Theorem 6 ([23]). *Consider a non-trivial Σ-protocol $(\mathcal{P}, \mathcal{V})$ for an NP language $\mathcal{L}$, which satisfies the quasi-unique response property of Definition 11. In the random oracle model, the proof system $(\mathcal{P}^H, \mathcal{V}^H)$ derived from $(\mathcal{P}, \mathcal{V})$ via the Fiat-Shamir transform is a simulation-sound NIZK.*

We note that, unlike the result of Fouque and Pointcheval [42], Theorem 6 does not rely on the forking lemma [42] and thus provides tighter concrete security bounds.

B DEFERRED PROOFS

B.1 Proof of Lemma 3

Proof. Let us assume that a prover can come up with two valid transcripts $((C_0', C_1', D_0', D_1), c_1, (\tilde{t}_{c,1}, \tilde{t}_{d,1}, \tilde{m}_1)), ((C_0', C_1', D_0', D_1), c_2, (\tilde{t}_{c,2}, \tilde{t}_{d,2}, \tilde{m}_2))$. We show that $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$.

We define $\Delta c = c_1 - c_2$, $\Delta m = \tilde{m}_1 - \tilde{m}_2 \bmod N^\zeta$, $\Delta t_c = \tilde{t}_{c,1} - \tilde{t}_{c,2}$, $\Delta t_d = \tilde{t}_{d,1} - \tilde{t}_{d,2}$ with $\Delta t_c, \Delta t_d \in [-2^{2\lambda} \cdot \frac{N-1}{4}, 2^{2\lambda} \cdot \frac{N-1}{4}]$. From the verification equations (4), we have

$$\begin{aligned} C_0^{2\Delta c} &= g^{2\cdot\Delta t_c} \bmod N^{\zeta+1}, \\ C_1^{2\Delta c} &= (1+N)^{2\Delta m} \cdot h^{2\cdot\Delta t_c} \bmod N^{\zeta+1}, \\ D_0^{2\Delta c} &= g^{2\cdot\Delta t_d} \bmod N^{\zeta+1}, \\ D_1^{2\Delta c} &= (1+N)^{2\Delta m} \cdot h_d^{2\cdot\Delta t_d} \bmod N^{\zeta+1}, \end{aligned} \tag{11}$$

Since $\Delta c < \min(p, q)$, we necessarily have $\gcd(\Delta c, pq) = 1$ and $\gcd(\Delta c, N) = 1$. Then, if we raise all members of (11) to the power $\hat{c} = \Delta c^{-1} \bmod N^\zeta pq$, we obtain

$$\begin{aligned} C_0^2 &= g^{2\cdot\Delta t_c\cdot\hat{c}} \bmod N^{\zeta+1}, \\ C_1^2 &= (1+N)^{2\Delta m\cdot(\Delta c^{-1} \bmod N^\zeta)} \cdot h^{2\cdot\Delta t_c\cdot\hat{c}} \bmod N^{\zeta+1}, \\ D_0^2 &= g^{2\cdot\Delta t_d\cdot\hat{c}} \bmod N^{\zeta+1}, \\ D_1^2 &= (1+N)^{2\Delta m\cdot(\Delta c^{-1} \bmod N^\zeta)} \cdot h_d^{2\cdot\Delta t_d\cdot\hat{c}} \bmod N^{\zeta+1}, \end{aligned}$$

since $\hat{c} \bmod N^\zeta = \Delta c^{-1} \bmod N^\zeta$. The above equalities show that $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$,[6] as claimed. □

B.2 Proof of Lemma 4

Proof. In the special honest verifier setting, the simulator is given a challenge c and has to produce a properly distributed accepting transcript for $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$. We can uniformly sample $\tilde{t}_c, \tilde{t}_d \hookleftarrow \mathcal{U}([0, 2^{2\lambda} \cdot \frac{N-1}{4}])$, $\tilde{m} \hookleftarrow \mathcal{U}(\mathbb{Z}_{N^\zeta})$ and then set

$$\begin{aligned} C_0' &= C_0^{-2c} \cdot g^{2\cdot\tilde{t}_c} \bmod N^{\zeta+1} \\ C_1' &= C_1^{-2c} \cdot (1+N)^{2\tilde{m}} \cdot h^{2\cdot\tilde{t}_c} \bmod N^{\zeta+1} \\ D_0' &= D_0^{-2c} \cdot g^{2\cdot\tilde{t}_d} \bmod N^{\zeta+1} \\ D_1' &= D_1^{-2c} \cdot (1+N)^{2\tilde{m}} \cdot h_d^{2\cdot\tilde{t}_d} \bmod N^{\zeta+1} \end{aligned}$$

We then return $(C_0', C_1', D_0', D_1', c, \tilde{t}_c, \tilde{t}_d, \tilde{m})$.

This is a valid transcript as it made to satisfy all conditions of the verification step. Since $(h, h_d, C_0, C_1, D_0, D_1) \in \mathcal{L}_{NY}$, we know that $\exists t_c, t_d \in \mathbb{Z}_{pq},\ m \in \mathbb{Z}_{N^\zeta}$ such that

$$\begin{aligned} C_0^2 &= g^{2\cdot t_c} \bmod N^{\zeta+1} \\ C_1^2 &= (1+N)^{2m} \cdot h^{2\cdot t_c} \bmod N^{\zeta+1} \\ D_0^2 &= g^{2\cdot t_d} \bmod N^{\zeta+1} \\ D_1^2 &= (1+N)^{2m} \cdot h_d^{2\cdot t_d} \bmod N^{\zeta+1} \end{aligned}$$

[6] By applying the same arguments as in the proof of Lemma 6, we can show that, unless the cheating prover is able to compute a non-trivial root of $g \in \mathbb{Z}^*_{N^{\zeta+1}}$, Δc divides both Δt_c and Δt_d, so that witnesses t_c and t_d are extractable. However, we do not rely on this property as we only aim at soundness (rather than extractability).

It follows that

$$\begin{aligned} C_0' &= g^{2(\tilde{t}_c - ct_c)} \bmod N^{\zeta+1} \\ C_1' &= (1+N)^{2(\tilde{m}-cm)} \cdot h^{2(\tilde{t}_c - ct_c)} \bmod N^{\zeta+1} \\ D_0' &= g^{2(\tilde{t}_d - ct_d)} \bmod N^{\zeta+1} \\ D_1' &= (1+N)^{2(\tilde{m}-cm)} \cdot h_d^{2(\tilde{t}_d - ct_d)} \bmod N^{\zeta+1} \end{aligned}$$

Since $|c \cdot t_c|, |c \cdot t_d| < 2^\lambda \cdot \frac{N-1}{4}$, Lemma 2 implies

$$(t_c' + c \cdot t_c, t_d' + c \cdot t_d) \approx_s \mathcal{U}([0, 2^{2\lambda} \cdot \frac{N-1}{4}]) \times \mathcal{U}([0, 2^{2\lambda} \cdot \frac{N-1}{4}])$$

and we also have $\{\tilde{m}' + cm \bmod N^\zeta \mid \tilde{m} \hookleftarrow \mathcal{U}(\mathbb{Z}_{N^\zeta})\} \approx_s \mathcal{U}(\mathbb{Z}_{N^\zeta})$. Since (C_0', C_1', D_0', D_1') is uniquely determined by the (true) statement and the challenge-response pair $(c, (\tilde{t}_c, \tilde{t}_d, \tilde{m}))$ in the verification equations, our simulated transcript is statistically close to a real transcript. □

B.3 Proof of Lemma 5

Proof. Towards a contradiction, let as assume that an adversary can find two valid transcripts $(C_0', C_1', D_0', D_1', c, \tilde{t}_{c,0}, \tilde{t}_{d,0}, \tilde{m}_0)$ and $(C_0', C_1', D_0', D_1', c, \tilde{t}_{c,1}, \tilde{t}_{d,1}, \tilde{m}_1)$ for some statement $(h, h_d, C_0, C_1, D_0, D_1)$, with $(\tilde{t}_{c,0}, \tilde{t}_{d,0}, \tilde{m}_0) \neq (\tilde{t}_{c,1}, \tilde{t}_{d,1}, \tilde{m}_1)$.

We first assume that $\tilde{t}_{c,0} \neq \tilde{t}_{c,1}$. By the first equation of the verification equations (4), we have

$$C_0' = C_0^{-2c} \cdot g^{2\cdot\tilde{t}_{c,0}} \bmod N^{\zeta+1} = C_0^{-2c} \cdot g^{2\cdot\tilde{t}_{c,1}} \bmod N^{\zeta+1}$$

so that $g^{2\cdot(\tilde{t}_{c,0} - \tilde{t}_{c,1})} = 1 \bmod N^{\zeta+1}$. Since $\tilde{t}_{c,0} \neq \tilde{t}_{c,1}$, this implies $\tilde{t}_{c,0} - \tilde{t}_{c,1} = 0 \bmod n$. Given a non-trivial multiple of $n = pq$, Miller's algortihm [38] allows computing a non-trivial factor of N with high probability.

The case where $\tilde{t}_{d,0} \neq \tilde{t}_{d,1}$ is similar. We now assume that $\tilde{t}_{d,0} = \tilde{t}_{d,1}$ and $\tilde{t}_{c,0} = \tilde{t}_{c,1}$ but $\tilde{m}_0 \neq \tilde{m}_1$. From the verification equations (4), we obtain

$$\begin{aligned} D_1' &= D_1^{-2c} \cdot (1+N)^{2\tilde{m}_0} \cdot h_d^{2\cdot\tilde{t}_{d,0}} \bmod N^{\zeta+1} \\ &= D_1^{-2c} \cdot (1+N)^{2\tilde{m}_1} \cdot h_d^{2\cdot\tilde{t}_{d,1}} \bmod N^{\zeta+1} \end{aligned}$$

and $(1+N)^{2\tilde{m}_0 - 2\tilde{m}_1} = 1 \bmod N^{\zeta+1}$, which implies that $2\tilde{m}_0 = 2\tilde{m}_1 \bmod N^\zeta$. However, this is impossible if the quasi-uniqueness is broken since $\gcd(2, N) = 1$ and $\tilde{m}_0, \tilde{m}_1 \in \mathbb{Z}_{N^\zeta}$. □

B.4 Proof of Theorem 3

Proof. The proof proceeds with a sequence of games. For each i, we call W_i the event that the adversary $\mathcal{A}$ outputs 0 in Game_i.

Game_0: This is the original IND-CR-CCA game where the challenger's bit is $b = 0$. The challenger initially generates public parameters pp containing (N, ζ, g, h_d). It generates a key pair $(pk_0, sk_0) = (g^x \bmod N^{\zeta+1}, x)$ and gives pk_0 to $\mathcal{A}$. It handles update queries and decryption queries using the real secret key sk_i at any epoch i. At stage 2, $\mathcal{A}$ outputs (m_0^*, m_1^*) and obtains $c^* = (C_0^*, C_1^*, D_0^*, D_1^*, \pi^*)$ of the form

$$\begin{aligned} (C_0^*, C_1^*) &= \left(g^{t_c^*} \bmod N^{\zeta+1}, (1+N)^{m_0^*} \cdot h_\ell^{t_c^*} \bmod N^{\zeta+1}\right), \\ (D_0^*, D_1^*) &= \left(g^{t_d^*} \bmod N^{\zeta+1}, (1+N)^{m_0^*} \cdot h_d^{t_d^*} \bmod N^{\zeta+1}\right) \end{aligned}$$

where $h_\ell = g^{x_\ell} \bmod N^{\zeta+1}$ denotes the public key at epoch ℓ. At stage 6, $\mathcal{A}$ obtains (sk^*, up^*), where

$$up^* = (U^*, V^*) = \left(g^{k^*} \bmod N^{\zeta+1}, (1+N)^{r^*} \cdot h_{\ell'}^{k^*} \bmod N^{\zeta+1}\right),$$

where $r^* \hookleftarrow \mathcal{U}([-B, B])$ and $h_{\ell'} \in \mathbb{Z}_{N^{\zeta+1}}$ is the public key at stage 5. Finally, $\mathcal{A}$ ouputs a bit $b' \in \{0,1\}$ and wins if $b' = 0$. The latter event is called W_0.

Game_1: In this game, we modify the generation of public parameters and now choose h_d as $h_d = g^{x_d} \bmod N^{\zeta+1}$, where $x_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$. Since h_d remains statistically uniform in the subgroup of $2N^\zeta$-th residues, the distribution of pp is statistically close to that of Game_0. We have $|\Pr[W_1] - \Pr[W_0]| \leq 2^{-\lambda}$.

Game_2: We change the decryption oracle. For a decryption query $ct = (C_0, C_1, D_0, D_1, \pi)$ at epoch i, the challenger does no longer use the current secret key sk_i. Instead, it uses x_d to decrypt (D_0, D_1) by computing

$$m = \mathsf{DLog}\left(D_1^2 \cdot D_0^{-2x_d} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^\zeta.$$

Clearly, Game_2 is perfectly indistinguishable from Game_1 until the event that $\mathcal{A}$ queries $\mathcal{O}_{dec}(\cdot)$ on a ciphertext for which (C_0, C_1) and (D_0, D_1) decrypt to distinct messages although π is a valid proof for $\mathcal{L}_{NY}$. However, Lemma 3 and the Fiat-Shamir heuristic ensure that, in the random oracle model, this can only occur with negligible probability: Concretely, for a false statement and any first prover message, a valid response exists for at most one challenge. We have $|\Pr[W_2] - \Pr[W_1]| \leq Q_H \cdot 2^{-\lambda}$, where Q_H is the number of H-queries.

Game_3: We modify the challenge $ct^* = (C_0^*, C_1^*, D_0^*, D_1^*, \pi^*)$. The NIZK proof π^* that $(h_\ell, h_d, C_0^*, C_1^*, D_0^*, D_1^*) \in \mathcal{L}_{NY}$ is now simulated by programming the random oracle H and using the NIZK simulator of Lemma 4. By Lemma 4, $|\Pr[W_3] - \Pr[W_2]| \leq Q_H \cdot 2^{-\lambda}$ as the distribution of π^* is statistically unchanged.

Game_4: We change the distribution of $ct^* = (C_0^*, C_1^*, D_0^*, D_1^*, \pi^*)$. In this game, the challenger computes

$$\begin{aligned}(C_0^*, C_1^*) &= \left(g^{t_c^*} \bmod N^{\zeta+1}, (1+N)^{m_1^*} \cdot h_\ell^{t_c^*} \bmod N^{\zeta+1}\right),\\ (D_0^*, D_1^*) &= \left(g^{t_d^*} \bmod N^{\zeta+1}, (1+N)^{m_0^*} \cdot h_d^{t_d^*} \bmod N^{\zeta+1}\right)\end{aligned}$$

where $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and $h_\ell = g^{x_\ell} \bmod N^{\zeta+1}$ is the public key at epoch ℓ.

The IND-CR-CPA security of the UPKE scheme in Section 3.2 implies – via a reduction that proceeds identically to that in the proof Theorem 2 – the indistinguishability of Game_4 and Game_3 under the DCR assumption. We have $|\Pr[W_4] - \Pr[W_3]| \leq \mathbf{Adv}^{\mathrm{DCR}}(\lambda)$.

Game_5: We change again the decryption oracle. For a decryption query $ct = (C_0, C_1, D_0, D_1, \pi)$ at any epoch i, the challenger does no longer decrypt (D_0, D_1) using x_d. Instead, it comes back to decrypting (C_0, C_1) using the current secret key sk_i, by computing

$$m = \mathsf{DLog}\left(C_1^2 \cdot C_0^{-2 \cdot sk_i} \bmod N^{\zeta+1}\right) \cdot 2^{-1} \bmod N^\zeta.$$

We note that Game_5 is perfectly indistinguishable from Game_4 until $\mathcal{A}$ queries $\mathcal{O}_{dec}(\cdot)$ on a ciphertext ct where (C_0, C_1) and (D_0, D_1) decrypt to distinct messages even though π is a verifying proof for $\mathcal{L}_{NY}$.[7] Here, the simulation-soundness of the NIZK construction in Section 4.1 – which holds in the random oracle model assuming that factoring is hard when we apply the Fiat-Shamir heuristic – ensure that this only occurs with negligible probability if the DCR assumption holds[8]. We have $|\Pr[W_5] - \Pr[W_4]| \leq \mathbf{Adv}^{\text{sim-sound}}(\lambda)$.

Game_6: We change again the distribution of the challenge ciphertext $ct^* = (C_0^*, C_1^*, D_0^*, D_1^*, \pi^*)$. In the challenge phase, the challenger now computes

$$\begin{aligned}(C_0^*, C_1^*) &= \left(g^{t_c^*} \bmod N^{\zeta+1}, (1+N)^{m_1^*} \cdot h_\ell^{t_c^*} \bmod N^{\zeta+1}\right),\\ (D_0^*, D_1^*) &= \left(g^{t_d^*} \bmod N^{\zeta+1}, (1+N)^{m_1^*} \cdot h_d^{t_d^*} \bmod N^{\zeta+1}\right)\end{aligned}$$

where $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$ and $h_\ell = g^{x_\ell} \bmod N^{\zeta+1}$ is the public key at epoch ℓ. The IND-CPA security of the Elgamal-Paillier PKE scheme ensures that Game_6 is indistinguishable from Game_5 under the DCR assumption: i.e., $|\Pr[W_6] - \Pr[W_5]| \leq \mathbf{Adv}^{\mathrm{DCR}}(\lambda)$.

Game_7: In the generation of the challenge $ct^* = (C_0^*, C_1^*, D_0^*, D_1^*, \pi^*)$, we change again the generation of π^*, which is now computed as a real proof using the witnesses $t_c, t_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$. By Lemma 4, we have $|\Pr[W_7] - \Pr[W_6]| \leq Q_H \cdot 2^{-\lambda}$ as the distribution of π^* is statistically close to that of Game_6.

Game_8: Here, we change again the generation of public parameters. Instead of choosing h_d as $h_d = g^{x_d} \bmod N^{\zeta+1}$, where $x_d \hookleftarrow \mathcal{U}(\mathbb{Z}_{(N-1)/4})$, we get back to sampling h_d uniformly in the subgroup of $2N^\zeta$-th residues in $\mathbb{Z}^*_{N^{\zeta+1}}$. The distribution of pp is statistically close to that of Game_7 and we have $|\Pr[W_7] - \Pr[W_6]| \leq 2^{-\lambda}$.

In Game_8, we are exactly in the real game of Definition 3 when the challenger's bit is $b = 1$. Moreover, by combining the above, we find that that Game_0 and Game_8 are indistinguishable under the DCR assumption as the latter implies the simulation-soundness of our NIZK argument for $\mathcal{L}_{NY}$. □

[7] This includes the case of a decryption query on the challenge ciphertext ct^* for an epoch $i > \ell$. We included the epoch number among the inputs of the hash function H to cover this case: If $\mathcal{A}$ can come up with a valid proof π' for $(C_0^*, C_1^*, D_0^*, D_1^*)$ at a later epoch $i > \ell$, we have $\pi' \neq \pi^*$ with probability $1 - 2^{-\lambda}$.

[8] Alain: Clarifier d'où on obtient la statistical soundness (cf mon mail)