Epoch Confidentiality in Updatable Encryption

Jodie Knapp* and Elizabeth A. Quaglia

Information Security Group, Royal Holloway, University of London,
jodie.knapp.2018@rhul.ac.uk, elizabeth.quaglia@rhul.ac.uk

Abstract. In this paper we formalise *public-key* Updatable Encryption (PKUE), a primitive so far studied formally only in the symmetric setting. Defining UE in the public-key setting enables us to establish a new notion of security we call *epoch confidentiality* (EC) which considers the ability of an adversary to distinguish the public keys used in periods known as epochs and in turn reflects the leakage of the time in which a ciphertext was created. We propose a public-key UE construction and prove that it satisfies our new notion of security alongside a notion of ciphertext confidentiality such that efficiency is not affected by moving to the public-key setting.

1 Introduction

In recent years there has been an increase in the outsourcing of encrypted data to a potentially untrusted host. To protect the underlying data and mitigate the security risks of key compromise over a long time, several cryptographic schemes have been proposed that employ a technique called *key-rotation* which enables an entity to move existing ciphertexts from the old to the new key [13]. Trivially, a scheme can update a ciphertext by decrypting and then re-encrypting the underlying plaintext with the updated key. However, when the encrypted data has been outsourced, this is an impractical method. Either the owner must download, re-encrypt and update all ciphertexts themselves, which is computationally inefficient, or they outsource the update by sending the encryption keys to the untrusted host to perform re-encryption, which no longer ensures security [4].

The authors of [4] introduced the *updatable encryption* (UE) primitive to provide a more elegant, non-trivial solution to the above. Instead of re-encrypting a ciphertext from an old to a new key, the data owner instead generates a *token* that enables the host to convert the ciphertext to encryption under the new key (provided it is trusted to delete old tokens and ciphertexts after an update) *without* the need to decrypt.Traditionally, UE schemes have been designed in the symmetric-key setting [4, 9, 5, 6, 14, 10] to convert ciphertexts in a periodic manner marked by set time-intervals known as *epochs* and using encryption keys

* The research of Knapp was supported by the EPSRC and the UK government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/P009301/1).

valid only for their associated epoch. The advantage of equipping an encryption scheme with an update functionality is met with the challenge of modelling security, due to the corruption capabilities of an adversary. As a consequence, the main focus of UE research has been on defining new notions of security and strengthening past notions. The complexities in ensuring security could potentially dominate important factors such as the efficiency, practicality and cost of implementation. For several reasons and with careful consideration, we will formalise a *public-key ciphertext-independent* UE scheme with *probabilistic updates*. As a consequence, we formalise and prove *replayable* chosen-ciphertext security of our construction, which has previously been shown to be the gold-standard level of security for probabilistic UE schemes [11]. We defer the reader to the extended version of this paper for an in-depth overview of the current literature, which comprehensively discusses the trade-offs between such factors and their consequences.

Contributions and Motivation Our first major contribution in this paper is to explicitly define updatable encryption as a public-key primitive (PKUE) in Section 2. Notably, several symmetric UE works [9, 13, 11] have adopted public-key techniques, however, lifting UE to the public-key setting enables us to extend the already rigorously defined security of symmetric UE to capture notions of security only relevant in the context of public-key primitives and building blocks. We note that one can view UE as a special case of the public-key primitive known as *proxy re-encryption* (PRE) [2, 12, 7] and we are motivated to explore UE in the public-key setting for the same applications in which PRE schemes are utilised. However, we emphasise that UE and PRE are fundamentally different primitives in the sense of security guarantees.

Moreover, we identify a gap in previously proposed security modelling which sees us introduce a new notion of security called *epoch confidentiality* in Section 3. Epoch confidentiality can only be modelled in the public-key setting which further illustrates our motivation for formalising public-key UE: guaranteeing security notions specific to public-key building blocks. In more words, we simultaneously achieve confidentiality of both epochs and ciphertexts (UP-IND-EC-RCCA security) by asking an adversary to distinguish the underlying message *and* public key used in encryption that results in the given challenge ciphertext. Our definition of epoch confidentiality is inspired by and can be viewed as achieving key privacy [1] in public-key *updatable* encryption. Key privacy is especially important in UE schemes as the epoch keys have more function than keys in standard PKE schemes. Specifically, epoch keys are required in update token generation and they directly relate to the corresponding epoch in which they are used.

We argue that the notion of epoch confidentiality must be satisfied in any UE scheme in which the data owner cares about the leakage of the age of their encrypted information – e.g., from dating app profiles to individual medical records outsourced for storage. Whilst the leakage of ciphertext age has previously been discussed in [8, 5], not only are their proposed schemes designed for different types of UE schemes, the ciphertext-dependent and deterministic ciphertext-

independent update setting respectively, one critical oversight in both works is to consider the direct relationship the epoch keys have to ciphertext updates. For instance, epoch keys are used to derive the update token, and the inferable information from these is not captured in the ciphertext confidentiality model of both [3, 5]. By contrast, our security model not only asks the adversary to distinguish the underlying message in a UE scheme but also requires the adversary to distinguish the epoch public key used to encrypt the ciphertext. Thus, the notion of epoch confidentiality fully captures the leakage of an epoch in a probabilistic ciphertext-independent update setting by modelling *both* epoch key indistinguishability and ciphertext confidentiality.

In the full version of the paper we present a concrete public-key UE scheme and prove it satisfies epoch confidentiality. Our PKUE scheme is an adaptation of an existing symmetric UE construction [11] explicitly using *updatable* public-key building blocks. In doing so, we can lift the security of the scheme from CPA-security to RCCA-security, thus demonstrating the existence of a public-key UE primitive satisfying (UP-IND-EC-RCCA) security. We conclude our work in the extended version by detailing the security analysis and discussing the efficiency of our construction.

2 Public-Key Updatable Encryption

Notation An updatable encryption scheme is defined by epochs of time e_i from the range of time $i = \{0, \ldots, \mathsf{max}\}$. We denote the current epoch e or use subscript notation e_i for $i \in \mathbb{N}$ if we define multiple epochs at once and in security games the challenge epoch is represented by $\tilde{e}$. To signify epoch keys the notation k_e, k_{e+1} and k^{old}, k^{new} is used interchangeably in this work, depending on whether we require explicit epoch notation or we only need to define consecutive epoch keys (similarly for update tokens Δ).

Traditional symmetric UE is for an owner outsourcing encrypted data over a long period. Time in a UE scheme is formally divided into equal periods known as epochs in which epochs are associated with distinct keys. A ciphertext is updated (re-encrypted) by a potentially untrusted host to the next epoch to provide stronger security by rotating the key used for encryption. Crucially, this update is performed by the host using an update token derived by the date owner, which is formed from the current and preceding epoch keys, such that the host is incapable of learning anything about the encrypted information. Following the discussion in the Introduction, we are motivated to formalise a public-key UE scheme which will be provided below. The key idea is to lift the definition of UE to the public-key setting by generating an epoch key consisting of a public key and a secret key component, and the update token is derived from the past epoch secret key and the current (full) epoch key.

Definition 1 (Updatable Encryption). *A public-key updatable encryption (UE) scheme for message space $\mathcal{MSP}$ consists of a set of polynomial-time algorithms (UE.Setup, UE.KG, UE.TG, UE.Enc, UE.Dec, UE.Upd), defined as follows:*

- $\mathsf{UE.Setup}(1^\lambda) \xrightarrow{\$} pp$: The *owner* runs the *probabilistic* algorithm $\mathsf{UE.setup}$ on input security parameter λ, outputting public parameters pp. Whilst not made explicit, assume throughout that the security parameter (1^λ) is input into the algorithms of the scheme.
- $\mathsf{UE.KG}(pp, e) \xrightarrow{\$} k_e$: The owner runs the probabilistic key-generation algorithm $\mathsf{UE.KG}$ for epoch e on input the public parameters. The output is an epoch key $k_e := (pk_e, sk_e)$ composed of public key (pk_e) and secret-key (sk_e) elements.
- $\mathsf{UE.TG}(sk_e, k_{e+1}) \rightarrow \Delta_{e+1}$: The owner generates the update token by running the *deterministic* algorithm $\mathsf{UE.TG}$ on input the secret key sk_e of epoch key k_e and epoch key k_{e+1} for the proceeding epoch.
- $\mathsf{UE.Enc}(pk_e, m) \xrightarrow{\$} C_e$: The owner runs the probabilistic algorithm $\mathsf{UE.Enc}$ on input a message $m \in \mathcal{MSP}$ and public key pk_e of some epoch e, outputting a ciphertext C_e.
- $\mathsf{UE.Dec}(sk_e, C) \rightarrow \{m', \perp\}$: The owner runs the deterministic algorithm $\mathsf{UE.Dec}$ on input a ciphertext C and secret key sk_e for some epoch e, returning either the message m or abort $\perp$.
- $\mathsf{UE.Upd}(\Delta_{e+1}, C_e) \xrightarrow{\$} C_{e+1}$: The *host* runs the probabilistic algorithm $\mathsf{UE.Upd}$. This is run on input ciphertext C_e for epoch e, and update token Δ_{e+1} for the *next* epoch $(e+1)$, and returns as output the updated ciphertext C_{e+1}.

Informally, the correctness property ensures that fresh encryptions and updated ciphertexts should decrypt to the underlying plaintext, given the appropriate epoch key [13, 11, 5].

Correctness Given security parameter λ, an updatable encryption scheme (UE) formalised in Definition 1 is correct if, for any message $m \in \mathcal{MSP}$ and for any $j \in \{1, \ldots, e\}$, $i \in \{0, \ldots, e\}$ with $e > i$, there exists a negligible function negl such that the following holds,

$$\Pr\left[\begin{array}{l} pp \xleftarrow{\$} \mathsf{UE.Setup}(1^\lambda); k_{e_j} \xleftarrow{\$} \mathsf{UE.KG}(pp, e_j); \\ \Delta_{e_j} \leftarrow \mathsf{UE.TG}(sk_{e_{j-1}}, k_{e_j}); C_{e_i} \xleftarrow{\$} \mathsf{UE.Enc}(pk_{e_i}, m); \\ \{C_{e_j} \leftarrow \mathsf{UE.Upd}(\Delta_{e_j}, C_{e_{j-1}}) : j \in \{i+1, \cdots, \mathsf{max}\}\} : \\ \mathsf{UE.Dec}(sk_e, C_e) = m \end{array}\right] \geq 1 - \mathsf{negl}(\lambda).$$

Before defining a novel security notion for PKUE, we highlight that the extended version of the paper contains the comprehensive details of *lists* and *oracles* required for security modelling in an experiment capturing post-compromise security and ciphertext unlinkability. To be exact, the latter notion of ciphertext unlinkability formalises replayable chosen-ciphertext security ($\mathsf{UP\text{-}IND\text{-}RCCA}$) which assumes an adversary queries updates of *arbitrary* ciphertexts, however, they are incapable of distinguishing updated ciphertexts from the original ciphertext, despite access to prior ciphertexts and update tokens. Notably, modelling RCCA-security is viewed as the benchmark notion of *ciphertext-independent* UE

security given update attacks in an untrusted environment, in line with the recent work of [11].

3 Epoch Confidentiality

In this section, we introduce the notion of an epoch confidential public-key UE primitive. We capture both epoch and ciphertext confidentiality in the UP-IND-EC-RCCA security notion, (Definition 2). We are motivated by the fact that ciphertext-independent UE literature has not yet captured epoch confidentiality, which we argue next is an important security property a UE scheme must satisfy. Namely, in the UE literature the number of key updates on a file indicates the age of the encrypted file (ciphertext). The authors of [3, 5] independently highlighted ciphertext-age leakage to be problematic in real-world scenarios. For instance, [3] considers the setting of dating apps where the number of updates in a UE scheme would reveal how long the person has been a customer which is sensitive information. Indeed, numerous schemes proposed in the literature, such as [9], create a ciphertext expansion as time progresses which results in ciphertext length variance. The authors of [3] demonstrated how *ciphertext length* can be used to trivially infer ciphertext-age in a UE scheme.

A tentative solution given by [3] is to require the length of fresh and updated ciphertexts to be equal, a notion known as *compactness*. However, not only is compactness a strong property to ensure, it does not guarantee there will be no leakage of ciphertext age. Despite the satisfaction of traditional notions of security for UE schemes *and* ciphertext compactness, ciphertext patterns can indicate if the ciphertext is generated by fresh encryption or ciphertext update. Indeed, in [3] a simple example is given in which the last bit of the respective ciphertexts differ, and an adversary can determine whether the ciphertext was derived from an update of a pre-existing ciphertext *or* fresh encryption simply by comparing the last bits of the ciphertexts, thus leaking age information. In [3, 5] the above issues are handled by modelling the computational indistinguishability between fresh ciphertexts and re-encrypted ones to prevent leakage of ciphertext-age. However, without further security modelling an adversary can still infer an epoch and consequently the age of a ciphertext by distinguishing the public component of the epoch key used in encryption, token generation and ciphertext updates.

In more words, our contribution in defining epoch confidentiality not only captures age leakage as in [5, 3] but goes one step further by modelling the indistinguishability of epoch public keys. By definition, the epoch key is designed such that the update token can be derived from the current and proceeding epoch keys. Without specific conditions in security modelling, the corruption of epoch keys and update tokens in challenge-equal epochs enables an adversary to infer information about a version of the challenge ciphertext. In addition to requiring the computational indistinguishability of *ciphertexts* from encryption and update, we necessitate the computational indistinguishability of the *epoch public keys* to provide epoch confidentiality in a given UE scheme.

Security Modelling To achieve epoch confidentiality for a public-key updatable encryption scheme as in Definition 1, an adversary should be unable to distinguish the public-key component of the epoch key under which a ciphertext has been generated. Thus, possession of distinct public keys and a challenge ciphertext should not give an adversary an advantage in determining which public key and therefore which epoch the ciphertext was encrypted under. This approach to modelling security is inspired by and similar in manner to key privacy [1], which is used to define anonymity in public-key encryption schemes (see the Appendices in the extended version of the paper).

To formalise Definition 2, we use the the security experiment given in Figure 1 ($\mathsf{Exp}^{\mathsf{UP\text{-}IND\text{-}EC\text{-}CCA},b}_{\mathsf{UE},\mathcal{A}}(\lambda)$). The intuition is to model an indistinguishability game between the challenger and an adversary $\mathcal{A}$. Initially, the adversary is given two challenge public keys (pk_{e_0}, pk_{e_1}) and $\mathcal{A}$ proceeds to query the oracles detailed in the full version of this paper.

The extended version of the paper contains the details of the lists maintained by the challenger and oracles. We highlight an important list to epoch confidentiality, $\tilde{\mathcal{K}}$ which captures the epochs in which adversary $\mathcal{A}$ receives *challenge public keys* and this list must be checked before responding to all oracle queries, to prevent trivial wins.
In more words, the game in Figure 1 starts by initialising the global state GS. Next, the key-generation algorithm is run twice in order to generate epoch keys $k_{e_0} = (pk_{e_0}, sk_{e_0})$ and $k_{e_1} = (pk_{e_1}, sk_{e_1})$ for distinct epochs of time e_0, e_1. The public keys (pk_{e_0}, pk_{e_1}) are then given to the adversary. The adversary can query oracles $\mathcal{O} = \{\mathcal{O}_{\mathsf{Dec}}, \mathcal{O}_{\mathsf{Upd}}, \mathcal{O}_{\mathsf{Next}}, \mathcal{O}_{\mathsf{Corrupt\text{-}Token}}, \mathcal{O}_{\mathsf{Corrupt\text{-}Key}}\}$ to output valid challenge messages $(m_0, m_1) \in \mathcal{MSP}$ required to be of the same length, alongside some state information s. Subsequently, the challenger encrypts m_b using public key pk_{e_b}, for a pre-determined bit $b \in \{0, 1\}$, sending the challenge ciphertext C to the adversary. Using this challenge ciphertext alongside the state information s and further access to previously detailed oracles, $\mathcal{A}$ guesses the bit b' and succeeds in the game if their guess corresponds to the bit b chosen before the experiment began. More formally,

Definition 2 (UP-IND-EC-RCCA-Security).

A public-key updatable encryption scheme (UE), formalised in Definition 1, satisfies $\mathsf{UP\text{-}IND\text{-}EC\text{-}RCCA}$ security if for any PPT adversary $\mathcal{A}$ there exists a negligible function negl such that

$$\Pr[\mathsf{Exp}^{\mathsf{UP\text{-}IND\text{-}EC\text{-}RCCA}}_{\mathsf{UE},\mathcal{A}}(\lambda) = 1] \leq 1/2 + \mathsf{negl}(\lambda).$$

Preventing Trivial Wins and Ciphertext Updates The winning condition states that the intersection of lists $\mathcal{K}$ and $\mathcal{C}^*$ must be empty which is crucial in preventing the adversary from trivially winning in the security game and if these conditions are not met, then $\mathcal{A}$'s guess is discarded and the output is $\perp$. In more words, the challenge epoch of the experiment cannot belong to the set of epochs in which an update token has been learned or inferred, nor can there exist a single epoch where the adversary knows both the epoch key (public and

$\underline{\mathsf{Exp}^{\mathsf{UP\text{-}IND\text{-}EC\text{-}RCCA},b}_{\mathsf{UE},\mathcal{A}}(\lambda)}$

`Initialise global state`
$\mathsf{GS} \xleftarrow{\$} \mathsf{Init}(1^\lambda)$; $\mathsf{GS} = (pp, k_0, \Delta_0, \mathbf{S}, 0)$
$k_{e_0} \xleftarrow{\$} \mathsf{UE.KG}(pp, e_0)$, $k_{e_1} \xleftarrow{\$} \mathsf{UE.KG}(pp, e_1)$ `such that` $k_{e_0} \neq k_{e_1}$, $(e_0, e_1) \notin \mathcal{K}$
$k_{e_0} := (pk_{e_0}, sk_{e_0}), k_{e_1} := (pk_{e_1}, sk_{e_1})$
`Challenger sends` (pk_{e_0}, pk_{e_1}) `to` $\mathcal{A}$;
$\tilde{\mathcal{K}} \leftarrow \{(e_0, e_1)\} \cap \tilde{\mathcal{K}}$
$(m_0, m_1, s) \xleftarrow{\$} \mathcal{A}^{\mathcal{O}}(pp, pk_{e_0}, pk_{e_1})$
`Some state information` s
if $|m_0| \neq |m_1| \vee \{m_0, m_1\} \notin \mathcal{MSP} \vee (m_0 = m_1)$ **then**
  return $\perp$
else
  $C \xleftarrow{\$} \mathsf{UE.Enc}(pk_{e_b}, m_b)$
  $\mathcal{M}^* \leftarrow \mathcal{M}^* \cup (m_0, m_1)$; $\mathcal{C} \leftarrow \mathcal{C} \cup \{e\}$;$\tilde{e} \leftarrow \{e\}$
$b' \xleftarrow{\$} \mathcal{A}^{\mathcal{O}}(pp, C, s)$
if $(b' = b) \wedge (\mathcal{K} \cap \mathcal{C}^* = \emptyset)$ **then**
  return 1
else
  return $\perp$

Fig. 1. The security game for a UE scheme satisfying UP-IND-EC-RCCA-security, where set $\mathbf{S} = \{\tilde{\mathcal{L}}, \mathcal{M}^*, \mathcal{T}, \mathcal{K}, \tilde{\mathcal{K}}, \mathcal{C}, \mathcal{C}^*\}$ is initially empty, s defines some *state* information output by the adversary and $\mathcal{O} = \{\mathcal{O}_{\mathsf{Dec}}, \mathcal{O}_{\mathsf{Next}}, \mathcal{O}_{\mathsf{Upd}}, \mathcal{O}_{\mathsf{Corrupt\text{-}Token}}, \mathcal{O}_{\mathsf{Corrupt\text{-}Key}}\}$ is the set of oracles an adversary $\mathcal{A}$ calls.

secret key components) and the (updated) challenge-ciphertext [13]. To see this, if the adversary $\mathcal{A}$ corrupts token Δ_{e+1} in an epoch after which $\mathcal{A}$ has obtained the challenge ciphertext $\tilde{C}$ during epoch e, either by inference or via an update, then the adversary is capable of updating the ciphertext into the next epoch $(e+1)$ [11].

Conclusions Our first contribution in this work was re-imagining updatable encryption as a public-key primitive and modelling a public-key equivalent of a prior security notion, which we deem as a necessary security requirement of all probabilistic UE schemes. Our second major contribution was to introduce a new concept of security called epoch confidentiality. In the full version of this work we modified an existing, symmetric UE construction to the public-key setting with no impact on the cost/efficiency of the public-key version of the UE scheme and we use this concrete scheme to show the feasibility of a UE construction satisfying epoch confidentiality.

References

1. M. Bellare, A. Boldyreva, A. Desai, and D. Pointcheval. Key-privacy in public-key encryption. In *Lecture Notes in Computer Science*, volume 2248, pages 566–582. Advances in Cryptology- ASIACRYPT 2001, Springer, 2001.
2. M. Blaze and M. Bleumer, G.and Strauss. Divertible protocols and atomic proxy cryptography. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 127–144. Springer, 1998.
3. D. Boneh, S. Eskandarian, S. Kim, and M. Shih. Improving speed and security in updatable encryption schemes. In S. Moriai and H. Wang, editors, *Advances in Cryptology – ASIACRYPT 2020*, volume 12493. Lecture Notes in Computer Science, Springer, 2020.
4. K. Boneh, D.and Lewi, H. Montgomery, and A. Raghunathan. Key homomorphic prfs and their applications. In R. Canetti and Garay J.A., editors, *Lecture Notes in Computer Science*, volume 8042, pages 410–428. Advances in Cryptology, CRYPTO 2013, Springer, 2013.
5. C. Boyd, Davies. G.T., K. Gjøsteen, and Y. Jiang. Fast and secure updatable encryption†. Technical report, Cryptology ePrint Archive, Report 2019/1457, 2019. https://eprint.iacr.org/2019/1457.pdf, 2020.
6. L. Chen, Y. Li, and Q. Tang. Cca updatable encryption against malicious re-encryption attacks. In *Lecture Notes in Computer Science*, volume 12493, pages 590–620. Advances in Cryptology - ASIACRYPT 2020, Springer, 2020.
7. A. Davidson, A. Deo, E. Lee, and K. Martin. Strong post-compromise secure proxy re-encryption. In *Australasian Conference on Information Security and Privacy-ACISP 2019*, volume 11547, pages 58–77. Lecture Notes in Computer Science, Springer, 2019.
8. E. Eaton, D. Jao, and C. Komlo. Towards post-quantum updatable public-key encryption via supersingular isogenies. *IACR Cryptol. ePrint Arch.*, 2020:1593, 2020.
9. A. Everspaugh, K. Paterson, T. Ristenpart, and S. Scott. Key rotation for authenticated encryption. In J. Katz and H. Shacham, editors, *Lecture Note in Computer Science*, volume 10403, pages 98–129. Advances in Cryptology- CRYPTO 2017, 2017.
10. Y Jiang. The direction of updatable encryption does not matter much. In *Lecture Notes in Computer Science*, volume 12493, pages 529–558. Advances in Cryptology - ASIACRYPT 2020, Springer, 2020.
11. M. Klooß, A. Lehmann, and A. Rupp. (r) cca secure updatable encryption with integrity protection. In Y. Ishai and V. Rijmen, editors, *Lecture Notes in Computer Science*, volume 11476, pages 68–99. Advances in Cryptology- EUROCRYPT 2019, Springer, 2019.
12. E. Lee. Improved security notions for proxy re-encryption to enforce access control. In T. Lange and O. Dunkelman, editors, *Lecture Notes in Computer Science*, volume 11368, pages 66–85. International Conference on Cryptology and Information Security in Latin America- LATINCRYPT 2017, Springer, 2017.
13. A. Lehmann and B. Tackmann. Updatable encryption with post-compromise security. In J. Nielsen and V. Rijmen, editors, *Lecture Notes in Computer Science*, volume 10822, pages 685–716. Advances in Cryptology, EUROCRYPT 2018, Springer, 2018.
14. R. Nishimaki. The direction of updatable encryption does matter. *Cryptology ePrint Archive*, 2021.