General Certificateless Encryption and Timed-Release Encryption

Sherman S.M. Chow[1*], Volker Roth[2], and Eleanor G. Rieffel[2]

[1] Department of Computer Science
Courant Institute of Mathematical Sciences
New York University, NY 10012, USA
schow@cs.nyu.edu
[2] FX Palo Alto Laboratory
3400 Hillview Avenue
Palo Alto, CA 94304, USA
{vroth, rieffel}@fxpal.com

Abstract. Recent non-interactive timed-release encryption (TRE) schemes can be viewed as being supported by a certificateless encryption (CLE) mechanism. However, the security models of CLE and TRE differ and there is no generic transformation that turns a CLE into a TRE. In this paper, we give a generalized model for CLE that is also sufficient to fulfill the requirements of TRE.
Our model is secure against an adversary with adaptive trapdoor extraction capabilities for arbitrary identifiers (instead of selective identifiers), decryption capabilities for arbitrary public keys (as considered in strongly-secure CLE) and partial decryption capabilities (as considered in security-mediated certificateless encryption, or SMCLE). Our model also supports hierarchical identities, which have not been considered formally in paradigms of TRE and CLE.
We propose a concrete scheme under our generalized model and prove it secure without random oracles. Our proposal yields the first strongly-secure SMCLE and the first TRE in the standard model. In addition, our technique of partial decryption is different from the previous approach.

Key words: security-mediated certificateless encryption, timed-release encryption

1 Introduction

The distinguishing feature of identity-based encryption (IBE) (e.g. [7, 12, 17, 18, 29–31, 46]) is that a public key can be derived from any arbitrary string that acts as an identifier (ID). There exists a trusted authority, called a key generation center (KGC), which is responsible for the generation of the ID-based private key on demand. Since the birth of practical constructions of IBE, we see many cryptographic schemes borrowing the idea of IBE for other security goals (e.g. broadcast encryption [9] and oblivious transfer [31]). This paper studies two of them: certificateless encryption (CLE) [2–4, 44, 19, 21, 23, 24, 37, 42] and timed-release encryption (TRE) [6, 14–16, 20, 22, 25, 32, 34]. Our main result provides a transformation from a generalized CLE to a TRE.

CLE is intermediate between IBE and traditional public key encryption (PKE). Traditional PKE requires a certification infrastructure but allows users to create their own public/private key pairs so that their private keys are truly private. Conversely, IBE avoids the need for certificates at the expense of adding a KGC that generates the private keys which means the KGC has the capability to decrypt all messages. CLE combines the advantages of both: no certificates are needed and messages can only decrypted by the recipient. Generally, CLE is constructed by combining IBE and PKE. The existence of the PKE component means that the KGC cannot decrypt messages. Instantaneous revocation is difficult for typical CLE schemes. Security-mediated certificateless encryption (SMCLE) addresses this issue. Here we give the first strongly-secure SMCLE in the standard model. Our scheme also supports hierarchical identities.

[*] This research is done while the author was a research intern of FX Palo Alto Laboratory.

In TRE, the sender encrypts the message under a public key and a time, so knowledge of both the matching private key and a time-dependent trapdoor are necessary for decryption. A time-server is trusted to keep a time-dependent trapdoor confidential until an appointed time, so that the recipient cannot decrypt prior to that time. A feature of modern TRE schemes is that the sender need only interact with the time-server once. Apart from the obvious application of delayed release of information, the need for sending a ciphertext into the future supports other applications which can be classified into two categories: rapid dissemination of information and commitment of confidential information. With TRE, one can send bulky ciphertexts ahead of time. When the information should be made public, a small trapdoor can be widely distributed. Because the size of the time-dependent trapdoor is small compared with the ciphertext (and the text message), this approach avoids the problem of any network impedance at the release time. Applications include release of stock market values, strategic business plans, news agencies timed publications, licensed software updates, scheduled payments, or "casual" applications like internet contests. Commitment of confidential information is needed in many scenarios, such as sealed-bid auction, electronic lotteries, legal will, certified e-mail [34] etc. Text encrypted using TRE can be viewed as a kind of commitment made by the sender; once the ciphertext is sent, the sender cannot change the message that will be received.

Our generalized CLE model, together with our method for converting a generalized CLE to a TRE, provides the first TRE proven secure in the standard model.

1.1 The difficulty of converting between CLE and TRE

A practical TRE requires the system parameter size to be small compared with the number of supported time periods. This is where the relation with IBE comes into the play. By treating the identities as time periods, IBE gives rise to a time-based unlock mechanism (e.g. [7, 40, 41]). However, this approach only supports universal disclosure of encrypted documents since one trapdoor can decrypt all ciphertexts for a specific time. In other words, the inherent key-escrow property of IBE prohibits the encryption for a designated receiver.

Since CLE is an "escrow-free version" of IBE, and both TRE and CLE are a kind of double-encryption, it is natural to think CLE is what we are looking for to realize a TRE. While most recent non-interactive TRE schemes can be seen as converted from a corresponding implicit CLE mechanism, a generic conversion is not known. Despite similarities in syntax and functionality, as has been pointed out in [14], a generic transformation from CLE to TRE is unlikely to be provable secure. Difficulty in reducing the confidentiality of TRE to that of CLE arises when the adversary is a "curious" time-server. In CLE, each user is determined by a combination of an identity and a public key, which means an identity is only associated to a certain public key. In CLE, a curious KGC is not allowed to replace the public key associated with an identifier (otherwise, decryption of the ciphertext will be trivial since it holds both pieces of secrets). On the other hand, a time identifier is never bound to any public key in TRE, which means that the public key associated with a time identifier can be replaced. There is no way to simulate this implicit public key replacement when the CLE is viewed as a black box. Section 2.2 provides four examples of CLE [4, 36, 42, 44] which cannot be trivially extended to TRE.

1.2 Our Contributions

The generalized model for CLE given here overcomes the difficulties described in [14] and has sufficient power to fulfill the requirements of TRE. Our model is secure against an adversary with adaptive trapdoor extraction capabilities for arbitrary identifiers (instead of selective identifiers, e.g. [7, 42]), decryption capabilities for arbitrary public keys (as considered in strongly-secure CLE [24]) and partial decryption capabilities (as considered in security-mediated certificateless encryption, or SMCLE [21]). Our model also supports hierarchical identities which have not been considered formally for CLE and TRE.

We propose a concrete construction under our generalized model. All existing concrete TRE schemes [6, 14–16, 20, 22, 25, 32, 34] and the only concrete SMCLE scheme [21] are proven in the random oracle model. It is true that the generic construction of SMCLE [21] can be instantiated by an IBE and a PKE without random oracles, nevertheless, the resulting scheme is not strongly-secure. Our proposal yields the first strongly-secure SMCLE and the first TRE in the standard model. Moreover, our technique of partial decryption is different from that in [21].

2 Related Work

2.1 Timed-Release Encryption

The concept of timed-release cryptographic protocols was suggested by May [39] in 1993, and further studied by many researchers including [5, 8, 26, 27]. Early TRE schemes require interaction with the time-server. Rivest *et al.*'s idea [43] requires senders to reveal their identities and the messages' release-time in their interactions with the server. In Di Crescenzo *et al.*'s scheme [22], the job of interacting with the time-server is moved from the sender to the receiver since a "conditional oblivious transfer protocol" will be executed between the server and the receiver. Such a protocol ensures that if the release-time is greater than the current time (the condition), the receiver learns nothing (obliviousness). However, this protocol [22] is computationally intensive and thus vulnerable to denial-of-service attacks.

The first attempt to construct a non-interactive and user-anonymous TRE was made in [6]. A concrete construction is provided, but not supported by a formal security model and security properties are only argued for heuristically. The formal security model of message confidentiality is later considered independently by Cheon *et al.* [20] and Cathalo-Libert-Quisquater [14]. The former focuses on authenticated TRE. The latter claims to have a stronger model than the implicit non-authenticated version of [20]. Cathalo-Libert-Quisquater [14] also formalizes the release-time confidentiality, but not recipient-anonymity. The recovery of past time-dependent trapdoors from a current trapdoor is studied in [16] and [41], which employs a hash chain and a tree structure [13] respectively.

A special class of TRE scheme supports pre-open capability: the sender can help the recipient to decrypt the ciphertext early by publishing a pre-open key. Since the pre-open key is held by the sender, by manipulating the pre-open key, the sender might be able to control somehow what message is decrypted. Using a TRE with pre-open capability as a way to commit confidential information requires the TRE scheme to be binding (see Appendix C). The study of the pre-open capability was initiated in [34] and improved by [25].

Recently, Chalkias *et al.* proposed an efficient TRE scheme [15]. They claim their scheme is the most computationally efficient one for unknown recipients. However, we show in Appendix E that the confidentiality of their scheme can be broken by a curious time-server. A plausible fix makes the decryption algorithm of their scheme less efficient and lessens the purported comparative advantage.

The time-lock puzzle approach [43] provides a way to realize TRE without a trusted server: delayed release is obtained by requiring the recipient to invest significant computational effort to solve a difficult problem. However, not only is this approach computationally expensive, but the release-time is not precisely controllable.

2.2 Certificateless Encryption

Certificateless cryptography was suggested by Al-Riyami and Paterson [2] in 2003. We need a basic understanding of the security model to understand the contribution of different proposals. An extensive survey of CLE including various security models can be found in [23]. Two types of adversaries are considered in certificateless cryptography. A Type-I adversary models coalitions of rogue users without the master secret. Due to the lack of a certificate, the adversary is allowed to replace the public keys of users at will. A Type-II adversary models a curious KGC who has the master key but cannot replace the public keys of any users. In Al-Riyami and Paterson's security model for the encryption [2], a Type-I adversary can ask for the decryption of a ciphertext under a replaced public key. Schemes secure against such attacks are called "strongly-secure" [24], and the oracle is termed a "strong decryption oracle". A weaker type of adversary, termed Type-I$^-$, can only obtain a correct plaintext if the ciphertext is submitted along with the corresponding private key.

The first CLE scheme by Al-Riyami and Paterson [2] is secure against both Type-I and Type-II adversary in the random oracle model. Many generic constructions of CLE from IBE and PKE exist, some later shown to be insecure [28, 37, 42], while others [19, 37] rely on the random oracle heuristics. The authors later proposed a more efficient CLE scheme in [3], which has been shown to be insecure [19, 48]. It is believed [36, 38, 42] that [38] gives the first CLE in the standard model. However, it is possible to instantiate a prior generic construction in [21] with a PKE and an IBE in the standard model to obtain a secure CLE without random oracles. Both [38] and the instantiation of [21] are only secure against Type-I$^-$ attacks. Based on [29], a selective-ID secure CLE without random oracles is proposed in [42]. This scheme cannot be trivially extended to a TRE since the user's public key is dependent on the identity, but a user's public key is never coupled with a single time-identifier in TRE. Recently, the first strongly-secure CLE secure against Type-I adversaries in the standard model is proposed in [24].

Al-Riyami and Paterson scheme is also the basis for the hierarchical CLE described in [2]. However, neither a security model nor a security proof are given for this hierarchical extension. We are not aware of any literature with formal work on hierarchical CLE, particularly none proven secure in the standard model.

A CLE that does not use pairings is proposed in [4]. However, the reduction used in the security proof does not hold up if the public key associated with the challenge ciphertext can be replaced. Another CLE proposal without pairing [36] uses similar ideas. No formal evidence was provided to show prove the scheme secure under public key replacement, but this limitation was recently removed by [44]. To replace the pairing, these schemes make part of the user's public key dependent on the identity-specific trapdoor given by the KGC, which means TRE cannot be obtained trivially from these constructions.

Security-mediated certificateless encryption (SMCLE), introduced by Chow, Boyd and González Nieto [21], adds a security-mediator (SEM) who performs partial decryption for the user by request. This idea gives another variant for the decryption queries in the CLE paradigm: the adversary can ask for partial decryption results under either the SEM trapdoor generated by the KGC or the user private key. Intuitively, the notion of SMCLE is more general than that of CLE since two partial decryption algorithms can always be combined into a single one, but the converse is not necessary true (see Section 3.4). A concrete construction in the random oracle model and a generic construction in the standard model are proposed in [21]. Prior to our work, no strongly-secure SMCLE existed that had been proven secure in the standard model.

3 General Security-Mediated Certificateless Encryption

We propose a new definition of the (security-mediated) certificateless encryption. We will also highlight the relationship between our definition and existing definitions.

3.1 Notation

We use an ID-vector $\overrightarrow{ID} = (\mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_L)$ to denote a hierarchy of identifiers $(\mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_L)$. The length of $\overrightarrow{ID}$ is denoted by $|\overrightarrow{ID}| = L$. Let $\overrightarrow{ID}||\mathsf{ID}_r$ denote the vector $(\mathsf{ID}_1, \mathsf{ID}_2, \cdots, \mathsf{ID}_L, \mathsf{ID}_r)$ of length $|\overrightarrow{ID}|+1$. We say that $\overrightarrow{ID}$ is a prefix of $\overrightarrow{ID'}$ if $|\overrightarrow{ID}| \leq |\overrightarrow{ID'}|$ and $\mathsf{ID}_i = \mathsf{ID}'_i$ for all $1 \leq i \leq |\overrightarrow{ID}|$. We use $\emptyset$ to denote an empty ID-vector where $|\emptyset| = 0$ and $\emptyset||\mathsf{ID}_r = \mathsf{ID}_r$. Finally, we use the notation $(\{0,1\}^n)^{\leq h}$ to denote the set of vectors of length less than or equal to h, where each component is a n-bit long bit-string.

3.2 Syntax

Extending the definition of a 1-level SMCLE scheme [21], we define an h-level SMCLE scheme as follows.

Definition 1. *An h-level SMCLE scheme for identifiers of length n (where h and n are polynomially-bounded functions) is defined by the following sextuple of* PPT *algorithms:*

– Setup *(run by the server) is a probabilistic algorithm which takes a security parameter 1^λ, outputs a master secret key* Msk*, which can also be denoted as $d_\emptyset$, and the global parameters* Pub*. We assume that λ, $h = h(\lambda)$ and $n = n(\lambda)$ are implicit in* Pub *and all other algorithms take* Pub *implicitly as an input.*

– $\mathsf{Extract}$ *(run by the server or any one who hold a trapdoor) is a possibly probabilistic algorithm which takes a trapdoor* $d_{\overrightarrow{ID}}$ *corresponding to an* h*-level identity* $\overrightarrow{ID} \in (\{0,1\}^n)^{\leq h}$ *, and a string* $\mathsf{ID}_r \in \{0,1\}^n$*, outputs a trapdoor key* $d_{\overrightarrow{ID}||\mathsf{ID}_r}$ *associated with the ID-vector* $\overrightarrow{ID}||\mathsf{ID}_r$*. The master secret key* Msk *is a trapdoor corresponding to a* 0*-level identity.*
– KeyGen *(run by a user) is a probabilistic algorithm which generates a public/private key pair* $(\mathsf{pk}_u, \mathsf{sk}_u)$*.*
– Enc *(run by a sender) is a probabilistic algorithm which takes a message* m *from some implicit message space, an identifier* $\overrightarrow{ID} \in (\{0,1\}^n)^{\leq h}$*, and the receiver's public key* pk_u *as input, returns a ciphertext* C*.*
– Dec^S *(run by the one who hold the trapdoor, either a SEM in SMCLE or a receiver in CLE) is a possibly probabilistic algorithm which takes a ciphertext* C *and the trapdoor key* $d_{\overrightarrow{ID}}$ *as input, returns either a token* D *which can be seen as a partial decryption result of* C*, or an invalid flag* $\perp$ *(which is not in the message space).*
– Dec^U *(run by a receiver) is a possibly probabilistic algorithm which takes the ciphertext* C*, the receiver's private key* sk_u *and a token* D *as input, returns either the plaintext, an invalid flag* $\perp_D$ *denoting* D *is an invalid token, or an invalid flag* $\perp_C$ *denoting the ciphertext is invalid.*

For correctness, we require that $\mathsf{Dec}^U(C, \mathsf{sk}, \mathsf{Dec}^S(C, \mathsf{Extract}(\mathsf{Msk}, \overrightarrow{ID}))) = m$ *for all* $\lambda \in \mathbb{N}$*, all* $(\mathsf{Pub}, \mathsf{Msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk}) \xleftarrow{\$} \mathsf{KeyGen}$*, all message* m*, all ID-vector* $\overrightarrow{ID}$ *in* $(\{0,1\}^n)^{\leq h}$ *and all* $C \xleftarrow{\$} \mathsf{Enc}(m, \overrightarrow{ID}, \mathsf{pk})$*.*

3.3 Security

Each adversary has access to the following oracles:

1. An $\mathsf{ExtractO}$ oracle that takes an ID-vector $\overrightarrow{ID} \in (\{0,1\}^n)^{\leq h}$ as input and returns its trapdoor $d_{\overrightarrow{ID}}$.
2. A DecO^S oracle that takes a ciphertext C and an ID-vector $\overrightarrow{ID}$, and outputs $\mathsf{Dec}^S(C, d_{\overrightarrow{ID}})$. Note that C may or may not be encrypted under $\overrightarrow{ID}$.
3. A DecO^U oracle that takes a ciphertext C, a public key pk and a token D, and outputs $\mathsf{Dec}^U(C, \mathsf{sk}, D)$ where sk is the secret key that matches pk.
4. A DecO oracle that takes a ciphertext C, an ID-vector $\overrightarrow{ID}$, and a public key pk, and outputs $\mathsf{Dec}^U(C, \mathsf{sk}, D)$ where sk is the secret key that matches pk and $D = \mathsf{Dec}^S(C, d_{\overrightarrow{ID}})$. Note that C may or may not be encrypted under $\overrightarrow{ID}$ and pk.

Following common practice, we consider the two kinds of adversaries.

1. A Type-I adversary that models any coalition of rogue users, and who aims to break the confidentiality of another user's ciphertext.
2. A Type-II adversary that models a curious KGC, who aims to break the confidentiality of an user's ciphertext[3].

We use the common security model in which the adversary plays a two-phased game against a challenger. The game is modeled by the experiment below, for $\mathsf{X} \in \{\mathrm{I}, \mathrm{II}\}$, denoting whether an PPT adversary $\mathcal{A} = (\mathcal{A}_{\mathsf{find}}, \mathcal{A}_{\mathsf{guess}})$ is of Type-I or Type-II. The allowed oracle queries $\mathcal{O}$ and the auxiliary information Aux depends on X.

Definition 2. Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA-X}}(\lambda)$

$$(\mathsf{Pub}, \mathsf{Msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$$
$$(m_0, m_1, \mathsf{pk}^*, \overrightarrow{ID}^*, \mathsf{state}) \xleftarrow{\$} \mathcal{A}_{\mathsf{find}}^{\mathcal{O}}(\mathsf{Pub}, \mathsf{Aux})$$
$$b \xleftarrow{\$} \{0,1\},\ C^* \xleftarrow{\$} \mathsf{Enc}(m_b, \overrightarrow{ID}^*, \mathsf{pk}^*)$$
$$b' \xleftarrow{\$} \mathcal{A}_{\mathsf{guess}}^{\mathcal{O}}(C^*, \mathsf{state})$$
If $(|m_0| \neq |m_1|) \vee (b \neq b')$ *then return 0 else return 1*

[3] We do not explicitly consider a rogue SEM since this type of adversary is weaker than the Type-II adversary.

where $\mathcal{O}$ refers to a set of four oracles $\mathsf{ExtractO}(\cdot), \mathsf{DecO}^S(\cdot,\cdot), \mathsf{DecO}^U(\cdot,\cdot,\cdot), \mathsf{DecO}(\cdot,\cdot,\cdot)$.

Those variables marked with * are basically about the challenge of the adversary. The adversary chooses a public key pk^* and an ID-vector $\overrightarrow{ID}^*$ to be challenged with, and the challenger returns C^* to the adversary as the challenge ciphertext. The two definitions below basically prohibit the adversary from trivially cheating by using the oracles to query for the answer to (parts of) the challenge.

Definition 3. *A hierarchical security-mediated certificateless encryption scheme is (t, q_E, q_D, ϵ) IND-CCA secure against a Type-I adversary if $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA-I}}(\lambda) = 1] - \frac{1}{2}| \le \epsilon$ for all t-time adversary $\mathcal{A}$ making at most q_E extraction queries and q_D decryption queries (of any type), subjects to the following constraints:*

1. $\mathsf{Aux} = \emptyset$*, i.e. no auxiliary information is given to the adversary.*
2. *No* $\mathsf{ExtractO}(\overrightarrow{ID}')$ *query throughout the game, where $\overrightarrow{ID}'$ is a prefix of $\overrightarrow{ID}^*$.*
3. *No* $\mathsf{DecO}^S(C^*, \overrightarrow{ID}^*)$ *query throughout the game.*
4. *No* $\mathsf{DecO}(C^*, \overrightarrow{ID}^*, \mathsf{pk}^*)$ *query throughout the game.*

Definition 4. *A hierarchical security-mediated certificateless encryption scheme is (t, q_K, q_D, ϵ) IND-CCA secure against a Type-II adversary if $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA-II}}(\lambda) = 1] - \frac{1}{2}| \le \epsilon$ for all t-time adversary $\mathcal{A}$ making at most q_D decryption queries (of any type), subjects to the following conditions:*

1. $\mathsf{Aux} = (\mathsf{Msk}, \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\})$*, i.e. the master secret key and a set of challenge public key* pk^* *is given to the adversary.*
2. $\mathsf{pk}^* \in \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\}$*, i.e. the challenge public key must be among the set given by the challenger.*
3. *No* $\mathsf{DecO}^U(C^*, \mathsf{pk}^*, D)$ *query throughout the game, where D is outputted by the algorithm* $\mathsf{Dec}^S(C^*, d_{\overrightarrow{ID}^*})$.
4. *No* $\mathsf{DecO}(C^*, \overrightarrow{ID}^*, \mathsf{pk}^*)$ *query throughout the game.*

Since Msk is given to the adversary, and it is natural to assume the adversary to know the secret key corresponding to any adversarially-chosen public key; the challenge public key must be in the set given by the challenger. Nevertheless, our definition places no restriction on the public key supplied to the decryption oracle, i.e. the decryption oracle should work even if the public key is adversarially chosen and the corresponding private key is not supplied. It is easy to weaken the strong decryption oracle to one corresponding to the Type-I$^-$ attack by placing the below restriction in Definition 3.

5. (Type-II$^-$) No $\mathsf{DecO}(C, \overrightarrow{ID}, \mathsf{pk})$ query throughout the game where $\mathsf{pk} \notin \{\mathsf{pk}_1^*, \cdots \mathsf{pk}_{q_K}^*\}$, unless the corresponding private key sk is supplied when the DecO query is made.

Definition 3 can also be easily modified to Type-I$^-$ to give a similar weakened definition.

3.4 Discussions on Our Choices for Definition

In addition to the formalisms, we explain the intuitions behind the choices made in formulating our definition.

User key generation. In order to support more general applications like TRE, we also need to generalize our syntax describing the interface of the algorithms. A subtle change in our definition of algorithm description is that the user key generation algorithm KeyGen only takes the system parameter as input but *not* the identifier. In particular, there exists CLE schemes [4, 36, 42, 44] which the inclusion of the identifier or the trapdoor for an identifier is *essential* for the generation of the user public key. In the latter case, KeyGen can only be executed after $\mathsf{Extract}$. A straightforward adaption of these classes of CLE results in inefficient TRE in which the size of the user public key grows linearly with the total number of supported time periods.

Simplification of Type-I adversary. In most existing models for 1-level CLE (e.g. [24]), $\mathsf{ExtractO}$ query of $\overrightarrow{ID}^*$ is allowed; but if such a query is issued, the challenge public key pk^* can no longer chosen by the adversary. In our discussion, we try to separate this from Type-I model and consider this type of adversarial behavior ($\mathsf{ExtractO}$ query on $(\overrightarrow{ID}')$ where $\overrightarrow{ID}'$ is a prefix of $\overrightarrow{ID}^*$) as a weaker variant of, and hence covered by, a Type-II adversary. It is true that our resulting definition for Type-I adversary is weaker [23, Section 2.3.5]. However, the "missing part" will not be omitted from the security requirement since it is unreasonable to define a CLE without considering Type-II adversary. Indeed, this simplification has already been justified and adopted [33, Section 2.3].

Implicit public key replacement. In our generalization of CLE, we "remove" the oracle for replacing the public key corresponding to an identifier, which is present in the existing model for CLE. This may make a difference in the following.

1. The adversary's choice of the victim user it wishes to be challenged with,
2. The choice of user in decryption oracle queries.

Our model still allows the adversary to choose which identifier/public key it wants to attack. For decryption queries, the adversary can just supply different combination of identifier and public key to the DecO^S and DecO^U oracles. In this way, implicitly replacement is done. In other words, when compared with the original model [2], the security model is not weakened, but generalized to cover applications of CLE such as TRE.

Reason for "removing" public key replacement oracle. In the traditional definition of CLE [2], public key replacement oracle is defined upon the fact that an identifier is always bound to a particular user. Replacing a particular user's public key means the public key associated with a particular identifier should be changed. In TRE, and other related paradigms such as cryptographic workflow [1], identifiers correspond to different policies governing the decryption. It is entirely possible that a single identifier is "shared" among more than one user. Hence we decide to remove the public key replacement oracle from the definition, resulting a model free from the concept of "user = identifier".

Alternative definition of public key replacement. It is possible to give another definition supporting TRE (and cryptographic workflow [1]) by allowing a "restricted" public key replacement, such that a public key "associated" with an identifier can be replaced by a public key associated with another identifier, but not an arbitrary one supplied by the user. Again, this definition makes the model still leads to the concept of an identifier is belonged to a single user. Moreover, this definition may make the treatment of strong decryption oracle more complicated. The idea of restricted replacement among a fixed set of public keys does not naturally correspond to decryption oracle under adversarially chosen public key.

SMCLE is more general than plain CLE. Having two separated decryption oracles in the SMCLE model gives a more general notion than CLE. This can be justified as follows:

1. Partial decryption result cannot be made available in the CLE model. There exists CLE schemes which are not secure when the adversary is given accesses of a partial decryption oracle [21].
2. Since the decryption oracle is separated into two, the SMCLE model does not have the notion of a "full" private key which is present in previous CLE models (a full private key is a single secret for the complete decryption of the ciphertext). On the ground that separated secrets can always be concatenated into a single full one, this simplification (of private key) has already been adopted in more recent models [33].

Difference with the previous SMCLE definition. In our user decryption oracle, different invalid flags will be returned by DecO^U to distinguish the case that the token from the SEM is invalid or the ciphertext is invalid. This is not captured by the original SMCLE model in [21]. We remark that it is possible to incorporate such feature into the concrete scheme in [21] by an interactive proof-of-knowledge, which can be easily turn to non-interactive assuming random oracle.

User decryption oracle in SMCLE. One of the restrictions for excluding trivial attack in our Type-II adversary model disallows the challenge ciphertext C^* to be decrypted by the decryption oracle under the challenge public key and a token D outputted by the algorithm (not the oracle) $\mathsf{Dec}^S(C^*, \mathsf{ID}^*)$, where ID^* is the challenge identifier. This restriction requires the ability to check if a token D is a *valid* token corresponding to a ciphertext and an identifier, which is ensured by our new SMCLE definition.

From the first glance, our security definition is tightly coupled with the ability to check the token. However, we think that it is natural for the user to be able to perform such a test (which is especially important if the user need to pay for each SEM decryption). Even there is no explicit testing algorithm, it maybe possible that the challenger can setup the system in a way that it can do the test for the challenge ciphertext. It is also possible to weaken the definition such that no user decryption query is allowed for the challenge ciphertext under the challenge public key, no matter what the token is.

Justifications for our definition of hierarchical CLE. In the hierarchical scheme suggested (without a security definition) in [2], an entity at level k derives a trapdoor for its children at level $k+1$ using both its trapdoor and its secret key; while in our proposed model, a level k entity only uses its trapdoor obtained from its parent at level $k-1$ to derive keys for its children. However, we do not see any practical reason for requiring the secret key in the trapdoor derivation. On the other hand, the resulting scheme will be more complicated. For example, in the scheme of [2], the decryption requires the public keys of all the ancestors.

Note that we do allow the decryption of the ciphertext under $\overrightarrow{ID}'$ which is a prefix of $\overrightarrow{ID}^*$. This is stronger than the counterpart in some hierarchical IBE model [30].

Our definition is more general than plain CLE. The following theorem summarizes our discussion.

Theorem 1 *If there exists an* 1*-level SMCLE scheme which is secure under Definition 3 and 4, there exists a CLE scheme which is secure under the definition of [2].*

Proof. We describe how to build a simulator which make use of an adversary of CLE to break the security of our 1-level SMCLE scheme. The simulator basically forwards everything (the system parameters, the oracle queries and responses) back and forth between its own challenger (of breaking SMCLE) and the CLE adversary. For most queries, the monotonic details are omitted. The complete decryption queries made by the CLE adversary is entertained by combining the result of two partial decryption oracle queries. An important distinction between these "two worlds" is about public key replacement. The simulator needs to maintain a table to store the relationship between an identifier and a public key. Whenever a key replacement query is made, the simulator updates its own table. For every other requests regarding a particular identifier, the simulator retrieves the corresponding public key in its table and queries its own challenger accordingly. □

4 Our Proposed Construction

4.1 Preliminaries

Let $\mathbb{G}$ be a multiplicative group of prime order p and $\mathbb{G}_T$ be a multiplicative group also of order p. We assume the existence of an efficiently computable bilinear map $\hat{e} : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ such that

1. *Bilinearity*: For all $u, v \in \mathbb{G}$ and $r, s \in \mathbb{Z}_p$, $\hat{e}(u^r, v^s) = \hat{e}(u, v)^{rs}$.
2. *Non-degeneracy*: $\hat{e}(u, v) \neq 1_{\mathbb{G}_T}$ for all $u, v \in \mathbb{G} \setminus \{1_{\mathbb{G}}\}$.

We also assume the following problems are intractable in such groups.

Definition 5. *The* Decision **3**-Party Diffie-Hellman Problem *(3-DDH) in $\mathbb{G}$ is to decide if $T = g^{\beta\gamma\delta}$ given $(g, g^\beta, g^\gamma, g^\delta, T) \in \mathbb{G}^5$. Formally, defining the advantage of a* PPT *algorithm $\mathcal{D}$, $Adv_{\mathcal{D}}^{3-\mathrm{DDH}}(k)$, as*

$$|\Pr[1 \xleftarrow{\$} \mathcal{D}(g, g^\beta, g^\gamma, g^\delta, T) | T \leftarrow g^{\beta\gamma\delta} \wedge \beta, \gamma, \delta \xleftarrow{\$} \mathbb{Z}_p^*] - \Pr[1 \xleftarrow{\$} \mathcal{D}(g, g^\beta, g^\gamma, g^\delta, T) | T \xleftarrow{\$} \mathbb{G} \wedge \beta, \gamma, \delta \xleftarrow{\$} \mathbb{Z}_p^*]|.$$

We say 3-DDH is intractable if the advantage is a negligible function for all PPT *algorithms $\mathcal{D}$.*

Compared with the Bilinear Diffie-Hellman (BDH) problem, the problem instance of 3-DDH is purely in $\mathbb{G}$ while that of BDH contains an element $\hat{t} \in \mathbb{G}_T$. If BDH problem is solvable, one can solve 3-DDH by feeding $(g, g^\beta, g^\gamma, g^\delta, \hat{e}(g,T))$ to a BDH oracle. Apart from CLE [24], the above assumption has been employed in other advanced pairing-based cryptographic schemes such as [9].

We introduce a variant of the weak Bilinear Diffie-Hellman Inversion (BDHI) assumption [7] below in the favor of 3-DDH. The original h-wBDHI problem in $(\mathbb{G}, \mathbb{G}_T)$ [7] is to decide whether $\hat{t} = \hat{e}(g, g^\gamma)^{\alpha^{h+1}}$. The naming of "inversion" comes from the equivalence to the problem of deciding whether $\hat{t} = \hat{e}(g, g^\gamma)^{1/\alpha}$.

Definition 6. *The* Modified h-Weak Bilinear-Diffie-Hellman Inversion Problem *(h-wBDHI') in $\mathbb{G}$ is to decide if $T = g^{\gamma\alpha^{h+1}}$ given $(g, g^\gamma, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^h}, T) \in \mathbb{G}^{h+3}$. Formally, defining the advantage of a* PPT *algorithm $\mathcal{D}$, $Adv_{\mathcal{D}}^{h-\text{wBDHI}'}(k)$, as*

$$
\begin{aligned}
&|\Pr[1 \xleftarrow{\$} \mathcal{D}(g, g^\gamma, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^h}, T) | T \leftarrow g^{\gamma\alpha^{h+1}} \wedge \alpha, \gamma \xleftarrow{\$} \mathbb{Z}_p^*] \\
&\quad - \Pr[1 \xleftarrow{\$} \mathcal{D}(g, g^\gamma, g^\alpha, g^{\alpha^2}, \cdots, g^{\alpha^h}, T) | T \xleftarrow{\$} \mathbb{G} \wedge \alpha, \gamma \xleftarrow{\$} \mathbb{Z}_p^*]|.
\end{aligned}
$$

We say h-wBDHI' is intractable if the advantage is a negligible function for all PPT *algorithms $\mathcal{D}$.*

We require a hash function H drawn from a family of collision resistant hash functions too.

Definition 7. *A hash function $H \xleftarrow{\$} \mathcal{H}(k)$ is* collision resistant *if for all* PPT *algorithms $\mathcal{C}$ the advantage*

$$
Adv_{\mathcal{C}}^{\text{CR}}(k) = \Pr[H(x) = H(y) \wedge x \neq y | (x,y) \xleftarrow{\$} \mathcal{C}(1^k, H) \wedge H \xleftarrow{\$} \mathcal{H}(k)]
$$

is negligible as a function of the security parameter k.

4.2 Proposed Construction

Our construction is an h-level generalization of the concrete construction for 1-level in [24]. While [24] uses the technique of [11] to achieve strong decryption oracle, we use the same technique for a different purpose, which is a new way (other than the only known way in [21]) to support partial decryption oracle.

$\mathsf{Setup}(1^\lambda, n)$: Let $\mathbb{G}$, $\mathbb{G}_T$ be two multiplicative groups with a bilinear map $\hat{e}$ as defined before. They are of the same order p, which is a prime and $2^\lambda < p < 2^{\lambda+1}$.

- **Encryption key:** choose two generators $g, g_2 \in_R \mathbb{G}$.
- **Master public key:** choose an exponent $\alpha \in_R \mathbb{Z}_p$ and set $g_1 = g^\alpha$.
- **Hash key for identity-based public key derivation:** choose h many $(\ell+1)$-length vectors $\vec{U}_1, \cdots, \vec{U}_h \in_R \mathbb{G}^{\ell+1}$, where each $\vec{U}_j = (u'_j, u_{j,1}, \cdots, u_{j,\ell})$, $1 \leq j \leq h$. ℓ is a tunable parameter which is a factor of n and $1 \leq \ell \leq n$.
  Each vector $\vec{U}_j$ $(1 \leq j \leq h)$ corresponds to the j-th level of the hierarchy. We use the notation $\overrightarrow{ID} = (\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$ to denote an identity which is a hierarchy of different identities at different levels. Each ID_j is an n-bit string. We write ID_j as ℓ blocks each of length n/ℓ bits $(\mathsf{ID}_{j,1}, \cdots, \mathsf{ID}_{j,\ell})$. We define $F_{\vec{U}_j}(\mathsf{ID}_j) = u'_j \prod_{i=1}^{\ell} u_{j,i}^{\mathsf{ID}_{j,i}}$.
- **Hash key for ciphertext validity:** choose an $(n+1)$-length vector $\vec{V} = (v', v_1, \cdots, v_n) \in_R \mathbb{G}^{n+1}$ This vector defines the hash function $F_{\vec{V}}(w) = v' \prod_{j=1}^{n} {v_j}^{b_j}$ where w is a n-bit string $b_1 b_2 \cdots b_n$.
- **Hash function:** pick a function $H : \{0,1\}^* \to \{0,1\}^n$ from a family of collision-resistant hash functions.

The public parameters Pub and the master secret key Msk are given by

$$
\mathsf{Pub} = (\lambda, p, \mathbb{G}, \mathbb{G}_T, \hat{e}(\cdot,\cdot), n, \ell, g, g_1, g_2, \vec{U}_1, \cdots, \vec{U}_h, \vec{V}, H(\cdot)), \qquad \mathsf{Msk} = g_2^\alpha.
$$

We require the discrete logarithms (with respect to g) of all $\mathbb{G}$ elements in Pub except g_1 (and g) are unknown to the KGC. In practice, KGC can generate these elements from a pseudorandom function of a public seed.

$\mathsf{Extract}(d_{\overrightarrow{ID}}, \mathsf{ID}_r)$: For an identity $\overrightarrow{ID} = (\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$ for $k \leq h$, the trapdoor is in the form of

$$d_{\overrightarrow{ID}} = (a_1, a_2, \overrightarrow{z}_{k+1}, \cdots, \overrightarrow{z}_h) = (g_2^\alpha \cdot (\prod_{j=1}^{k} F_{\overrightarrow{U}_j}(\mathsf{ID}_j))^r, g^r, (\overrightarrow{U}_{k+1})^r, \cdots, (\overrightarrow{U}_h)^r),$$

where $r \in_R \mathbb{Z}_p^*$ and $(\overrightarrow{U}_j)^r = ((u'_j)^r, (u_{j,1})^r, \cdots, (u_{j,\ell})^r)$.

Note that (a_1, a_2) is sufficient for decryption, while $\overrightarrow{z}_{k+1}, \cdots, \overrightarrow{z}_h$ can help the derivation of the trapdoor for $(\mathsf{ID}_1, \cdots, \mathsf{ID}_k, \mathsf{ID}_{k+1})$ for any n-bit string ID_{k+1} and $k+1 \leq h$. The exact algorithm is as follows.

To generate $d_{\overrightarrow{ID}||\mathsf{ID}_r}$ parse $d_{\overrightarrow{ID}} = (a_1, a_2, (z_{k+1}, z_{k+1,1}, \cdots, z_{k+1,\ell}), \cdots (z_h, z_{h,1}, \cdots, z_{h,\ell}))$ and parse ID_r as ℓ blocks each of length n/ℓ bits $(\mathsf{ID}_{r,1}, \cdots, \mathsf{ID}_{r,\ell})$ pick $t \in_R \mathbb{Z}_p^*$ and output

$$d_{\overrightarrow{ID}||\mathsf{ID}_r} = (a_1 \cdot z_{k+1} \prod_{i=1}^{\ell} (z_{k+1,i})^{\mathsf{ID}_{r,i}} \cdot (\prod_{j=1}^{k+1} F_{\overrightarrow{U}_j}(\mathsf{ID}_j))^t, a_2 \cdot g^t, \overrightarrow{z}_{k+2} \cdot (\overrightarrow{U}_{k+2})^t \cdots, \overrightarrow{z}_h \cdot (\overrightarrow{U}_h)^t$$

where the multiplication of two vectors are defined component-wise, i.e. $\overrightarrow{z}_j \cdot \overrightarrow{\nu}_j = (z_j \cdot \nu_j, z_{j,1} \cdot \nu_{j,1}, \cdots, z_{j,\ell} \cdot \nu_{j,\ell})$. $d_{\overrightarrow{ID}}$ becomes shorter as the length of $\overrightarrow{ID}$ increases.

$\mathsf{KeyGen}()$: Pick $\mathsf{sk} \in_R \mathbb{Z}_p^*$, return sk as the secret key and $\mathsf{pk} = (X, Y) = (g^{\mathsf{sk}}, g_1^{\mathsf{sk}})$ as the public key.

$\mathsf{Enc}(m, \overrightarrow{ID}, \mathsf{pk})$: To encrypt $m \in \mathbb{G}_T$ for $\overrightarrow{ID} = (\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$ where $k \leq h$, parse pk as (X, Y), then check that it is a valid public key by verifying that $\hat{e}(X, g_1) = \hat{e}(g, Y)$[4]. If equality holds, pick $s \in_R \mathbb{Z}_p^*$ and compute

$$\begin{aligned} C &= (C_1, C_2, \tau, \sigma) \\ &= (m \cdot \hat{e}(Y, g_2)^s, \prod_{j=1}^{k} F_{\overrightarrow{U}_j}(\mathsf{ID}_j)^s, g^s, F_{\overrightarrow{V}}(w)^s) \end{aligned}$$

where $w = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk})$.

$\mathsf{Dec}^S(C, d_{\overrightarrow{ID}})$: Parse C as (C_1, C_2, τ, σ), and $d_{\overrightarrow{ID}}$ as $(a_1, a_2, \cdots)$. First check if $\hat{e}(\tau, \prod_{j=1}^k F_{\overrightarrow{U}_j}(\mathsf{ID}_j) \cdot F_{\overrightarrow{V}}(w')) = \hat{e}(g, C_2 \cdot \sigma)$ where $w' = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk})$. Return $\perp$ if inequality holds or any parsing is not possible, otherwise pick $t \in_R \mathbb{Z}_p^*$ and return

$$\begin{aligned} D &= (D_1, D_2, D_3) \\ &= (a_1 \cdot F_{\overrightarrow{V}}(w')^t, a_2, g^t) \end{aligned}$$

$\mathsf{Dec}^U(C, \mathsf{sk}, D)$: Parse C as (C_1, C_2, τ, σ) and check if $\hat{e}(\tau, \prod_{j=1}^k F_{\overrightarrow{U}_j}(\mathsf{ID}_j) \cdot F_{\overrightarrow{V}}(w')) = \hat{e}(g, C_2 \cdot \sigma)$ where $w' = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk})$. If equality does not hold or parsing is not possible, return $\perp_C$. Next, parse D as (D_1, D_2, D_3) and check if $\hat{e}(g, D_1) = \hat{e}(g_1, g_2)\hat{e}(D_2, \prod_{j=1}^k F_{\overrightarrow{U}_j}(\mathsf{ID}_j))\hat{e}(D_3, F_{\overrightarrow{V}}(w'))$[5]. If equality does not hold or parsing is not possible, return $\perp_D$. Otherwise, return

$$m \leftarrow C_1 \cdot \left(\frac{\hat{e}(C_2, D_2)\hat{e}(\sigma, D_3)}{\hat{e}(\tau, D_1)} \right)^{\mathsf{sk}}.$$

[4] One pairing computation can be saved by a technique in [24] (which is similar to the technique in [35]): pick $\xi \in_R \mathbb{Z}_p^*$ and compute $C_1 = m \cdot \hat{e}(Y, g_2 \cdot g^\xi)^s / \hat{e}(X, g_1^{s\xi})$.

[5] The same trick for minimizing the number of pairing computations [35] involved in checking the ciphertext and the token can be incorporated to the final decryption step. The modified decryption algorithm only uses 4 pairing computations; however, it gives a random message for an invalid ciphertext.

4.3 Security Analysis

Theorem 2 *Our scheme is secure against Type-I attack (Definition 3) if h-wBDHI' problem is intractable.*

Theorem 3 *Our scheme is secure against Type-II attack (Definition 4) if 3-DDH problem is intractable.*

Proofs for the above two theorems can be found in a single proof in Appendix A.

5 Applying General Certificateless Encryption to Timed-Release Encryption

Now we provide the formal evidence for our thesis – generic certificateless encryption can be used as TRE.

5.1 Syntax of Timed-Release Encryption

For ease of discussion, we consider a definition supporting only a single-level of time-identifier as in [14]. It can be shown that our results hold for an h-level analog. Below gives the security model of [14][6].

Definition 8. *A TRE scheme for time-identifiers of length n (where n is a polynomially-bounded function) is defined by the following sextuple of* PPT *algorithms:*

- Setup *(run by the server) is a probabilistic algorithm which takes a security parameter 1^λ, outputs a master secret key* Msk*, and the global parameters* Pub*. We assume that λ and $n = n(\lambda)$ are implicit in* Pub *and all other algorithms take* Pub *implicitly as an input.*
- $\mathsf{Extract}$ *(run by the server) is a possibly probabilistic algorithm which takes the master secret key* Msk *and a string $\mathsf{ID} \in \{0,1\}^n$, outputs a trapdoor key d_{ID} associated with the identity* ID*.*
- KeyGen *(run by a user) is a probabilistic algorithm which generates a public/private key pair $(\mathsf{pk}_u, \mathsf{sk}_u)$.*
- Enc *(run by a sender) is a probabilistic algorithm which takes a message m from some implicit message space, an identifier $\mathsf{ID} \in \{0,1\}^n$, and the receiver's public key pk_u as input, returns a ciphertext C.*
- Dec^S *(run by the one who hold the trapdoor, either a SEM or a receiver) is a possibly probabilistic algorithm which takes a ciphertext C and the trapdoor key d_{ID} as input, returns either a token D which can be seen as a partial decryption result of C, or an invalid flag $\perp$ (which is not in the message space).*
- Dec^U *(run by a receiver) is a possibly probabilistic algorithm which takes the ciphertext C, the receiver's private key sk_u and a token D as input, returns either the plaintext, an invalid flag $\perp_D$ denoting D is an invalid token, or an invalid flag $\perp_C$ denoting the ciphertext is invalid.*

For correctness, we require that $\mathsf{Dec}^U(C, \mathsf{sk}, \mathsf{Dec}^S(C, \mathsf{Extract}(\mathsf{Msk}, \mathsf{ID}))) = m$ *for all $\lambda \in \mathbb{N}$, all* $(\mathsf{Pub}, \mathsf{Msk}) \stackrel{\$}{\leftarrow} \mathsf{Setup}(1^\lambda)$*, all* $(\mathsf{pk}, \mathsf{sk}) \stackrel{\$}{\leftarrow} \mathsf{KeyGen}$*, all message m, all identifier ID in $\{0,1\}^n$ and all $C \stackrel{\$}{\leftarrow} \mathsf{Enc}(m, \mathsf{ID}, \mathsf{pk})$.*

5.2 Timed-Release Encryption from General Certificateless Encryption

Given a SMCLE scheme $\{\mathcal{SMC}.\mathsf{Setup}, \mathcal{SMC}.\mathsf{Extract}, \mathcal{SMC}.\mathsf{KeyGen}, \mathcal{SMC}.\mathsf{Enc}, \mathcal{SMC}.\mathsf{Dec}^S, \mathcal{SMC}.\mathsf{Dec}^U\}$, a TRE scheme $\{\mathcal{TRE}.\mathsf{Setup}, \mathcal{TRE}.\mathsf{Extract}, \mathcal{TRE}.\mathsf{KeyGen}, \mathcal{TRE}.\mathsf{Enc}, \mathcal{TRE}.\mathsf{Dec}^S, \mathcal{TRE}.\mathsf{Dec}^U\}$ can be constructed in the following straightforward way.

$\mathcal{TRE}.\mathsf{Setup}(1^\lambda, n)$: Given a security parameter λ and the length of the time-identifier n, execute $(\mathsf{Msk}, \mathsf{Pub}) \leftarrow \mathcal{SMC}.\mathsf{Setup}(1^\lambda, n)$, retain Msk as the master secret key and publish Pub as the global parameters.

---

[6] The model in [14] is stronger than its counterparts in [34, 20]. We chose not to use the framework from [25] since it is tightly coupled with the pre-open capability. However, the essence of the message confidentiality requirements in [25] is still being captured in the model of [14].

$\mathcal{TRE}$.Extract(Msk, ID): For a time-identifier $\mathsf{ID} \in \{0,1\}^n$, the time-server executes $d_{\mathsf{ID}} \leftarrow \mathcal{SMC}.\mathsf{Extract}(\mathsf{Msk}, \mathsf{ID})$ and return the trapdoor d_{ID}.

$\mathcal{TRE}$.KeyGen(): Return $(\mathsf{sk}, \mathsf{pk}) \leftarrow \mathcal{SMC}.\mathsf{KeyGen}()$ as the user's private/public key pair.

$\mathcal{TRE}.\mathsf{Enc}(m, \mathsf{ID}, \mathsf{pk})$: To encrypt $m \in \mathbb{G}_T$ for pk under the time $\mathsf{ID} \in \{0,1\}^n$, first perform any checking of pk that is required by the $\mathcal{SMC}$ scheme. If pk is a valid public key, return $\mathcal{SMC}.\mathsf{Enc}(m, \mathsf{ID}, \mathsf{pk})$.

$\mathcal{TRE}.\mathsf{Dec}^S(C, d_{\mathsf{ID}})$: To partially decrypt the ciphertext C by the time-dependent trapdoor d_{ID}, just return the token $D \leftarrow \mathcal{SMC}.\mathsf{Dec}^S(C, d_{\mathsf{ID}})$.

$\mathcal{TRE}.\mathsf{Dec}^U(C, \mathsf{sk}, D)$: To decrypt the ciphertext C by the secret key sk and the token D, just return $\mathcal{SMC}.\mathsf{Dec}^U(C, \mathsf{sk}, D)$.

Theorem 4 *If $\mathcal{SMC}$ is an 1-level SMCLE scheme which is CCA-secure against Type-I adversary (Definition 3), $\mathcal{TRE}$ is CCA-secure against Type-I adversary (Definition 10).*

Theorem 5 *If $\mathcal{SMC}$ is an 1-level SMCLE scheme which is CCA-secure against Type-II adversary (Definition 4), $\mathcal{TRE}$ is CCA-secure against Type-II adversary (Definition 11).*

Proof. We prove the above theorems by contradiction. Suppose $\mathcal{A}$ is a PPT Type-X adversary such that $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA'-X}}(\lambda) = 1] - \frac{1}{2}| > \epsilon$, we construct an adversary $\mathcal{B}$ such that $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA-X}}(\lambda) = 1] - \frac{1}{2}| > \epsilon$ in the face of a SMCLE challenger $\mathcal{C}$ where the running times of $\mathcal{B}$ and $\mathcal{A}$ are equal.

Setup: When $\mathcal{C}$ gives $\mathcal{B}$ $(\mathsf{Pub}, \mathsf{Aux})$, $\mathcal{B}$ just forwards it to $\mathcal{A}$.

First Phase of Queries: $\mathcal{B}$ forwards every request of $\mathcal{A}$ to the oracles of its own challenger $\mathcal{C}$. From the description of $\mathcal{TRE}$, we can see that every legitimate oracle query made by $\mathcal{A}$ can be answered faithfully.

Challenge: When $\mathcal{A}$ gives $\mathcal{B}$ $(m_0, m_1, \mathsf{pk}^*, \mathsf{ID}^*)$, $\mathcal{B}$ just forwards it to $\mathcal{C}$.

Second Phase of Queries: Again, $\mathcal{B}$ just forwards every request of $\mathcal{A}$ to the oracles of its own challenger $\mathcal{C}$. From the description of $\mathcal{TRE}$, it is easy to see that every oracle query which does not violate the restriction made by $\mathcal{A}$ also does not violate the restriction made enforced by $\mathcal{C}$.

Output: Finally, $\mathcal{A}$ outputs a bit b, $\mathcal{B}$ forwards it to $\mathcal{C}$ as its own answer. The probability for $\mathcal{A}$ to win the TRE experiment simulated by $\mathcal{B}$ is equals to the probability for $\mathcal{B}$ to win the SMCLE game played against $\mathcal{C}$. It is easy to see that the running times of $\mathcal{A}$ and $\mathcal{B}$ are the same. □

Section 4 presented a CLE scheme in the standard model, the above theorems imply that our scheme can be instantiated as a TRE scheme without random oracle, which is the first one in the literature.

5.3 Certificateless Encryption from Timed-Release Encryption

One may expect that a general CLE can be constructed from any TRE. Nevertheless, note that the usage of time-identifier is only a specific instantiation of the timed-release idea. For example, there exists TRE scheme [16] which time is denoted by a repeated computation of one-way hash function similar to S/Key password system. On the other hand, the notion of CLE supports an exponential number of arbitrary identifiers[7]. A CLE scheme cannot be realized by a TRE if the total number of different time periods supported is too few.

5.4 Security-Mediator in Timed-Release Encryption

We introduce the concept of security-mediator in the TRE paradigm, which gives a new business model for the operation of the time-server. Traditional TRE only allows the time-server to release a system-wide time-dependent trapdoor. With the possibility of partial decryption, the time-server can charge for each decryption. The time-server can decrypt a ciphertext partially by the time-dependent trapdoor per request, while the partial decryption of one ciphertext would not help the decryption of any other ciphertext.

[7] Even though the scheme may be insecure when more than a polynomial number of trapdoors are compromised by a single adversary.

5.5 Time Hierarchy

Each identifier corresponds to a single time period, which means that the server has to publish t private keys on a bulletin board after t time-periods have passed. Given a hierarchical CLE, the amount data on the bulletin board can be reduced by using CHK forward secure encryption scheme [13] in reverse, as suggested in [7]. For a total of T time periods, the CHK framework is setup as a tree of depth $\lg T$. To encrypt a message for time $t < T$, the time identifier is the CHK identifier for time period $T - t$. Release of trapdoor is done in the same manner, the private key for the time period $T-t$ is released on the t^{th} time period. This single private key enables anyone to derive the private keys for CHK time periods $T-t, T-t+1, \cdots, T$, which means the user can obtain the trapdoors for time in the range of $1, \cdots, t$. By using this trick, the server only needs to publish a single private key comprising $O(\lg^2 T)$ group elements at any time.

6 Conclusions

In the study of cryptography, we always seek for the strongest definition and try to achieve it. Previous models of certificateless encryption (CLE) were too restrictive. In particular, they cannot give the desired security properties when instantiated as timed-release encryption (TRE). Our generalized model for CLE is sufficient to fulfill the requirements of TRE. All future CLE proposals in our general model automatically give secure TRE schemes. Our model is defined against full-identifier extraction, decryption under arbitrary public key, and partial decryption, which incorporates the strongest properties one may desire. Our concrete scheme yields the first strongly-secure (hierarchical) security-mediated CLE and the first TRE in the standard model.

References

1. Sattam S. Al-Riyami, John Malone-Lee, and Nigel P. Smart. Escrow-free Encryption Supporting Cryptographic Workflow. *Int. J. Inf. Sec.*, 5(4):217–229, 2006.
2. Sattam S. Al-Riyami and Kenneth G. Paterson. Certificateless Public Key Cryptography. In Chi-Sung Laih, editor, *Advances in Cryptology - ASIACRYPT 2003, 9th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, November 30 - December 4, 2003, Proceedings*, volume 2894 of *Lecture Notes in Computer Science*, pages 452–473. Springer, 2003. Full version at `http://eprint.iacr.org/2003/126`.
3. Sattam S. Al-Riyami and Kenneth G. Paterson. CBE from CL-PKE: A Generic Construction and Efficient Schemes. In Serge Vaudenay, editor, *Public Key Cryptography*, volume 3386 of *Lecture Notes in Computer Science*, pages 398–415. Springer, 2005.
4. Joonsang Baek, Reihaneh Safavi-Naini, and Willy Susilo. Certificateless Public Key Encryption Without Pairing. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *Information Security, 8th International Conference, ISC 2005, Singapore, September 20-23, 2005, Proceedings*, volume 3650 of *Lecture Notes in Computer Science*, pages 134–148. Springer, 2005.
5. Mihir Bellare and Shafi Goldwasser. Verifiable Partial Key Escrow. In *ACM Conference on Computer and Communications Security*, pages 78–91, 1997.
6. Ian F. Blake and Aldar C-F. Chan. Scalable, Server-Passive, User-Anonymous Timed Release Cryptography. In *Distributed Computing Systems, ICDCS 2005*, pages 504–513. IEEE, 2005.
7. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In Ronald Cramer, editor, *EUROCRYPT*, volume 3494 of *Lecture Notes in Computer Science*, pages 440–456. Springer, 2005.
8. Dan Boneh and Moni Naor. Timed Commitments. In Mihir Bellare, editor, *Advances in Cryptology - CRYPTO 2000, 20th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2000, Proceedings*, volume 1880 of *Lecture Notes in Computer Science*, pages 236–254. Springer, 2000.
9. Dan Boneh, Amit Sahai, and Brent Waters. Fully Collusion Resistant Traitor Tracing with Short Ciphertexts and Private Keys. In Serge Vaudenay, editor, *EUROCRYPT*, volume 4004 of *Lecture Notes in Computer Science*, pages 573–592. Springer, 2006.

10. Dan Boneh and Brent Waters. Conjunctive, Subset, and Range Queries on Encrypted Data. In Salil P. Vadhan, editor, *Theory of Cryptography, 4th Theory of Cryptography Conference, TCC 2007, Amsterdam, The Netherlands, February 21-24, 2007, Proceedings*, volume 4392 of *Lecture Notes in Computer Science*, pages 535–554. Springer, 2007.
11. Xavier Boyen, Qixiang Mei, and Brent Waters. Direct Chosen Ciphertext Security from Identity-based Techniques. In Vijay Atluri, Catherine Meadows, and Ari Juels, editors, *ACM Conference on Computer and Communications Security*, pages 320–329. ACM, 2005.
12. Xavier Boyen and Brent Waters. Anonymous Hierarchical Identity-Based Encryption (Without Random Oracles). In Cynthia Dwork, editor, *Advances in Cryptology - CRYPTO 2006, 26th Annual International Cryptology Conference, Santa Barbara, California, USA, August 20-24, 2006, Proceedings*, volume 4117 of *Lecture Notes in Computer Science*, pages 290–307. Springer, 2006.
13. Ran Canetti, Shai Halevi, and Jonathan Katz. A Forward-Secure Public-Key Encryption Scheme. *Journal of Cryptology*, 20(3):265–294, 2007.
14. Julien Cathalo, Benoît Libert, and Jean-Jacques Quisquater. Efficient and Non-interactive Timed-Release Encryption. In Sihan Qing, Wenbo Mao, Javier Lopez, and Guilin Wang, editors, *Information and Communications Security, ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, pages 291–303. Springer, 2005.
15. Konstantinos Chalkias, Dimitrios Hristu-Varsakelis, and George Stephanides. Improved Anonymous Timed-Release Encryption . In Joachim Biskup and Javier Lopez, editors, *Computer Security - ESORICS 2007, 12th European Symposium on Research in Computer Security, Dresden, Germany, September 24-26, 2007, Proceedings*, volume 4734 of *Lecture Notes in Computer Science*, pages 311–326. Springer, 2007.
16. Konstantinos Chalkias and George Stephanides. Timed Release Cryptography from Bilinear Pairings Using Hash Chains. In Herbert Leitold and Evangelos P. Markatos, editors, *Communications and Multimedia Security*, volume 4237 of *Lecture Notes in Computer Science*, pages 130–140. Springer, 2006.
17. Sanjit Chatterjee and Palash Sarkar. New Constructions of Constant Size Ciphertext HIBE Without Random Oracle. In Min Surp Rhee and Byoungcheon Lee, editors, *ICISC*, volume 4296 of *Lecture Notes in Computer Science*, pages 310–327. Springer, 2006.
18. Sanjit Chatterjee and Palash Sarkar. On (Hierarchical) Identity Based Encryption Protocols with Short Public Parameters (With an Exposition of Waters' Artificial Abort Technique). Cryptology ePrint Archive, Report 2006/279, 2006.
19. Zhaohui Cheng, Liqun Chen, Li Ling, and Richard Comley. General and Efficient Certificateless Public Key Encryption Constructions. In Takagi et al. [45], pages 83–107.
20. Jung Hee Cheon, Nicholas Hopper, Yongdae Kim, and Ivan Osipkov. Timed-Release and Key-Insulated Public Key Encryption. In Giovanni Di Crescenzo and Avi Rubin, editors, *Financial Cryptography*, volume 4107 of *Lecture Notes in Computer Science*, pages 191–205. Springer, 2006.
21. Sherman S. M. Chow, Colin Boyd, and Juan Manuel González Nieto. Security-Mediated Certificateless Cryptography. In Yung et al. [47], pages 508–524.
22. Giovanni Di Crescenzo, Rafail Ostrovsky, and Sivaramakrishnan Rajagopalan. Conditional Oblivious Transfer and Timed-Release Encryption. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 74–89. Springer, 1999.
23. Alexander W. Dent. A Survey of Certificateless Encryption Schemes and Security Models. Cryptology ePrint Archive, Report 2006/211, 2006. `http://eprint.iacr.org/`.
24. Alexander W. Dent, Benoit Libert, and Kenneth G. Paterson. Certificateless Encryption Schemes Strongly Secure in the Standard Model. In *Public Key Cryptography - PKC 2008*, volume 4939 of *Lecture Notes in Computer Science*. Springer, 2008. To Appear. Full version at `http://eprint.iacr.org/2007/121`.
25. Alexander W. Dent and Qiang Tang. Revisiting the Security Model for Timed-Release Public-Key Encryption with Pre-Open Capability. In Juan Garay, Arjen K. Lenstra, Masahiro Mambo, and Rene Peralta, editors, *Information Security, ISC 2007*, volume 4779 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2007.
26. Yevgeniy Dodis and Dae Hyun Yum. Time Capsule Signature. In Andrew S. Patrick and Moti Yung, editors, *Financial Cryptography and Data Security, 9th International Conference, FC 2005, Roseau, The Commonwealth of Dominica, February 28 - March 3, 2005, Revised Papers*, volume 3570 of *Lecture Notes in Computer Science*, pages 57–71. Springer, 2005.
27. Cynthia Dwork and Moni Naor. Pricing via Processing or Combatting Junk Mail. In Ernest F. Brickell, editor, *Advances in Cryptology - CRYPTO '92, 12th Annual International Cryptology Conference, Santa Barbara, California, USA, August 16-20, 1992, Proceedings*, volume 740 of *Lecture Notes in Computer Science*, pages 139–147. Springer, 1992.

28. David Galindo, Paz Morillo, and Carla Ràfols. Breaking Yum and Lee Generic Constructions of Certificate-Less and Certificate-Based Encryption Schemes. In Andrea S. Atzeni and Antonio Lioy, editors, *EuroPKI*, volume 4043 of *Lecture Notes in Computer Science*, pages 81–91. Springer, 2006.
29. Craig Gentry. Practical Identity-Based Encryption Without Random Oracles. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.
30. Craig Gentry and Alice Silverberg. Hierarchical ID-Based Cryptography. In Yuliang Zheng, editor, *Advances in Cryptology - ASIACRYPT 2002, 8th International Conference on the Theory and Application of Cryptology and Information Security, Queenstown, New Zealand, December 1-5, 2002, Proceedings*, volume 2501 of *Lecture Notes in Computer Science*, pages 548–566. Springer, 2002.
31. Matthew Green and Susan Hohenberger. Blind Identity-Based Encryption and Simulatable Oblivious Transfer. In Kaoru Kurosawa, editor, *Advances in Cryptology - ASIACRYPT 2007, 13th International Conference on the Theory and Application of Cryptology and Information Security, Kuching, Sarawak, Malaysia, December 2-6, 2007, Proceedings*, volume 4833 of *Lecture Notes in Computer Science*, pages 265–282. Springer, 2007.
32. Dimitrios Hristu-Varsakelis, Konstantinos Chalkias, and George Stephanides. Low-cost Anonymous Timed-Release Encryption. In *The Third International Symposium on Information Assurance and Security*, pages 77–82. IEEE, 2007.
33. Bessie C. Hu, Duncan S. Wong, Zhenfeng Zhang, and Xiaotie Deng. Certificateless Signature: A New Security Model and An Improved Generic Construction. *Designs, Codes and Cryptography*, 42(2):109–126, 2007.
34. Yong Ho Hwang, Dae Hyun Yum, and Pil Joong Lee. Timed-Release Encryption with Pre-open Capability and Its Application to Certified E-mail System. In Jianying Zhou, Javier Lopez, Robert H. Deng, and Feng Bao, editors, *ISC*, volume 3650 of *Lecture Notes in Computer Science*, pages 344–358. Springer, 2005.
35. Eike Kiltz and David Galindo. Direct Chosen-Ciphertext Secure Identity-Based Key Encapsulation Without Random Oracles. In Lynn Margaret Batten and Reihaneh Safavi-Naini, editors, *ACISP*, volume 4058 of *Lecture Notes in Computer Science*, pages 336–347. Springer, 2006.
36. Junzuo Lai and Weidong Kou. Self-Generated-Certificate Public Key Encryption Without Pairing. In Tatsuaki Okamoto and Xiaoyun Wang, editors, *Public Key Cryptography*, volume 4450 of *Lecture Notes in Computer Science*, pages 476–489. Springer, 2007.
37. Benoît Libert and Jean-Jacques Quisquater. On Constructing Certificateless Cryptosystems from Identity Based Encryption. In Yung et al. [47], pages 474–490.
38. Joseph K. Liu, Man Ho Au, and Willy Susilo. Self-Generated-Certificate Public Key Cryptography and Certificateless Signature / Encryption Scheme in the Standard Model. In Feng Bao and Steven Miller, editors, *ASIACCS*. ACM, 2007.
39. Timothy May. Time-release Crypto, Feb 1993. Manuscript, available at `http://www.cyphernet.org/cyphernomicon/chapter14/14.5.html`.
40. Marco Casassa Mont, Keith Harrison, and Martin Sadler. The HP Time Vault Service: Exploiting IBE for Timed Release of Confidential Information. In *Proceedings of the Twelfth International World Wide Web Conference, WWW2003, Budapest, Hungary, 20-24 May 2003. ACM, 2003*, pages 160–169, 2003.
41. Deholo Nali, Carlisle M. Adams, and Ali Miri. Hierarchical Time-based Information Release. *International Journal of Information Security*, 5(2):92–104, 2006.
42. Jong Hwan Park, Kyu Young Choi, Jung Yeon Hwang, and Dong Hoon Lee. Certificateless Public Key Encryption in the Selective-ID Security Model (Without Random Oracles). In Takagi et al. [45], pages 60–82.
43. Ronald L. Rivest, Adi Shamir, and David A. Wagner. Time-lock Puzzles and Timed-release Crypto. Technical Report MIT/LCS/TR-684, Massachusetts Institute of Technology, 1996.
44. Yinxia Sun, Futai Zhang, and Joonsang Baek. Strongly Secure Certificateless Public Key Encryption Without Pairing. In Feng Bao, San Ling, Tatsuaki Okamoto, Huaxiong Wang, and Chaoping Xing, editors, *CANS*, volume 4856 of *Lecture Notes in Computer Science*, pages 194–208. Springer, 2007.
45. Tsuyoshi Takagi, Tatsuaki Okamoto, Eiji Okamoto, and Takeshi Okamoto, editors. *Pairing-Based Cryptography - Pairing 2007, First International Conference, Tokyo, Japan, July 2-4, 2007, Proceedings*, volume 4575 of *Lecture Notes in Computer Science*. Springer, 2007.
46. Brent Waters. Efficient Identity-Based Encryption Without Random Oracles. In Ronald Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.
47. Moti Yung, Yevgeniy Dodis, Aggelos Kiayias, and Tal Malkin, editors. *Public Key Cryptography - PKC 2006, 9th International Conference on Theory and Practice of Public-Key Cryptography, New York, NY, USA, April 24-26, 2006, Proceedings*, volume 3958 of *Lecture Notes in Computer Science*. Springer, 2006.

48. Zhenfeng Zhang and Dengguo Feng. On the Security of a Certificateless Public-Key Encryption. Cryptology ePrint Archive, Report 2005/426, 2005.

A Formal Security Proof for Our Proposed Construction

We now define a series of games where each one is an interactive game between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$, which is either an insider attacker (Type-I adversary) or a curious server (Type-II adversary), depending on the allowed queries. The skeleton of the proof is based on the proof given in [24].

Game 1 (The Original Game). This game is the one played between a simulator $\mathcal{S}$ and an adversary $\mathcal{A}$ as specified in the experiment $\mathbf{Exp}^{\mathsf{CCA-X}}$. We use the following notation: For the queries, let $\mathcal{T} = \{\overrightarrow{ID}_1, \cdots, \overrightarrow{ID}_{q_E}\}$ denote the trapdoors extraction queries and $\mathcal{W} = \{w_1, \cdots, w_{q_D}\}$ be the set of strings involved in decryption queries where $w_j = H(C_1, C_2, \tau, \overrightarrow{ID}_j, \mathsf{pk}_j)$. For the challenges, let $\overrightarrow{ID}^*$ and pk^* denote the challenge identifier and the challenge public key respectively, and let $C^* = (C_1^*, C_2^*, \tau^*, \sigma^*)$ be the returned challenge ciphertext and let $w^* = H(C_1^*, C_2^*, \tau^*, \overrightarrow{ID}^*, \mathsf{pk}^*)$. The random bit ι is chosen by $\mathcal{S}$ in order to select which message is encrypted.

Game 2 (Change of Public Parameters). Let $Z_i = (g)^{\alpha^i}$, $1 \le i \le h+1$. This game is the same as Game 1 except that the generation of the parameters is changed. $\mathcal{S}$ picks $\alpha, \beta \in_R \mathbb{Z}_p$, and set $g_1 = Z_1$, $g_2 = Z_h \cdot g^{\beta}$.

The simulator also changes the vectors as follows. Let ρ_u and ρ_v be integers such that $\rho_u(n+1) < p$ and $\rho_v(n+1) < p$. The exact choices of ρ_u and ρ_u will be determined later. The simulator randomly selects

- $\kappa_{u_1}, \cdots, \kappa_{u_n}, \kappa_v$ from $\{0, \cdots, \ell(n^{1/\ell} - 1)\}$,
- h many $(\ell+1)$-length vectors $\vec{x}_1, \cdots, \vec{x}_h$ from $\mathbb{Z}_{\rho_u}$, where each $\vec{x}_j = (x'_j, x_{j,1}, \cdots, x_{j,\ell})$.
- h many $(\ell+1)$-length vectors $\vec{y}_1, \cdots, \vec{y}_h$ from $\mathbb{Z}_p$, where each $\vec{y}_j = (y'_j, y_{j,1}, \cdots, y_{j,\ell})$.
- $(x'_v, x_{v,1}, \cdots, x_{v,n})$ from $\mathbb{Z}_{\rho_v}^{n+1}$
- $(y'_v, y_{v,1}, \cdots, y_{v,n})$ from $\mathbb{Z}_p^{n+1}$.

The hash keys for the identity-based key derivation, for $1 \le j \le h$, are set as:

$$u'_j = Z_{h-j+1}^{(p+\rho_u\kappa_j - x'_j)} \cdot g^{y'_j}, \quad u_{j,i} = Z_{h-j+1}^{-x'_{j,i}} \cdot g^{y_{j,i}} \text{ for } 1 \le i \le \ell.$$

The hash key for the ciphertext validity is set as (note that $g_2 = Z_h \cdot g^{\beta}$):

$$v' = g_2^{(p+\rho_v\kappa_v - x'_v)} \cdot g^{y'_v}, \qquad v_i = g_2^{-x_{v,i}} g^{y'_{v,i}} \text{ for } 1 \le i \le n.$$

Define the following functions

$$J_{u_1}(\mathsf{ID}_1) = p + \rho_u\kappa_1 - x'_1 - \sum_{i=1}^{\ell} x_{1,i}\mathsf{ID}_{1,i},\ K_{u_1}(\mathsf{ID}_1) = y'_1 + \textstyle\sum_{1=1}^{\ell} y_{1,i}\mathsf{ID}_{h,i},$$

$$\vdots \qquad\qquad \vdots$$

$$J_{u_h}(\mathsf{ID}_h) = p + \rho_u\kappa_h - x'_h - \sum_{i=1}^{\ell} x_{h,i}\mathsf{ID}_{h,i},\ K_{u_h}(\mathsf{ID}_h) = y'_h + \textstyle\sum_{1=1}^{\ell} y_{h,i}\mathsf{ID}_{h,i},$$

$$J_v(w) = p + \rho_v\kappa_v - x'_v - \sum_{i=1}^{\ell} x_{v,j}b_j, \qquad K_v(w) = y'_v + \textstyle\sum_{1=1}^{\ell} y_{v,j}b_j,$$

that take as input $\mathsf{ID}_j = (\mathsf{ID}_{j,1}, \cdots, \mathsf{ID}_{j,\ell})$ or $w = b_1 \cdots b_n$. The settings above give

$$F_{\overrightarrow{U}_j}(\mathsf{ID}_j) = u'_j \prod_{i=1}^{\ell} u_{j,i}^{\mathsf{ID}_{j,i}} = Z_{h-j+1}^{J_{u_j}(\mathsf{ID}_j)} \cdot g^{K_{u_j}(\mathsf{ID}_j)}, j \in \{1, \cdots, h\}$$

$$F_{\overrightarrow{V}}(w) = v' \prod_{j=1}^{n} v_j{}^{b_j} = g_2^{J_v(w)} \cdot g^{K_v(w)}$$

These changes do not change the distribution of the public parameters, so we have $\Pr[S_2] = \Pr[S_1]$.

Game 3 (Elimination of Hash Collisions). The simulator aborts and assumes $\mathcal{A}$ outputs a random bit in this game if $\mathcal{A}$ submits a decryption query $(C, \overrightarrow{ID}, \mathsf{pk} = (g^{\mathsf{sk}}, g_1^{\mathsf{sk}}))$ for a well-formed ciphertext $C = (C_1, C_2, \tau, \sigma)$ where $w = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk})$ is either equal to the same value as a previously submitted ciphertext or w^* of the challenge ciphertext.

For such a decryption query to be legal, we have $C \neq C^*$ or $(\overrightarrow{ID}, \mathsf{pk}) \neq (\overrightarrow{ID}^*, \mathsf{pk}^*)$. In either case, this implies a collision for H, which means we can construct an adversary $\mathcal{C}$ against the collision resistance of H such that $|\Pr[S_3] - \Pr[S_2]| \leq Adv_{\mathcal{C}}^{\mathrm{CR}}(k)$.

Game 4 (Preparation for the Simulation of the Challenge Ciphertext). Let $\overrightarrow{ID}^* = (\mathsf{ID}_1^*, \cdots, \mathsf{ID}_k^*)$ where $k \leq h$. This time $\mathcal{S}$ aborts if $J_{u_j}(\mathsf{ID}_j^*) \neq 0 \bmod p$ for any $j \in \{1, \cdots, k\}$ or $J_v(w^*) \neq 0 \bmod p$.

Since the values determining these functions are information theoretically hidden from $\mathcal{A}$, such ID^* and w^* can only be produced by chance. Therefore

$$\begin{aligned}
&\Pr[J_v(w^*) = 0 \bmod p] \\
&= \Pr[J_v(w^*) = 0 \bmod p | J_v(w^*) = 0 \bmod \rho_v] \cdot \Pr[J_v(w^*) = 0 \bmod \rho_v] \\
&= \frac{1}{\rho_v(n+1)}
\end{aligned}$$

Unless $\mathcal{S}$ aborts, Game 3 and Game 4 are identical and we have $|\Pr[S_4] - \Pr[S_3]| \leq \frac{1}{(\rho_u)^h \rho_v (\ell+1)^{h+1}}$ by a similar computation ($n \geq \ell$). The significance of this extra abort condition will be manifested in Game 7.

Game 5 (Artificial Abort for Consistent View of Adversary). Now $\mathcal{S}$ aborts if $J_{u_1}(\mathsf{ID}'_1) = \cdots = J_{u_k}(\mathsf{ID}'_k) = 0 \bmod \rho_u$ for some $\overrightarrow{ID}' = (\mathsf{ID}'_1, \cdots, \mathsf{ID}'_k) \in \mathcal{T}$ or $J_v(w') = 0 \bmod \rho_v$ for some $w' \in \mathcal{W}$.

Since $\mathcal{A}$'s power is dependent on the extraction and decryption queries, the above abort event is not independent of S_4, and we cannot relate the probability of S_4 and S_5 in a similar way as before.

This problem can be circumvented by the "re-normalization" technique due to Waters [46], such that "artificial aborts" are added to make sure that the probability of aborts is exactly equal to some negligible upper bound for the probability that E occurs for any set of oracle queries.

Conditioning on $\Pr[J_v(w^*) = 0 \bmod p]$ the theoretical lower bound of $\Pr[J_v(w^*) \neq 0 \bmod p]$ is $(1 - \frac{q_D}{\rho_v})$. Setting $\rho_v = 2q_D$ and will make it bounded by $1/2$. On the other hand, a lower bound on the probability for the first event is $\frac{1}{2(4\ell q_E 2^{n/\ell})^h}$ by setting $\rho_u = 4q_E$ [17].

We estimate the probability that $\mathcal{A}$'s oracle queries will cause $\mathcal{S}$ to abort by repeatedly sampling values determining $J_{u_1}(\cdot), \cdots, J_{u_h}(\cdot), J_v(\cdot)$. This would not involve re-running $\mathcal{A}$ as $\mathcal{A}$'s view (of the public parameters) remains unchanged by assuming y's are changing accordingly. Waters [46] has shown that a polynomial number of trials is sufficient to give an estimate of the abort probability η to within a negligible error term.

If $\mathcal{S}$ did not abort, we force an artificial abort with probability $(\eta - 1/(4(4\ell q_E 2^{n/\ell})^h))/\eta$, and $\mathcal{S}$ will abort with probability sufficiently close to $\frac{1}{4(4\ell q_E 2^{n/\ell})^h}$. Now we can say $\Pr[S_5] = \Pr[S_4]/4(4\ell q_E 2^{n/\ell})^h$. An exposition of Waters' technique can be found at [18].

Game 6 (Simulation of Extraction and Decryption). This game changes the simulation of all $\mathcal{A}$'s queries for trapdoor extractions, partial decryptions, and complete decryptions. We will have $\Pr[S_6] = \Pr[S_5]$.

Trapdoor extraction: For trapdoor key extraction query of $\overrightarrow{ID} = (\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$ where $k \le h$. Let $j' \in \{1, \cdots, k\}$ be a minimum one such that $J_{u_{j'}}(\mathsf{ID}_{j'}) \ne 0$. There exists such a j' or $\mathcal{S}$ has aborted in Game 5. $\mathcal{S}$ needs to return $d_{\overrightarrow{ID}} = (a_1, a_2, \overrightarrow{z}_{k+1}, \cdots, \overrightarrow{z}_h)$.

We first show how to compute $a_{1|j'}$, a "trapdoor for only $\mathsf{ID}_{j'}$" (without any appearance of any elements from other levels); then we will show how to compute a trapdoor $(a_1, a_2, \overrightarrow{z}_{k+1}, \cdots, \overrightarrow{z}_h)$ that matches the same implicit random factor used in $a_{1|j'}$. Recall that $F_{\overrightarrow{U}_{j'}}(\mathsf{ID}_{j'}) = Z_{h-j'+1}^{J_{u_{j'}}(\mathsf{ID}_{j'})} \cdot g^{K_{u_{j'}}(\mathsf{ID}_{j'})}$. $\mathcal{S}$ picks $r \in \mathbb{Z}_p^*$ and computes

$$a_{1|j'} = (Z_1^{\beta} \cdot Z_{j'}^{-\frac{K_{u_{j'}}(\mathsf{ID}_{j'})}{J_{u_{j'}}(\mathsf{ID}_{j'})}}) \cdot (Z_{h-j'+1}^{J_{u_{j'}}(\mathsf{ID}_{j'})} \cdot g^{K_{u_{j'}}(\mathsf{ID}_{j'})})^r$$

The second component of $a_{1|j'}$ is only for randomization. We will show the first component of $a_{1|j'}$ is in the form of $g_2^{\alpha}(F_{\overrightarrow{U}_{j'}}(\mathsf{ID}_{j'}))^{-\frac{\alpha^{j'}}{J_{u_{j'}}(\mathsf{ID}_{j'})}}$, which means $a_{1|j'}$ is in the form of $g_2^{\alpha}(F_{\overrightarrow{U}_{j'}}(\mathsf{ID}_{j'}))^{\tilde{r}}$ where $\tilde{r} = r - \frac{\alpha^{j'}}{J_{u_{j'}}(\mathsf{ID}_{j'})}$.

$$\begin{aligned}
&g_2^{\alpha}(F_{\overrightarrow{U}_{j'}}(\mathsf{ID}_{j'}))^{-\frac{\alpha^{j'}}{J_{u_{j'}}(\mathsf{ID}_{j'})}} \\
&= (Z_h \cdot g^{\beta})^{\alpha}(Z_{h-j'+1}^{J_{u_{j'}}(\mathsf{ID}_{j'})} \cdot g^{K_{u_{j'}}(\mathsf{ID}_{j'})})^{-\frac{\alpha^{j'}}{J_{u_{j'}}(\mathsf{ID}_{j'})}} \\
&= Z_{h+1} \cdot Z_1^{\beta} \cdot Z_{h+1}^{-\frac{J_{u_{j'}}(\mathsf{ID}_{j'})}{J_{u_{j'}}(\mathsf{ID}_{j'})}} \cdot Z_{j'}^{-\frac{K_{u_{j'}}(\mathsf{ID}_{j'})}{J_{u_{j'}}(\mathsf{ID}_{j'})}} \\
&= Z_1^{\beta} \cdot Z_{j'}^{-\frac{K_{u_{j'}}(\mathsf{ID}_{j'})}{J_{u_{j'}}(\mathsf{ID}_{j'})}}
\end{aligned}$$

To compute $a_1 = g_2^{\alpha} \cdot (\prod_{j=1}^{k} F_{\overrightarrow{U}_j}(\mathsf{ID}_j))^{\tilde{r}}$, $\mathcal{S}$ needs to compute $F_{\overrightarrow{U}_j}(\mathsf{ID}_j)^{\tilde{r}} = (Z_{h-j+1}^{J_{u_j}(\mathsf{ID}_j)})^{\tilde{r}} \cdot (g^{K_{u_j}(\mathsf{ID}_j)})^{\tilde{r}}$ for $j \ne j'$. We would like to compute it without knowing α and Z_{h+1}, but with the help of $(Z_1, \cdots, Z_h)$. Now the only difficulty comes from the fact that $\alpha^{j'}$ in $\tilde{r}$ is unknown. Note that the second term $(g^{K_{u_j}(\mathsf{ID}_j)})^{\alpha^{j'}}$ can be computed from $Z_{j'}$. We can see how the first term can be obtained by considering two different cases.

1. $j < j'$: $J_{u_j}(\mathsf{ID}_j) = 0$ by the choice of j'.
2. $j > j'$: $Z_{h-j+1}^{\alpha^{j'}} = Z_{h+1-(j-j')}$, note that $1 \le j - j' \le h - 1$.

By similar reasoning, since $k + 1 > j'$, it is easy to see that $\overrightarrow{z}_{k+1}, \cdots, \overrightarrow{z}_h$ can also be computed from $(Z_1, \cdots, Z_h)$. This completes the simulation of the trapdoor queries.

SEM partial decryption: $\mathcal{S}$ performs the usual validity checking to reject any invalid ciphertext C that is purported to be encrypted under $\overrightarrow{ID}$ and pk. For decrypting a valid ciphertext with hash w by the trapdoor of $\overrightarrow{ID}$, if $d_{\overrightarrow{ID}} = (a_1, a_2, \cdots)$ is computable by $\mathcal{S}$, it is easy to generate $(a_1 F_{\overrightarrow{V}}(w)^t, a_2, g^t)$ for a random $t \in \mathbb{Z}_p^*$.

$\mathcal{S}$ cannot generate the trapdoor for $d_{\overrightarrow{ID}}$ only if $J_{u_1}(\mathsf{ID}_1) = \cdots = J_{u_k}(\mathsf{ID}_k) = 0 \bmod \rho_u$. Note that $J_v(w) \ne 0 \bmod \rho_v$ or $\mathcal{S}$ has aborted in Game 5. Under this condition, $\mathcal{S}$ can generate the token similar to the generation of the trapdoor before. Recall that $F_{\overrightarrow{V}}(w) = (Z_h \cdot g^{\beta})^{J_v(w)} \cdot g^{K_v(w)}$, we have

$$\begin{aligned}
&g_2^{\alpha}(F_{\overrightarrow{V}}(w))^{-\frac{\alpha}{J_v(w)}} \\
&= (Z_h \cdot g^{\beta})^{\alpha}(Z_h^{J_v(w)} \cdot (g^{\beta})^{J_v(w)} \cdot g^{K_v(w)})^{-\frac{\alpha}{J_v(w)}} \\
&= Z_{h+1} \cdot Z_1^{\beta} \cdot Z_{h+1}^{-\frac{J_v(w)}{J_v(w)}} \cdot Z_1^{-\beta\frac{J_v(w)}{J_v(w)}} \cdot Z_1^{-\frac{K_v(w)}{J_v(w)}} \\
&= Z_1^{-\frac{K_v(w)}{J_v(w)}}
\end{aligned}$$

This means $Z_1^{-\frac{K_v(w)}{J_v(w)}}$ gives a token with the implicit random factor equals to $-\frac{\alpha}{J_v(w)}$. Randomization can be done easily by multiplying the above term by $(F_{\vec{V}}(w))^r$ where $r \in \mathbb{Z}_p^*$. Since $J_{u_1}(\mathsf{ID}_1) = \cdots = J_{u_k}(\mathsf{ID}_k) = 0 \bmod \rho_u$, all $\frac{\alpha}{J_v(w)}$ power terms appear in the construction of the token can be computed from Z_1.

User partial decryption: $\mathcal{A}$ queries $\mathcal{S}$'s oracle $\mathsf{DecO}^U(C, \mathsf{pk}, D)$. $\mathcal{S}$ performs the usual ciphertext validity checking to reject any invalid ciphertext C that is purported to be encrypted under $\overrightarrow{ID}$ and pk, and the token validity checking to reject any invalid token D that is purported to be a partial decryption of C. These validity checks prevent loss of information about the secret key sk. In particular, without the token checking, it is trivial for a Type-II adversary to derive the message in the challenge ciphertext by asking $\mathsf{DecO}^U(C^*, \mathsf{pk}^*, D')$ where D' is some invalid token derived from a valid one.

Suppose C is a valid ciphertext and D is valid for C and $\overrightarrow{ID}$, $\mathsf{DecO}^U(C, \mathsf{pk}, D)$ should give a correct decryption. For decrypting a valid ciphertext (C_1, C_2, τ, σ) with hash w, we have $\tau = g^s$ and $\sigma = F_{\vec{V}}(w)^s$ for some $s \in \mathbb{Z}_p^*$, i.e. $\sigma = g_2^{s \cdot J_v(w)} \cdot (g^s)^{K_v(w)}$. $\mathcal{S}$ can get g_2^s by $(\sigma/\tau^{K_v(w)})^{\frac{1}{J_v(w)}}$, $\hat{e}(Y, g_2)^s$ can thus be computed easily. Note that the secret key sk that matches pk is never explicitly used.

Complete decryption: After validity checking, $\mathcal{S}$ returns $m = C_1/\hat{e}(Y, (\sigma/\tau^{K_v(w)})^{\frac{1}{J_v(w)}})$ for valid ciphertext.

Game 7 (Simulation of the Ciphertext / Embedding of the Problem Instance). Depending on whether the adversary is an insider or the server, we have different modes of simulations. Now $\mathcal{S}$ introduces a variable $\gamma \in_R \mathbb{Z}_p^*$ and sets $\tau^* = g^\gamma$.

If $mode$ = I, g_1 is set to $Z_1 = g^\alpha$. $\mathcal{A}$ chooses an identifier $\overrightarrow{ID}^* = (\mathsf{ID}_1^*, \cdots, \mathsf{ID}_k^*)$, a public key $\mathsf{pk}^* = (X^*, Y^*)$ to be challenged with. $\mathcal{S}$ proceeds if $\hat{e}(Y^*, g) = \hat{e}(X^*, g_1)$. Let $T = (g^{\alpha^{h+1}})^\gamma$, $\mathcal{S}$ computes C_1^* by

$$\begin{aligned}
& m_\iota \cdot \hat{e}(X^*, T) \cdot (Y^*, g^\gamma)^\beta \\
= {} & m_\iota \cdot \hat{e}(X^*, (g^{\alpha^{h+1}})^\gamma) \cdot (Y^*, g^\gamma)^\beta \\
= {} & m_\iota \cdot \hat{e}(Y^*, g^{\alpha^h})^\gamma \cdot (Y^*, g^\beta)^\gamma \\
= {} & m_\iota \cdot \hat{e}(Y^*, Z_h \cdot g^\beta)^\gamma \\
= {} & m_\iota \cdot \hat{e}(Y^*, g_2)^\gamma
\end{aligned}$$

Note that it is the first time in the simulation that β is used directly (i.e. not in the form of g^β).

If $mode$ = II, $\mathcal{S}$ introduces a variable $\delta \in_R \mathbb{Z}_p^*$. Since $\mathcal{A}$ is a Type-II adversary, it can only choose a public key from $\{\mathsf{pk}_1^*, \cdots \mathsf{pk}_{q_K}^*\}$ given in aux to attack. $\mathcal{A}$ chooses an identifier $\overrightarrow{ID}^*$ and a public key $\mathsf{pk}_i^* = (X_i^*, Y_i^*) = ((g^\delta)^{\theta_i}, (g^\delta)^{\theta_i \alpha})$ to be challenged with. Note that the choice of $\theta_i \in_R \mathbb{Z}_p^*$ is known to $\mathcal{S}$ since it was $\mathcal{S}$ who prepared it. Let $T = g^{\beta\gamma\delta}$, $\mathcal{S}$ computes C_1^* by

$$\begin{aligned}
& m_\iota \cdot (\hat{e}(g^\delta, g^\gamma)^{\alpha^{h+1}} \cdot \hat{e}(g^\alpha, T))^{\theta_i} \\
= {} & m_\iota \cdot (\hat{e}(g^\delta, g^{\alpha^{h+1}})^\gamma \cdot \hat{e}(g^\alpha, g^{\beta\gamma\delta}))^{\theta_i} \\
= {} & m_\iota \cdot \hat{e}(g^{\delta\theta_i\alpha}, g^{\alpha^h})^\gamma \cdot \hat{e}(g^{\delta\theta_i\alpha}, g^\beta)^\gamma \\
= {} & m_\iota \cdot \hat{e}(Y^*, Z_h \cdot g^\beta)^\gamma \\
= {} & m_\iota \cdot \hat{e}(Y^*, g_2)^\gamma
\end{aligned}$$

Note that it is the first time in the simulation that α is used directly (i.e. not in the form of $g^\alpha, \cdots, g^{\alpha^h}$).

In both modes, $\mathcal{S}$ sets $C_2^* = \prod_{j=1}^k (g^\gamma)^{K_{u_j}(\mathsf{ID}_j^*)}$, $\sigma^* = (g^\gamma)^{K_v(w^*)}$ where $w^* = H(C_1^*, C_2^*, \tau^*, \overrightarrow{ID}^*, \mathsf{pk}^*)$ for the rest of the challenge, which is a perfect simulation if $\mathcal{S}$ did not abort in Game 4. We have $\Pr[S_7] = \Pr[S_6]$.

Game 8 (The Indistinguishability Cards). If $mode$ = I, $\mathcal{S}$ forgets (α, γ). If $mode$ = II, $\mathcal{S}$ forgets (β, γ, δ). Note that $\mathcal{S}$ can simulate the game in both modes as long as $(g^\alpha, \cdots, g^{\alpha^h}, g^\gamma)$ are known for $mode$ = I or

$(g^\beta, g^\gamma, g^\delta)$ are known for $mode$ = II, except computing the term T. Now $\mathcal{S}$ just picks a $T \in_R \mathbb{G}$. The transition from Game 7 to Game 8 is based on the intractability of either h-wBDHI' or 3-DDH. Both games are equal unless there exists a PPT algorithm $\mathcal{D}$ that distinguishes T from random. Therefore, we have $|\Pr[S_8] - \Pr[S_7]| \leq Adv_{\mathcal{D}}^{\mathrm{X}}(k)$ where X is either h-wBDHI' or 3-DDH. Finally, C_1^* perfectly hides m_ι from $\mathcal{A}$, we have $\Pr[S_8] = 1/2$.

B Security Models of Timed-Release Encryption

We consider the two kinds of adversaries. A Type-I adversary models any coalition of rogue users, and who aims to break the confidentiality of another user's ciphertext. A Type-II adversary that models a curious time server, who aims to break the confidentiality of an user's ciphertext. Security against these adversaries are modeled by the experiment below for $\mathsf{X} \in \{\mathrm{I}, \mathrm{II}\}$, denoting whether an PPT adversary $\mathcal{A} = (\mathcal{A}_{\mathsf{find}}, \mathcal{A}_{\mathsf{guess}})$ is of Type-I or Type-II. The allowed oracle queries $\mathcal{O}$ and the auxiliary information Aux depends on X.

Definition 9. Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA}'-\mathsf{X}}(\lambda)$

$(\mathsf{Pub}, \mathsf{Msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$(m_0, m_1, \mathsf{pk}^*, \mathsf{ID}^*, \mathsf{state}) \xleftarrow{\$} \mathcal{A}_{\mathsf{find}}^{\mathcal{O}}(\mathsf{Pub}, \mathsf{Aux})$
$b \xleftarrow{\$} \{0,1\}$, $C^* \xleftarrow{\$} \mathsf{Enc}(m_b, \mathsf{ID}^*, \mathsf{pk}^*)$
$b' \xleftarrow{\$} \mathcal{A}_{\mathsf{guess}}^{\mathcal{O}}(C^*, \mathsf{state})$
If $(|m_0| \neq |m_1|) \vee (b \neq b')$ *then return 0 else return 1*

where $\mathcal{O}$ *refers to a set of four oracles* $\mathsf{Extract}(\cdot), \mathsf{Dec}^S(\cdot,\cdot), \mathsf{Dec}^U(\cdot,\cdot,\cdot), \mathsf{Dec}(\cdot,\cdot,\cdot)$ *defined as below.*

1. *An* $\mathsf{Extract}$ *oracle that takes an identifier* $\mathsf{ID} \in \{0,1\}^n$ *as input and returns its trapdoor* d_{ID}.
2. *A* Dec^S *oracle that takes a ciphertext* C *and an identifier* ID, *and outputs* $\mathsf{Dec}^S(C, d_{\mathsf{ID}})$. *Note that* C *may or may not be encrypted under* ID.
3. *A* Dec^U *oracle that takes a ciphertext* C, *a public key* pk *and a token* D, *and outputs* $\mathsf{Dec}^U(C, \mathsf{sk}, D)$ *where* sk *is the secret key that matches* pk.
4. *A* Dec *oracle that takes a ciphertext* C, *an identifier* ID, *and a public key* pk, *and outputs* $\mathsf{Dec}^U(C, \mathsf{sk}, D)$ *where* sk *is the secret key that matches* pk *and* $D = \mathsf{Dec}^S(C, d_{\mathsf{ID}})$. *Note that* C *may or may not be encrypted under* ID *and* pk.

Definition 10. *A timed-release encryption scheme is* (t, q_E, q_D, ϵ) *IND-CCA secure against a Type-I adversary if* $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA}'-\mathsf{I}}(\lambda) = 1] - \frac{1}{2}| \leq \epsilon$ *for all* t*-time adversary* $\mathcal{A}$ *making at most* q_E *extraction queries and* q_D *decryption queries (of any type), subjects to the following constraints:*

1. $\mathsf{Aux} = \emptyset$, *i.e. no auxiliary information is given to the adversary.*
2. *No* $\mathsf{Extract}(\mathsf{ID}^*)$ *query throughout the game.*
3. *No* $\mathsf{Dec}^S(C^*, \mathsf{ID}^*)$ *query throughout the game.*
4. *No* $\mathsf{Dec}(C^*, \mathsf{ID}^*, \mathsf{pk}^*)$ *query throughout the game.*

Definition 11. *A timed-release encryption scheme is* (t, q_E, q_D, ϵ) *IND-CCA secure against a Type-II adversary if* $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{CCA}'-\mathsf{II}}(\lambda) = 1] - \frac{1}{2}| \leq \epsilon$ *for all* t*-time adversary* $\mathcal{A}$ *making at most* q_K *public key queries,* q_E *extraction queries and* q_D *decryption queries (of any type), subjects to the following conditions:*

1. $\mathsf{Aux} = (\mathsf{Msk}, \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\})$, *i.e. the master secret key and a set of challenge public key* pk^* *is given to the adversary.*
2. $\mathsf{pk}^* \in \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\}$, *i.e. the challenge public key must be among the set given by the challenger.*
3. *No* $\mathsf{Dec}^U(C^*, \mathsf{pk}^*, D)$ *query throughout the game, where* D *is obtained from* $\mathsf{Dec}^S(C^*, \mathsf{ID}^*)$.
4. *No* $\mathsf{Dec}(C^*, \mathsf{ID}^*, \mathsf{pk}^*)$ *query throughout the game.*

C Pre-open Capability

In many applications of TRE, it is desirable to have a pre-open mechanism that the sender can enable the recipient to decrypt the ciphertext before the pre-specified release-time, without re-sending the plaintext. In a TRE with such a pre-open capability [34], the sender gets hold of a pre-open key that can functionally substitute the role of the system's time-dependent trapdoor, for the ciphertext prepared by him/her.

The concept of pre-open capability is introduced to the TRE paradigm by [34]. However, the scheme of [34] does not consider the security threat that the sender can give a pre-open key which opens the ciphertext to another message that is different from the one originally being encrypted. This deficiency is pointed out by [25], where the property of binding is formally defined and a scheme with binding pre-open key is proposed.

C.1 Model

We chose not to cover pre-open capability in the our general model because we do not think it makes a good sense in the context of CLE. The syntactic changes for pre-open capability include:

- Enc (run by a sender) is a probabilistic algorithm which takes a message m from some implicit message space, an identifier $\overrightarrow{ID} \in (\{0,1\}^n)^{\leq h}$, and the receiver's public key pk_u as input, returns a ciphertext C and its pre-open key D_C.
- $\mathsf{PreOpen}$ (run by a receiver) is a possibly probabilistic algorithm which takes the ciphertext C, the receiver's private key sk_u and a pre-open key D_C as input, returns either the plaintext, an invalid flag $\perp_D$ denoting D_C is an invalid pre-open key, or an invalid flag $\perp_C$ denoting the ciphertext is invalid.

Correctness requires $\mathsf{PreOpen}(C, \mathsf{sk}, D_C) = m$ for all $\lambda, n \in \mathbb{N}$, all Pub given by $\mathsf{Setup}(1^\lambda, n)$, all $(\mathsf{pk}, \mathsf{sk})$ given by KeyGen, all message m, all time $\overrightarrow{ID}$ in $(\{0,1\}^n)^{\leq h}$ and all (C, D_C) given by $\mathsf{Enc}(m, \overrightarrow{ID}, \mathsf{pk})$.

Binding requires the following probability is negligible for all PPT algorithm $\mathcal{A}$.

$$\begin{aligned}\Pr[(C^*, \mathsf{ID}^*, D_C^* \xleftarrow{\$} \mathcal{A}(\mathsf{Pub}) \mid (\mathsf{Pub}, \mathsf{Msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)\\ \wedge\ \mathsf{PreOpen}(C^*, \mathsf{sk}, D_C^*) \notin \{\perp_D, \perp_C\}\\ \wedge\ \mathsf{PreOpen}(C^*, \mathsf{sk}, D_C^*) \neq \mathsf{Dec}^U(C^*, \mathsf{sk}, \mathsf{Dec}^S(C^*, \mathsf{Extract}(\mathsf{Msk}, \overrightarrow{ID^*})))]\end{aligned}$$

C.2 Construction

To make our concrete scheme supports pre-open capability, Enc just outputs $D_C = g_1^s$ as the pre-open key, where s is the random factor chosen in Enc. The pre-open mechanism is defined as below.

$\mathsf{PreOpen}(C, \mathsf{sk}, D_C)$: Firstly, check if the pre-open key is valid by $\hat{e}(D_C, g) = \hat{e}(g_1, \tau)$, returns $\perp_D$ if the equality does not hold. Otherwise, parse C as (C_1, C_2, τ, σ) and check if $\hat{e}(\tau, \prod_{j=1}^{k} F_{\vec{U}_j}(\mathsf{ID}_j) \cdot F_{\vec{V}}(w')) = \hat{e}(g, C_2 \cdot \sigma)$ where $w' = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk})$. Return $\perp_C$ if parsing is not possible or the equality does not hold. Otherwise, return $m \leftarrow C_1 / \hat{e}(D_C, g_2)^{\mathsf{sk}}$.

C.3 Security

A Type-I adversary is not entitled to have the pre-open key. We show that the addition of pre-open key will not compromise the confidentiality of the scheme against a time-server adversary. Using the knowledge of α (which is known to a Type-II adversary), the pre-open key of the challenge ciphertext can be easily computed by $(g^\gamma)^\alpha = g_1{}^\gamma$.

Next, we need to show it is binding. Given $(\overrightarrow{ID}, \mathsf{pk})$, the random factor in a valid ciphertext is uniquely fixed. From the pre-open key validity checking $\hat{e}(D_C, g) = \hat{e}(g_1, \tau)$ and the bilinearity, D_C must be in a correct form. Hence, the probability for breaking the binding property is zero.

D Release-time Confidentiality and Recipient Anonymity

D.1 Release-time Confidentiality

Release-time confidentiality protects the ciphertext release-time from being known to anyone but the recipient. In the context of CLE, this property means recipient-ID anonymity.

Naturally, one will consider an anonymous IBE (e.g. [10, 12, 29]), where the ciphertext does not reveal any information about its intended recipient. However, we can leverage the fact that a kind of double-encryption is done in TRE.

In the context of TRE, even the ciphertext may not leak any information about the release-time, the release-time should be sent to the intended recipient. So we add into our framework another algorithm called GetID for this purpose.

Model. The algorithm GetID is one executed by the intended recipient who holds the user private key and the ciphertext, but not the time-dependent trapdoor. This allows the intended recipient to get the time-identifier from the ciphertext by using the user secret key. The correctness requirement is as follows.

- $\mathsf{GetID}(\mathsf{Enc}(m, \overrightarrow{ID}, \mathsf{pk}), \mathsf{sk}) = \overrightarrow{ID}, C$ and $\mathsf{Dec}^U(C, \mathsf{sk}, \mathsf{Dec}^S(C, \mathsf{Extract}(\mathsf{Msk}, \overrightarrow{ID}))) = m$ for all $\ell, n \in \mathbb{N}$, all Pub given by $\mathsf{Setup}(1^\ell, n)$, all $(\mathsf{pk}, \mathsf{sk})$ given by KeyGen, all message m, and all identifier $\overrightarrow{ID}$ in $(\{0,1\}^n)^{\leq h}$.

Since $\overrightarrow{ID}$ is hidden in the ciphertext (in particular, hidden from the SEM), a ciphertext should be transformed by the sk so that the SEM can check whether the transformed ciphertext is one intended for the purported identifier.

The formal security requirement is similar to that in [14], but on top of that we need to add an GetID oracle that takes a ciphertext and a public key of the adversary's choice. Basically, the challenge ciphertext is not encrypting one of the two messages under the fixed identifier all given by the adversary, but encrypting a fixed message under one of the two identifiers. The adversary's goal is to tell which random identifier is employed by the challenger.

Security is only defined against a Type-II adversary. For any Type-I adversary, it can replace the public key, and hence obtaining $\overrightarrow{ID}$ from $\mathsf{GetID}(C, \mathsf{sk})$ is trivial.

Definition 12. Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{RTC-II}}(\lambda)$

$(\mathsf{Pub}, \mathsf{Msk}) \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
$(m, \mathsf{pk}^*, \overrightarrow{ID}_0^*, \overrightarrow{ID}_1^*, \mathsf{state}) \xleftarrow{\$} \mathcal{A}_{\mathsf{find}}^{\mathcal{O}}(\mathsf{Pub}, \mathsf{Aux})$
$b \xleftarrow{\$} \{0,1\}$, $C^* \xleftarrow{\$} \mathsf{Enc}(m, \overrightarrow{ID}_b^*, \mathsf{pk}^*)$
$b' \xleftarrow{\$} \mathcal{A}_{\mathsf{guess}}^{\mathcal{O}}(C^*, \mathsf{state})$
If $b \neq b'$ *then return 0 else return 1*

where $\mathcal{O}$ *refers to a set of five oracles* $\mathsf{Extract}(\cdot), \mathsf{GetID}(\cdot,\cdot), \mathsf{Dec}^S(\cdot,\cdot), \mathsf{Dec}^U(\cdot,\cdot,\cdot), \mathsf{Dec}(\cdot,\cdot,\cdot)$.

Definition 13. *A hierarchical security-mediated certificateless encryption scheme is* (t, q_E, q_D, ϵ) *RTC-CCA secure against a Type-II adversary if* $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{RTC-II}}(\lambda) = 1] - \frac{1}{2}| \leq \epsilon$ *for all* t*-time adversary* $\mathcal{A}$ *making at most* q_K *public key queries,* q_E *extraction queries and* q_D *decryption queries (of any type), subjects to the following conditions:*

1. $\mathsf{Aux} = (\mathsf{Msk}, \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\})$*, i.e. the master secret key and a set of challenge public key* pk^* *is given to the adversary.*
2. $\mathsf{pk}^* \in \{\mathsf{pk}_1^*, \cdots, \mathsf{pk}_{q_K}^*\}$*, i.e. the challenge public key must be among the set given by the challenger.*
3. *No* $\mathsf{GetID}(C^*, \mathsf{pk}^*)$ *query throughout the game.*
4. *No* $\mathsf{Dec}^U(C^*, \mathsf{pk}^*, D)$ *query throughout the game, where* D *is obtained from* $\mathsf{Dec}^S(C^*, \overrightarrow{ID}_0^*)$ *or* $\mathsf{Dec}^S(C^*, \overrightarrow{ID}_1^*)$.
5. *No* $\mathsf{Dec}(C^*, \overrightarrow{ID}^*, \mathsf{pk}^*)$ *query throughout the game, where* $\overrightarrow{ID}^* \in \{\overrightarrow{ID}_0^*, \overrightarrow{ID}_1^*\}$.

Construction. We need the help of a key-derivation function $K : \mathbb{G}_T \to \{0,1\}^{n \cdot h+k+1}$, which we assume the output of K is computationally indistinguishable from a random distribution when the input comes from a uniformly distribution. We also assume an implicit one-to-one mapping between $\mathbb{G}$ and $\{0,1\}^{k+1}$, i.e. the public parameters Pub is given by

$$\mathsf{Pub} = (\mathbb{G}, \mathbb{G}_T, \hat{e}(\cdot,\cdot), n, g, g_1, g_2, \vec{U}_1, \cdots, \vec{U}_h, \vec{V}, H(\cdot), K(\cdot))$$

The modified encryption algorithm Enc and the GetID algorithm are described as follows.
$\mathsf{Enc}(m, \overrightarrow{ID}, \mathsf{pk})$: To encrypt $m \in \mathbb{G}_T$ for $\overrightarrow{ID} = (\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$ where $k \leq h$, parse pk as (X, Y), then check that it is a valid public key by $\hat{e}(X, g_1) = \hat{e}(g, Y)$. If equality holds, pick $s \in_R \mathbb{Z}_p^*$ and compute

$$\begin{aligned} C &= (C_1, C_2, \tau, \sigma) \\ &= (m \cdot \hat{e}(Y, g_2)^s, (\overrightarrow{ID} || \prod_{j=1}^{k} F_{\vec{U}_j}(\mathsf{ID}_j)^s) \oplus K(\hat{e}(X, g_2)^s), g^s, F_{\vec{V}}(w)^s) \end{aligned}$$

where $w = H(C_1, C_2, \tau, \overrightarrow{ID}, \mathsf{pk}, K(\hat{e}(X, g_2)^s))$.

$\mathsf{GetID}(C, \mathsf{sk})$: Parse C as (C_1, C_2, τ, σ), $C_2 \oplus K(\hat{e}(\tau, g_2)^{\mathsf{sk}})$ as $(\overrightarrow{ID}' || f')$, and $\overrightarrow{ID}'$ as $(\mathsf{ID}_1, \cdots, \mathsf{ID}_k)$; check if $\hat{e}(\tau, \prod_{j=1}^{k} F_{\vec{U}_j}(\mathsf{ID}_j) \cdot F_{\vec{V}}(w')) = \hat{e}(g, f'\sigma)$ where $w' = H(C_1, C_2, \tau, \overrightarrow{ID}', \mathsf{pk}, K(\hat{e}(\tau, g_2)^{\mathsf{sk}}))$. Return $\perp$ if inequality holds or any parsing is not possible, otherwise return $\overrightarrow{ID}', C_1, f', \tau, \sigma$.

Security. We have the following theorems for the security of our modified scheme.

Theorem 6 *Our modified scheme is secure against Type-I attack (Definition 3) if h-wBDHI' problem is intractable.*

Theorem 7 *Our modified scheme is secure against Type-II attack (Definition 4) if 3-DDH problem is intractable.*

Theorem 8 *Our modified scheme is* $\mathsf{RTC\text{-}II}$*-secure (Definition 12) if 3-DDH problem is intractable.*

Here we only highlight the changes we need to make in our main proof in Appendix A. The new things in the security proof are the simulation of the challenge ciphertext in a new format, all decryption oracles including GetID, and the new well-formness checking of the ciphertext (since w is the output of the hash H which now takes $\hat{e}(X, g_2)^s$ as part of the input, but not just public information).

Note that the padding for encrypting the message is $\hat{e}(Y, g_2)^s$ while the padding we introduced for hiding all the information related to the identifier is $\hat{e}(X, g_2)^s$.

Even though the term $\prod_{j=1}^{k} F_{\vec{U}_j}(\mathsf{ID}_j)^s$ is now hidden by $K(\hat{e}(X, g_2)^s)$ and it seems that the ciphertext validity cannot be checked, the simulator (in Game 6) only takes C_1 and τ to compute g_2^s, and hence the term $K(\hat{e}(X, g_2)^s)$ can be recovered. Specifically, for user partial decryption, $\mathcal{S}$ computes $\hat{e}(Y, g_2)^s$ as $\hat{e}(Y, (\sigma/\tau^{K_v(w)})^{\frac{1}{J_v(w)}})$, $\hat{e}(X, g_2)^s$ can thus easily computed as $\hat{e}(X, (\sigma/\tau^{K_v(w)})^{\frac{1}{J_v(w)}})$. The same is true for computing w for well-formness checking.

Intuitively, the malleable XOR cipher can be used in C_2 since the decryption algorithms checks the σ term, which is computed from the hash taking C_2 as part of the input.

Although the release-time confidentiality is defined against Type-II attack, we still need to show the simulation goes through for Type-I proof for CCA-security against Type-I adversary. The simulation of the new ciphertext (in Game 7) is done in different way according to the type of the adversary. For Type-I, $\mathcal{S}$ computes $\hat{e}(X, g_2)^s$ by $\hat{e}(X^*, (g^{\alpha^h})^\gamma) \cdot (X^*, g^\gamma)^\beta = \hat{e}(X^*, Z_h \cdot g^\beta)^\gamma = \hat{e}(X^*, g_2)^\gamma$. For Type-II, by the relationship that $Y^* = (X^*)^\alpha$, $\mathcal{S}$ can easily obtain $\hat{e}(X^*, g_2)^s$ by $(\hat{e}(Y^*, g_2)^s)^{\frac{1}{\alpha}}$.

D.2 Recipient Anonymity

Release-time is not the only dimension about which key can be used to decrypt the ciphertext. It may be possible that information about the intended recipient is leaked from the ciphertext too, which against the requirement of recipient anonymity. In the context of CLE, release-time confidentiality and recipient anonymity ensures that no one can tell who is the intended recipient of a CLE-encrypted ciphertext.

The formal security requirement is similar to the release-time confidentiality. The adversary chooses two public keys to be challenged with. The challenger encrypts a message chosen by the adversary under a random key among the two. The adversary's goal is to tell which key is employed by the challenger.

Definition 14. Experiment $\mathbf{Exp}_{\mathcal{A}}^{\mathsf{RKA-II}}(\lambda)$

$(\mathsf{Pub}, \mathsf{Msk}) \stackrel{\$}{\leftarrow} \mathsf{Setup}(1^\lambda)$
$(m, \mathsf{pk}'_0, \mathsf{pk}'_1, \overrightarrow{ID}^*, \mathsf{state}) \stackrel{\$}{\leftarrow} \mathcal{A}_{\mathsf{find}}^{\mathcal{O}}(\mathsf{Pub}, \mathsf{Aux})$
$b \stackrel{\$}{\leftarrow} \{0,1\}$, $C^* \stackrel{\$}{\leftarrow} \mathsf{Enc}(m, \overrightarrow{ID}^*, \mathsf{pk}'_b)$
$b' \stackrel{\$}{\leftarrow} \mathcal{A}_{\mathsf{guess}}^{\mathcal{O}}(C^*, \mathsf{state})$
If $b \neq b'$ then return 0 else return 1

where $\mathcal{O}$ refers to a set of four oracles $\mathsf{Extract}(\cdot), \mathsf{GetID}(\cdot,\cdot), \mathsf{Dec}^S(\cdot,\cdot), \mathsf{Dec}^U(\cdot,\cdot,\cdot), \mathsf{Dec}(\cdot,\cdot,\cdot)$.

Definition 15. *A hierarchical security-mediated certificateless encryption scheme is (t, q_E, q_D, ϵ) RKA-CCA secure against a Type-II adversary if $|\Pr[\mathbf{Exp}_{\mathcal{A}}^{\mathsf{RKA-II}}(\lambda) = 1] - \frac{1}{2}| \leq \epsilon$ for all t-time adversary $\mathcal{A}$ making at most q_K public key queries, q_E extraction queries and q_D decryption queries (of any type), subjects to the following conditions:*

1. $\mathsf{Aux} = (\mathsf{Msk}, \{\mathsf{pk}^*_1, \cdots, \mathsf{pk}^*_{q_K}\})$, *i.e. the master secret key and a set of challenge public key pk^* is given to the adversary.*
2. $\mathsf{pk}'_0, \mathsf{pk}'_1 \in \{\mathsf{pk}^*_1, \cdots, \mathsf{pk}^*_{q_K}\}$, *i.e. the challenge pair of public keys must be among the set given by the challenger.*
3. *No $\mathsf{GetID}(C^*, \mathsf{pk}')$ query throughout the game, where $\mathsf{pk}' \in \{\mathsf{pk}'_0, \mathsf{pk}'_1\}$.*
4. *No $\mathsf{Dec}^U(C^*, \mathsf{pk}', D)$ query throughout the game, where D is obtained from $\mathsf{Dec}^S(C^*, \overrightarrow{ID}^*)$ and where $\mathsf{pk}' \in \{\mathsf{pk}'_0, \mathsf{pk}'_1\}$.*
5. *No $\mathsf{Dec}(C^*, \overrightarrow{ID}^*, \mathsf{pk}')$ query throughout the game, where $\mathsf{pk}' \in \{\mathsf{pk}'_0, \mathsf{pk}'_1\}$.*

Security. The recipient anonymity of our scheme can be easily seen in the Game 7 and 8 of our proof. The intended recipient of the ciphertext is uniquely determined by θ_i. When T is a random element, θ_i is perfectly hidden.

E Analysis of A Recent Timed-Release Encryption Scheme

E.1 Review

We first review how the ciphertext is constructed in the TRE scheme proposed by Chalkias *et al.* [15].

$\mathsf{Setup}(1^\lambda, 1^{\lambda_0})$: Given security parameters λ and λ_0, where λ_0 is a polynomially-bounded function of λ, Let $\mathbb{G}$, $\mathbb{G}_T$ be two multiplicative groups with a bilinear map $\hat{e}$ as defined before. They are of the same order p, which is a prime and $2^\lambda < p < 2^{\lambda+1}$. The public parameters Pub and the master secret key Msk are given by

$$\mathsf{Pub} = (\lambda, \lambda_0, p, \mathbb{G}, \mathbb{G}_T, \hat{e}(\cdot,\cdot), P, S = P^s, H_1(\cdot), H_2(\cdot), H_3(\cdot), H_4(\cdot)), \qquad \mathsf{Msk} = s \in_R \mathbb{Z}_p^*.$$

where P is an arbitrary generator of $\mathbb{G}$, $H_1(\cdot), H_2(\cdot), H_3(\cdot), H_4(\cdot)$ are cryptographic hash functions modeled as random oracles. Their domains and ranges will be clear from the description of the other algorithms.

$\mathsf{Extract}(\mathsf{Msk}, T)$: Given a time-identifier T, the time-dependent trapdoor is $d_T = P^{\frac{1}{(s+t)}}$, where $t = H_1(T)$.

$\mathsf{KeyGen}()$: Pick $\mathsf{sk} \in_R \mathbb{Z}_p^*$, return sk as the secret key and $\mathsf{pk} = g^{\mathsf{sk}}$ as the public key.

$\mathsf{Enc}(m, T, \mathsf{pk})$: Suppose the message m is encrypted under the time T and the public key pk, the algorithm proceeds as follow.

1. Compute $t = H_1(T) \in \mathbb{Z}_p^*$;
2. Choose $x \in_R \{0,1\}^{\lambda_0}$ and $h = H_2(m||x||T) \in \{0,1\}^{2\lambda}$;
3. Treat $\bar{h}$ as the 2λ-bit integer value of h, parse it as $r_1||r_2$, where $r_1, r_2 \in \mathbb{Z}_p^*$;
4. Compute $c_1 = (S \cdot P^t)^{r_1}$ and $c_2 = P^{r_2}$;
5. Compute $d = H_3(\hat{e}(P,P)^{r_1}) \in \mathbb{Z}_p^*$;
6. Compute $K = H_4(\mathsf{pk}^{(d \cdot r_2)})$ and $c_3 = (m||x||h) \oplus K$;
7. Return $C = (c_1, c_2, c_3, T)$.

$\mathsf{Dec}(C, d_T, \mathsf{sk})$: To decrypt the ciphertext $C = (c_1, c_2, c_3, T)$ using the trapdoor d_T and the secret key sk, the algorithm proceeds as follow.

1. Compute $d = H_3(\hat{e}(c_1, d_T))$;
2. Compute $K = H_4(c_2^{(d \cdot \mathsf{sk})})$;
3. Parse $c_3 \oplus K$ as $m||x||h$;
4. Return m if $H_2(m||x||T) = h$.

E.2 Attacks

Even though there is a checking of $H_2(m||x||T) = h$, there is no checking whether r_1 in $c_1 = (S \cdot P^t)^{r_1}$ and r_2 in $c_2 = P^{r_2}$ are really from the 2λ-bit integer value of h. Our attacks exploit this fact. Given the challenge ciphertext $C^* = (c_1^*, c_2^*, c_3^*, T^*)$ that is encrypted under the public key pk^*, our first attack proceeds as follows.

Attack 1 (with strong decryption oracle):

1. Randomly choose $z \in \mathbb{Z}_p$;
2. Compute $c_2' = {c_2^*}^z$;
3. Query the decryption oracle to decrypt $(c_1^*, c_2', c_3^*, T^*)$ with respect to the *replaced* public key ${\mathsf{pk}^*}^{1/z}$.

Suppose $\mathsf{pk}^* = g^{\mathsf{sk}^*}$, the adversary does not know the secret key sk^*/z corresponding to ${\mathsf{pk}^*}^{1/z}$. However, in the security model ([15, Definition 2]), the adversary is entitled with a decryption oracle that can decrypt any ciphertext except the challenge one, under any public key without supplying the corresponding private key. So this is a legitimate decryption query.

We claim that the decryption oracle will just return the message encrypted inside the challenge ciphertext. To see this, the decryption oracle computes $d = H_3(\hat{e}(c_1^*, d_{T^*}))$ and $K = H_4({c_2'}^{(d \cdot \mathsf{sk}')}) = H_4({c_2^*}^{z(d \cdot \mathsf{sk}^*/z)}) = H_4({c_2^*}^{(d \cdot \mathsf{sk}^*)})$, which is exactly the K computed by $\mathsf{Dec}(C^*, d_{T^*}, \mathsf{sk}^*)$.

Attack 2 (by a curious time-server):
Following the reasoning of the above attack, a curious time server can launch a similar attack without querying the decryption oracle with a replaced public key.

1. Compute $t^* = H_1(T^*) \in \mathbb{Z}_p^*$;
2. Randomly choose $z \in \mathbb{Z}_p$;
3. Compute $c_1' = (S \cdot P^{t^*})^z$
4. Compute $d' = H_3(\hat{e}(P,P)^z) \in \mathbb{Z}_p^*$;
5. Recover $d^* = H_3(\hat{e}(c_1^*, d_{T^*}))$;
6. Compute $c_2' = {c_2^*}^{(d^*/d')}$;

7. Query the decryption oracle to decrypt (c_1', c_2', c_3^*, T^*) with respect to *original* public key pk^*

To see the correctness, the decryption oracle computes $d = H_3(\hat{e}(c_1', d_{T^*})) = H_3(\hat{e}(P,P)^z) = d'$ and $K = H_4({c_2'}^{(d' \cdot \mathsf{sk}')}) = H_4({c_2^*}^{(d^* \cdot \mathsf{sk}^* \cdot d'/d')}) = H_4({c_2^*}^{(d^* \cdot \mathsf{sk}^*)})$, which is exactly the K computed by $\mathsf{Dec}(C^*, d_{T^*}, \mathsf{sk}^*)$.

It is possible to fix the scheme by requiring the decryption algorithm to return m if and only if $c_1 = (S \cdot P^t)^{r_1}$ and r_2 in $c_2 = P^{r_2}$ where $r_1 || r_2 = \bar{h}$ and $\bar{h}$ is the 2λ-bit integer value of h. However, it adds two exponentiations in the decryption algorithm and lessens the purported advantage of their scheme. We remark that the encryption algorithm is unaffected and is still more efficient than other existing schemes.

In the above attacks, the adversary never query H_4 directly; while in the proof of CCA security in [15], it is assumed that the adversary would have to request H_4 for a special value which let the simulator to solve the underlying computational problem.