Time-Specific Encryption*

Kenneth G. Paterson and Elizabeth A. Quaglia

Information Security Group,
Royal Holloway, University of London,
Kenny.Paterson@rhul.ac.uk, E.A.Quaglia@rhul.ac.uk

Abstract. This paper introduces and explores the new concept of Time-Specific Encryption (TSE). In (Plain) TSE, a Time Server broadcasts a key at the beginning of each time unit, a Time Instant Key (TIK). The sender of a message can specify any time interval during the encryption process; the receiver can decrypt to recover the message only if it has a TIK that corresponds to a time in that interval. We extend Plain TSE to the public-key and identity-based settings, where receivers are additionally equipped with private keys and either public keys or identities, and where decryption now requires the use of the private key as well as an appropriate TIK. We introduce security models for the plain, public-key and identity-based settings. We also provide constructions for schemes in the different settings, showing how to obtain Plain TSE using identity-based techniques, how to combine Plain TSE with public-key and identity-based encryption schemes, and how to build schemes that are chosen-ciphertext secure from schemes that are chosen-plaintext secure. Finally, we suggest applications for our new primitive, and discuss its relationships with existing primitives, such as Timed-Release Encryption and Broadcast Encryption.

* This research was sponsored in part by the US Army Research Laboratory and the UK Ministry of Defense and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the US Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

1 Introduction

Time has always played an important role in communication. Information can become useless after a certain point, sensitive data may not be released before a particular time, or we may wish to enable access to information for only a limited period of time. In this context, being able to specify during what time interval a ciphertext can be decrypted by a receiver is a useful and interesting property. In this paper, we introduce and develop a new cryptographic primitive called Time-Specific Encryption (TSE) which addresses this problem.

More specifically, we consider a setting in which we have a (semi-)trusted Time Server (TS). TS broadcasts a Time Instant Key (TIK) k_t at each time unit or "tick" of its clock, t, where $0 \leq t \leq T-1$. This TIK is available to all users, and we implicitly assume that it contains a description of t. A sender can specify any interval $[t_0, t_1]$, where $t_0 \leq t_1$, when encrypting a plaintext m to form a ciphertext c. In *Plain* TSE, we wish to achieve the property that c can only be decrypted by a receiver to recover m if the receiver is in possession of a TIK k_t for some t with $t \in [t_0, t_1]$. Notice that we cannot enforce the property that the receiver can only decrypt during the decryption time interval (DTI) $[t_0, t_1]$, since a receiver can always obtain an appropriate TIK and then use it at any time later on. Achieving this stronger notion could be done using trusted hardware, for example. Yet, as we discuss below, TSE has several intriguing applications exploiting its defining property that a receiver must obtain a suitable TIK before being able to decrypt.

We extend Plain TSE to the public-key and identity-based settings, where receivers are additionally equipped with private keys and either public keys or identities, and where decryption now requires the use of the relevant private key as well as an appropriate TIK. This provides protection against a curious Time Server, as well as ensuring that a ciphertext is decryptable only by a specified party. We introduce security models for the plain, public-key and identity-based settings, considering both chosen-plaintext and chosen-ciphertext adversaries.

We also provide constructions for schemes in the different settings. Firstly, we build Plain TSE schemes by adapting ideas of [25,27] which themselves employ identity-based and tree techniques. Secondly, we show how to combine Plain TSE with public-key and identity-based encryption schemes to obtain chosen-plaintext secure TSE schemes in the public-key and identity-based settings. Thirdly, we show how to adapt the CHK transform [8] to the TSE setting, obtaining a generic construction for a chosen-ciphertext secure TSE scheme in the public-key setting from a chosen-plaintext secure, identity-based TSE scheme. Our focus is on providing generic constructions that are secure in the standard model. Naturally, more efficient constructions and concrete schemes can be obtained by working in the Random Oracle Model (ROM), and we sketch such constructions where appropriate. In our closing section, we discuss possible extensions of our ideas and areas for future work.

1.1 Applications of TSE

TSE generalises Timed-Release Encryption (TRE), a concept first introduced in [23]. In TRE, a user can only decrypt *after* a specified release time. Existing approaches [9,10,20,15,12] to achieving TRE also make use of a trusted Time Server broadcasting time-specific keys, but suffer from the limitation that some back-up mechanism must be provided in case the receiver misses a key broadcast by the server. In this sense, TRE represents the special case of TSE in which the sender can specify only intervals of the form $[t, t]$. Typically in the literature, it is assumed that the Time Server (or some other agency) will make old keys available on a public server. Clearly this may be inconvenient and would require additional infrastructure on top of the broadcast capability. TSE provides an elegant solution to this problem: if the sender specifies an interval of the form $[t, T-1]$ (where $T-1$ is the maximum time supported by the scheme) then a receiver can decrypt using *any* TIK $k_{t'}$ broadcast by the Time Server at time $t' \geq t$. We note that the use of tree techniques to achieve this capability was sketched in [9,12], but without any formal security analysis. TSE, then, provides a useful extension of TRE that can be exploited in any of the many applications that have already been proposed for TRE in the literature, including electronic auctions, key escrow, on-line gaming, timed release of information such as press releases, and so on.

However, TSE is more flexible than this in the range of applications that it supports. For example, the encrypting party may specify an interval of the form $[0, t]$, meaning that a receiver can decrypt the

ciphertext as soon as it is received and a TIK has been obtained, but only up to time t. After this time, TIKs issued by the time server will not help in decryption. Yet, a user might obtain a useful TIK from some other user in the system, so this application of TSE only makes sense in situations where users have a vested interest in not sharing TIKs with one another, such as in situations where users are in competition with one another. For example, the ciphertext may encrypt a ticket for accessing a service that is valid up to time t. More generally, TSE can be used to support any application in which a user benefits from accessing plaintext in a timely manner, and where the utility of a TIK becomes limited shortly after its broadcast time. We sketch an example of such an application in the domain of entity authentication next.

Consider a typical time-stamp based network authentication protocol, in which entities A and B share a symmetric key K and in which A sends B messages of the form $\mathrm{MAC}_K(T||B)$ where T is the current time (at A) and MAC_K denotes a secure MAC algorithm using the key K. Such a protocol requires roughly synchronised clocks, and B needs to allow a "window of acceptance" for values T in A's messages, to cater for any loss of accuracy in synchronisation and network delay. In turn, this means that B needs to keep a log of recently received messages to prevent replays by an attacker during the window. How can TSE help? Suppose B generates a nonce N, encrypts it using a TSE scheme with an interval $[t_0, t_1]$, where $t_1 - t_0$ is equal to the width of a suitable window of acceptance (to cater for network delay and clock drift between A and B), and broadcasts the resulting ciphertext. Now A's ability to send a message of the form $\mathrm{MAC}_K(N||B)$ to B before time t_1 is a proof that A obtained a TIK k_t during the interval $[t_0, t_1]$ and decrypted to obtain the nonce N. Thus B obtains a proof of liveness of A within a certain window of acceptance, so authenticating A to B. This basic protocol can be extended in a number of ways. For example, B's ciphertexts can be pre-distributed to A, giving A a set of tokens which she can use to authenticate to B during specified time intervals. We can also adapt the basic scheme to use pseudo-randomly generated nonces, so saving state at B. We can modify it to provide key transport, by replacing the MAC with an authenticated encryption primitive and including a session key in A's message. We can also add mutual authentication in obvious ways. But what is notable about the protocol design is that we no longer require synchronised clocks, and we have a window of acceptance for responses by design. These features arise from the use of TSE.

1.2 Further Related Work

Range queries over encrypted data and related ideas: Shi *et al.* [25] proposed schemes enabling multi-dimensional range queries over encrypted data (MRQED). In the one-dimensional version of this primitive, data is associated with a single value and is encrypted with respect to that value, while users are equipped with keys enabling them to decrypt data whose values are in a given range. In contrast, in TSE, encryption is performed with respect to a range, while the Time Server makes available keys specific to a particular time value. Thus, our notion of Plain TSE is precisely equivalent to the notion of dual MRQED, introduced but not formalised by Shi *et al.* [25]. We note that [25] gives a construction which builds a dual MRQED scheme from a normal MRQED scheme, but this seems to involve a doubling of dimension and, therefore, a significant loss of efficiency, a problem from which our constructions do not suffer. In addition, in our work, we give constructions achieving non-selective security against chosen-ciphertext attackers, whereas [25] only considers selective security notions and chosen-plaintext attackers in any detail (and then in the MRQED setting rather than its dual). Moreover, we consider plain, public-key and identity-based settings, whereas [25] only handles what amounts to the plain setting. In work related to that of Shi *et al.*, Srivatsa *et al.* introduced [27] Trust-and-Identity Based Encryption (TIBE). Replacing "trust" with time in TIBE, and ignoring the identity-based aspects, we recover a special case of MRQED of dimension 1, but handling only intervals of the form $[t, T-1]$. Another related idea is sketched in [4], where it is shown how to transform a hierarchical identity-based encryption scheme into an encryption system that can send messages into the future. Translated into the language of this paper, this yields a Plain TSE scheme that can only support intervals of the form $[t, T-1]$. Unfortunately, because of specific details of the construction used, this approach does not seem capable of being extended to support more general intervals.

ABE and PE: TSE can be seen as arising from a special case of ciphertext-policy Attribute-Based Encryption (ABE) [19,3], itself a special case of Predicate Encryption (PE) [21], for a class of policies

which express interval membership and attributes which express specific times, and with the Time Server playing the role of Attribute Authority. We note that most work on ABE and PE to date, with the exception of [22], is limited to the "selective-attribute" case. In the context of TSE, converting to a non-selective security model would incur a cost of roughly $O(\frac{1}{T^2})$ in the tightness of the security reduction. However, our constructions for TSE already achieve fully adaptive security in the standard model with a *tight* reduction to the security of the IBE scheme used in the specific instantiation.

Broadcast Encryption: Broadcast Encryption (BE) is a cryptographic primitive designed to address the issue of broadcasting a message to an arbitrary subset drawn from a universe of users. Although conceptually opposites (in TSE the *keys* are broadcast while the message is sent beforehand), it can be shown that a BE scheme can be used to construct a Plain TSE scheme: assume the users in the BE scheme can be labeled with elements from $[0, T-1]$, consider a DTI as the target subset of addressed users in the BE encryption algorithm, and broadcast the private key for user with label t at time t. The functionality of the algorithms and the security of the schemes are preserved in this transformation (see Appendix B). There are however some *caveats* to this approach. First of all, to meet our TSE security requirement, we need the BE scheme to be *fully* collusion resistant. This condition immediately rules out many of the existing schemes. Furthermore, deploying BE schemes, as they are described generically in [6] and [18], requires the specification of the target set (in our case, the DTI) as an input to the decryption algorithm, inherently preventing the resulting Plain TSE scheme from having the DTI confidentiality property. Finally, the advantages and shortcomings of BE over our approach to the realisation of Plain TSE, as developed in Section 4.1, can be cast in a framework of trade-offs between the sizes of public parameters, private keys and ciphertexts, together with computational costs and strength of security achieved. We give a flavour of the possible trade-offs below.

For the purpose of our discussion, we consider the construction of Plain TSE schemes supporting T time instants from BE schemes. We focus on schemes that are provably secure in the standard model. In particular, the BE scheme of Boneh *et al.* in [6], proved only statically secure, results in a Plain TSE scheme with constant size private keys, public parameters and ciphertexts of size $O(\sqrt{T})$. In [18] Gentry and Waters introduce the new notion of semi-static security and provide a BE scheme achieving this notion which can be used to construct a Plain TSE scheme with constant size private keys and ciphertexts but public parameters whose size is linear in T. They also give a fully adaptive secure BE scheme with constant size private keys, but with public parameters and ciphertexts of size $O(\sqrt{T})$. The resulting Plain TSE scheme inherits these sizes. On the other hand, our solution results in schemes which are fully secure, with constant size public parameters, and private keys and ciphertexts of size $O(\log T)$. This brief comparison illustrates the value of a dedicated approach when realising Plain TSE.

Temporal access control: Significant related work in the symmetric key setting exists in the area of cryptographically-enabled "temporal access control", see for example [14] and the references therein. In this line of work, a key is associated with each time "point", and a key assignment scheme is used to ensure that an authorized user is able to derive keys for all the points contained in a designated interval. Such schemes generally require the publication of rather large amounts of information in order to achieve the desired functionality, but do allow efficient derivation of keys associated with time points. In contrast, we use public-key techniques, achieving small public parameters and greater flexibility in operation, at the cost of increased computation.

2 Preliminaries

Throughout the paper we will consider time as a discrete set of time units, regarding these as integers between 0 and $T-1$, where T represents the number of time units supported by the system. We denote by $[t_0, t_1]$, where $t_0 \leq t_1$, the interval containing all time units from t_0 to t_1 inclusive. Adversaries $\mathcal{A}$ are probabilistic polynomial-time algorithms. Bits b are selected uniformly at random from the set $\{0, 1\}$. We denote by $\Pr_{\mathcal{A},S}[\texttt{Event}]$ the probability that `Event` occurs when an adversary $\mathcal{A}$ interacts with a scheme S in a specified security game. By $\neg\texttt{Event}$ we denote the complement of `Event`. In particular, we denote by `Succ` the event that $b' = b$ in the games played in the following sections. We will use the standard definitions of and security notions for public-key encryption, identity-based encryption and signature schemes – see Appendix A for details.

3 Definitions and Security Notions

3.1 Plain TSE

We start by providing the definition and the security models for the basic form of Time-Specific Encryption, namely Plain TSE.

Definition 1. *A* Plain TSE scheme *is defined by four algorithms and has associated message space* $\mathcal{MSP} = \{0,1\}^l$*, ciphertext space* $\mathcal{CSP}$ *and time space* $\mathcal{T} = [0, T-1]$*. The parties involved in the scheme are the Time Server (TS), the sender (S) and a user (U). The four algorithms are as follows:*
`Plain.Setup`*. Run by TS, this algorithm takes as input the security parameter* κ *and* T *and outputs the master public key TS-MPK and the master secret key TS-MSK.*
`Plain.TIK-Ext`*. Run by TS, this algorithm takes as input TS-MPK, TS-MSK,* $t \in \mathcal{T}$ *and outputs the Time Instant Key (TIK)* k_t*. This is broadcast by TS at time* t*.*
`Plain.Enc`*. Run by S, this algorithm takes as input TS-MPK, a message* $m \in \mathcal{MSP}$ *and a Decryption Time Interval (DTI)* $[t_0, t_1] \subseteq \mathcal{T}$ *and outputs a ciphertext* c*, broadcast by S to all users.*
`Plain.Dec`*. Run by U, this algorithm takes as input TS-MPK, a ciphertext* $c \in \mathcal{CSP}$ *and a key* k_t *and outputs either a message* m *or a failure symbol* $\perp$*.*

The correctness property *requires that if* $c =$ `Plain.Enc`$(TS\text{-}MPK, m, [t_0, t_1])$ *and* k_t *is output by* `Plain.TIK-Ext` *on input* $t \in [t_0, t_1]$*, then* `Plain.Dec`$(TS\text{-}MPK, c, k_t) = m$*, and also that if* $t \notin [t_0, t_1]$ *then the decryption algorithm returns* $\perp$*.*

We define a model for IND-CPA security of a Plain TSE scheme.

Definition 2. *Consider the following game.*
Setup. *The challenger* $\mathcal{C}$ *runs* `Plain.Setup`(κ, T) *to generate master public key TS-MPK and master secret key TS-MSK and gives TS-MPK to the adversary* $\mathcal{A}$*.*
Phase 1. $\mathcal{A}$ *can adaptively issue TIK extraction queries to an oracle for any time* $t \in \mathcal{T}$*. The oracle will respond to each query with* k_t*.*
Challenge. $\mathcal{A}$ *selects two messages* m_0 *and* $m_1 \in \mathcal{MSP}$ *and a time interval* $[t_0, t_1] \subseteq \mathcal{T}$ *with the restriction that* $t \notin [t_0, t_1]$ *for all of the queries* t *in Phase 1.* $\mathcal{A}$ *passes* $m_0, m_1, [t_0, t_1]$ *to* $\mathcal{C}$*.* $\mathcal{C}$ *chooses a random bit* b *and computes* $c^* =$ `Plain.Enc`$(TS\text{-}MPK, m_b, [t_0, t_1])$*.* c^* *is passed to* $\mathcal{A}$*.*
Phase 2. $\mathcal{A}$ *continues to make queries to the TIK extraction oracle with the same restriction as in the Challenge phase.*
Guess. *The adversary outputs its guess* b' *for* b*.*

$\mathcal{A}$'s advantage in the above game is defined as $Adv_{\mathcal{A}}(\kappa) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 3. *We say that a Plain TSE scheme is* IND-CPA secure *if all polynomial-time adversaries have at most negligible advantage in the above game.*

We can extend this definition to address IND-CCA security by considering, in addition, a Decrypt oracle that acts as follows. On input the pair (c, t), where c is a ciphertext and $t \in \mathcal{T}$, it passes t to the TIK extraction oracle, which will respond with k_t. The Decrypt oracle will then compute `Plain.Dec`$(\text{TS-MPK}, c, k_t)$ and return either a message m or a failure symbol $\perp$ to the adversary. The Decrypt oracle can be adaptively issued queries (c, t) in both Phase 1 and Phase 2, but in the latter phase with the restriction that if c^* and $[t_0, t_1]$ are the challenge ciphertext and time interval, respectively, then the adversary cannot make a decryption query (c, t) where $c = c^*$ and $t \in [t_0, t_1]$. This restriction prevents the adversary from winning the game trivially.

3.2 Public-Key TSE

We now define another version of TSE called Public-Key TSE (PK-TSE) in which the sender S encrypts a message m to a *particular* receiver R who holds a key-pair (pk, sk). The message m has an associated decryption time interval $[t_0, t_1]$ specified by S. R can decrypt if he has his private key sk and a Time Instant Key (TIK) issued by TS between time t_0 and time t_1. We provide a more formal definition of PK-TSE.

Definition 4. *A* PK-TSE scheme *is defined by five algorithms and has associated message space* $\mathcal{MSP} = \{0,1\}^l$*, ciphertext space* $\mathcal{CSP}$ *and time space* $\mathcal{T} = [0, T-1]$*. The parties involved in the scheme are the Time Server (*TS*), the sender (*S*) and the receiver (*R*). The five algorithms are as follows:*

`PK.Setup`. Run by TS*, this algorithm takes as input the security parameter* κ *and* T *and outputs the master public key TS-MPK and the master secret key TS-MSK.*

`PK.TIK-Ext`. Run by TS*, this algorithm takes as input TS-MPK, TS-MSK,* $t \in \mathcal{T}$ *and outputs* k_t*. This is broadcast by* TS *at time* t*.*

`PK.KeyGen`. Run by R*, this algorithm takes as input the security parameter* κ *and outputs a key-pair* (pk, sk)*.*

`PK.Enc`. Run by S*, this algorithm takes as input TS-MPK, a message* $m \in \mathcal{MSP}$*, a time interval* $[t_0, t_1] \subseteq \mathcal{T}$ *and a public key* pk *and outputs a ciphertext* $c \in \mathcal{CSP}$*.*

`PK.Dec`. Run by R*, this algorithm takes as input TS-MPK, a ciphertext* $c \in \mathcal{CSP}$*, a key* k_t *and a private key* sk *and outputs either a message* m *or a failure symbol* $\perp$*.*

The correctness property *requires that if* $c =$ *`PK.Enc`*$(TS\text{-}MPK, m, [t_0, t_1], pk)$ *where* (pk, sk) *is output by `PK.KeyGen` and* k_t *is output by `PK.TIK-Ext` on input* $t \in [t_0, t_1]$*, then* *`PK.Dec`*$(TS\text{-}MPK, c, k_t, sk) = m$*, and also that if* $t \notin [t_0, t_1]$ *then the decryption algorithm returns* $\perp$*.*

To model the security of a PK-TSE scheme, we consider (as in [15]) the following kinds of adversaries:

- A curious TS who holds TS-MSK and wishes to break the confidentiality of the message.
- An intended but curious receiver who wishes to decrypt the message outside of the appropriate decryption time interval.

We observe that security against an outside adversary (who is not the intended recipient and does not know TS-MSK) trivially follows from security against either type of adversary considered above.

Remark 1. In Plain TSE there is only one type of adversary, i.e. the curious user, since there is no specific receiver and TS can trivially decrypt any message.

In defining the security models for PK-TSE we consider a single-user setting. We first define IND-CPA security against a curious TS.

Definition 5. *Consider the following game, which we call* $Game_{PK\text{-}TS}$*.*

Setup. *The challenger* $\mathcal{C}$ *runs* *`PK.Setup`*(κ, T) *to generate TS-MPK, TS-MSK, and runs* *`PK.KeyGen`*(κ) *to get a pair* (pk, sk)*.* $\mathcal{C}$ *gives* $(TS\text{-}MPK, TS\text{-}MSK, pk)$ *to the adversary* $\mathcal{A}$*.*

Challenge. $\mathcal{A}$ *selects two messages* m_0 *and* $m_1 \in \mathcal{MSP}$ *and a time interval* $[t_0, t_1] \subseteq \mathcal{T}$*.* $\mathcal{A}$ *passes* $m_0, m_1, [t_0, t_1]$ *to* $\mathcal{C}$*.* $\mathcal{C}$ *chooses a random bit* b *and computes* $c^* =$ *`PK.Enc`*$(TS\text{-}MPK, m_b, [t_0, t_1], pk)$*.* c^* *is passed to* $\mathcal{A}$*.*

Guess. *The adversary outputs its guess* b' *for* b*.*

We can extend this definition to address IND-CCA security by considering, in addition, a Decrypt oracle that on input the pair (c, t), where c is a ciphertext and $t \in \mathcal{T}$, returns either a message m or failure symbol $\perp$ to the adversary. The Decrypt oracle can be adaptively issued queries (c, t) before and after the challenge phase, but with the obvious restriction that the adversary cannot make queries of the form (c, t) where $c = c^*$ and $t \in [t_0, t_1]$ after the challenge phase.

We now address IND-CPA security against a curious receiver.

Definition 6. *Consider the following game, which we call* $Game_{PK\text{-}CR}$*.*

Setup. *The challenger* $\mathcal{C}$ *runs* *`PK.Setup`*(κ, T) *to generate TS-MPK, TS-MSK, and runs* *`PK.KeyGen`*(κ) *to get a pair* (pk, sk)*.* $\mathcal{C}$ *gives* $(TS\text{-}MPK, pk, sk)$ *to the adversary* $\mathcal{A}$*.*

Phase 1. $\mathcal{A}$ *can adaptively issue TIK extraction queries for any time* $t \in \mathcal{T}$*. The challenger will respond to each query with* $k_t =$ *`KP.TIK-Ext`*$(TS\text{-}MPK, TS\text{-}MSK, t)$*.*

Challenge. $\mathcal{A}$ *selects two messages* m_0 *and* $m_1 \in \mathcal{MSP}$ *and a time interval* $[t_0, t_1] \subseteq \mathcal{T}$ *with the restriction that* $t \notin [t_0, t_1]$ *for all of the queries* t *in Phase 1.* $\mathcal{A}$ *passes* $m_0, m_1, [t_0, t_1]$ *to* $\mathcal{C}$*.* $\mathcal{C}$ *chooses a random bit* b *and computes* $c^* =$ *`PK.Enc`*$(TS\text{-}MPK, m_b, [t_0, t_1], pk)$*.* c^* *is passed to* $\mathcal{A}$*.*

Phase 2. $\mathcal{A}$ *continues to make TIK extraction queries with the same restriction as in the Challenge phase.*

Guess. *The adversary outputs its guess* b' *for* b*.*

$\mathcal{A}$'s advantage in the above games is defined as $Adv_{\mathcal{A}}(\kappa) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

We observe that this chosen-plaintext notion of security is sufficient to capture all realistic attacks that can be mounted by a curious receiver, so that a chosen-ciphertext notion of security is not required for curious receivers. Indeed such a receiver would be the only entity in possession of its private key, and so would be the only entity that could implement a general decryption oracle in practice. We show next that such an oracle would either allow the receiver to win trivially or would not provide any advantage over just having access to the TIK extraction oracle in the chosen-plaintext setting. More precisely, suppose c^* is the challenge ciphertext and $[t_0, t_1]$ the challenge interval. To handle decryption queries of the form (c, t) where $c \neq c^*$ and $t \in [t_0, t_1]$, the receiver would need to use its private key in combination with a key k_t obtained from TS. But then having such a key k_t would also allow the curious receiver to decrypt c^* and so win the security game trivially. For any other decryption query (c, t), where now $t \notin [t_0, t_1]$, the curious receiver would be able to obtain the key k_t by making a TIK extraction query, and then use k_t together with its private key sk to decrypt c. So, in this situation, the curious receiver would gain no advantage from having access to a decryption oracle.

Definition 7. *We say that a PK-TSE scheme is* IND-CPA$_{TS}$ secure *if all polynomial-time adversaries have at most negligible advantage in* $Game_{PK\text{-}TS}$.

Definition 8. *We say that a PK-TSE scheme is* IND-CPA$_{CR}$ secure *if all polynomial-time adversaries have at most negligible advantage in* $Game_{PK\text{-}CR}$.

Definition 9. *We say that a PK-TSE scheme is* IND-CPA secure *if it is both is IND-CPA*$_{TS}$ *and IND-CPA*$_{CR}$ *secure.*

3.3 Identity-Based TSE

We finally consider an ID-based version of TSE, called ID-TSE, in which the sender encrypts a message m under the *identity* of a particular receiver. The message m has an associated decryption time interval $[t_0, t_1]$ specified by the sender. The receiver can decrypt if he holds the private key associated with his identity (as issued by a (semi-)trusted authority TA) and a Time Instant Key (TIK) issued by TS between time t_0 and time t_1. We now provide a more formal definition of ID-TSE.

Definition 10. *An* ID-TSE scheme *is defined by six algorithms and has associated message space* $\mathcal{MSP} = \{0,1\}^l$, *ciphertext space* $\mathcal{CSP}$, *identity space* $\mathcal{IDSP}$ *and time space* $\mathcal{T} = [0, T-1]$. *The parties involved in the scheme are the Time Server (TS), a trusted authority (TA), the sender (S) and the receiver (R). The six algorithms are as follows:*

`TS-Setup`. *Run by TS, this algorithm takes as input the security parameter* κ *and* T *and outputs the master public key TS-MPK and the master secret key TS-MSK.*

`ID-Setup`. *Run by TA, this algorithm takes as input the security parameter* κ *and outputs the master public key ID-MPK and the master secret key ID-MSK.*

`ID.TIK-Ext`. *Run by TS, this algorithm takes as input TS-MPK, TS-MSK,* $t \in \mathcal{T}$ *and outputs* k_t. *This is broadcast by TS at time t.*

`ID.Key-Ext`. *Run by TA, this algorithm takes as input ID-MPK, ID-MSK, an* $id \in \mathcal{IDSP}$ *and outputs the private key* sk_{id} *corresponding to id.*

`ID.Enc`. *Run by S, this algorithm takes as input TS-MPK, ID-MPK, a message* $m \in \mathcal{MSP}$, *a time interval* $[t_0, t_1] \subseteq \mathcal{T}$ *and an identity* $id \in \mathcal{IDSP}$ *and outputs a ciphertext* $c \in \mathcal{CSP}$.

`ID.Dec`. *Run by R, this algorithm takes as input TS-MPK, ID-MPK, a ciphertext* $c \in \mathcal{CSP}$, *a key* k_t *and a private key* sk_{id} *and outputs either a message* m *or a failure symbol* $\perp$.

The correctness property *requires that if* $c =$ `ID.Enc`$(TS\text{-}MPK, ID\text{-}MPK, m, [t_0, t_1], id)$, *where* sk_{id} *is output by* `ID.Key-Ext` *on input id, and* k_t *is output by* `PK.TIK-Ext` *on input* $t \in [t_0, t_1]$, *then* `ID.Dec`$(TS\text{-}MPK, ID\text{-}MPK, c, k_t, sk_{id}) = m$, *and also that if* $t \notin [t_0, t_1]$ *then the decryption algorithm returns* $\perp$.

We now model the security of an ID-TSE scheme. Since we are in an ID-based setting we will consider adversaries that interact with multiple users. We consider the following two types of adversaries:

– A curious TS who holds TS-MSK, and hence can derive TIKs for any time t, and wishes to break the confidentiality of the message.
– A curious TA who holds ID-MSK, and hence can derive private keys for any identity id, and wishes to break the confidentiality of the message.

We note that the latter adversary is more powerful than the natural analogue of the curious receiver in the ID setting.

We first define IND-CPA security against a curious TS.

Definition 11. *Consider the following game, which we call* $Game_{ID\text{-}TS}$.
Setup. *The challenger* $\mathcal{C}$ *runs* `TS-Setup`(κ, T) *to generate TS-MPK, TS-MSK, and runs* `ID-Setup`(κ) *to generate ID-MPK and ID-MSK.* $\mathcal{C}$ *gives* (*TS-MPK, TS-MSK, ID-MPK*) *to the adversary* $\mathcal{A}$.
Phase 1. $\mathcal{A}$ *can adaptively issue queries to a private key extraction oracle to get the private keys corresponding to identities id of its choice.* $\mathcal{C}$ *will respond with* sk_{id}=`ID.Key-Ext`(*ID-MPK, ID-MSK, id*).
Challenge. $\mathcal{A}$ *selects two messages* m_0 *and* $m_1 \in \mathcal{MSP}$ *and a time interval* $[t_0, t_1] \subseteq \mathcal{T}$ *and* $id^* \in \mathcal{IDSP}$ *with the restriction that* id^* *was not queried to the private key extraction oracle in Phase 1.* $\mathcal{A}$ *passes* $m_0, m_1, [t_0, t_1], id^*$ *to* $\mathcal{C}$. $\mathcal{C}$ *computes* $c^* =$ `ID.Enc`$(TS\text{-}MPK, ID\text{-}MPK, m_b, [t_0, t_1], id^*)$ *where b is a random bit.* c^* *is passed to* $\mathcal{A}$.
Phase 2. $\mathcal{A}$ *continues to make private key extraction queries with the same restriction as in the Challenge phase.*
Guess. *The adversary outputs its guess* b' *for b.*

We can easily modify this game to obtain a *selective-id* security notion, by requiring that at the beginning of the game, before the Setup phase, the adversary will output the identity id^* on which it wishes to be challenged.

We now define IND-CPA security against a curious TA.

Definition 12. *Consider the following game, which we call* $Game_{ID\text{-}TA}$.
Setup. *The challenger* $\mathcal{C}$ *runs* `TS-Setup`(κ, T) *to generate TS-MPK, TS-MSK, and runs* `ID-Setup`(κ) *to generate ID-MPK and ID-MSK.* $\mathcal{C}$ *gives* (*TS-MPK, ID-MPK, ID-MSK*) *to the adversary* $\mathcal{A}$.
Phase 1. $\mathcal{A}$ *can adaptively issue TIK extraction queries for any time* $t \in \mathcal{T}$. *The challenger will respond to each query with* $k_t =$`ID.TIK-Ext`$(TS\text{-}MPK, TS\text{-}MSK, t)$.
Challenge. $\mathcal{A}$ *selects two messages* m_0 *and* $m_1 \in \mathcal{MSP}$, *a time interval* $[t_0, t_1] \subseteq \mathcal{T}$, *with the restriction that* $t \notin [t_0, t_1]$ *for all of the the queries t made in Phase 1, and an identity* $id^* \in \mathcal{IDSP}$. $\mathcal{A}$ *passes* $m_0, m_1, [t_0, t_1], id^*$ *to* $\mathcal{C}$. $\mathcal{C}$ *chooses a random bit b.* $c^* =$ `ID.Enc`$(TS\text{-}MPK, ID\text{-}MPK, m_b, [t_0, t_1], id^*)$ *is computed and passed to* $\mathcal{A}$.
Phase 2. $\mathcal{A}$ *continues to make TIK extraction queries with the same restriction as in the Challenge phase.*
Guess. *The adversary outputs its guess* b' *for b.*

$\mathcal{A}$'s advantage in the above games is defined as $Adv_{\mathcal{A}}(\kappa) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 13. *We say that an ID-TSE scheme is* IND-CPA$_{TS}$ secure *if all polynomial-time adversaries have at most negligible advantage in* $Game_{ID\text{-}TS}$.

Definition 14. *We say that an ID-TSE scheme is* IND-CPA$_{TA}$ secure *if all polynomial-time adversaries have at most negligible advantage in* $Game_{ID\text{-}TA}$.

Definition 15. *We say that an ID-TSE scheme is* IND-CPA secure *if it is both IND-CPA*$_{TS}$ *and IND-CPA*$_{TA}$ *secure.*

IND-CCA security for ID-TSE can be defined by giving the adversary suitably restricted access to a Decrypt oracle. However, we do not formalise this notion here, since we are mainly interested in using ID-TSE as a building block to obtain PK-TSE schemes.

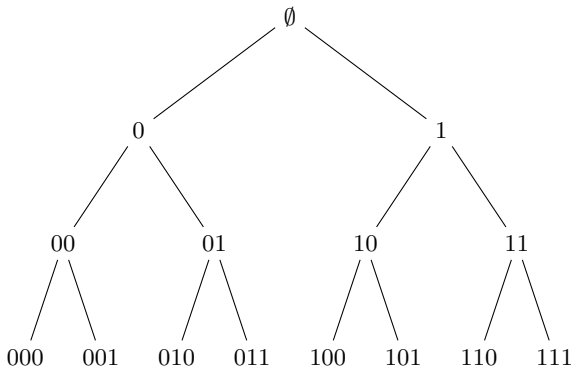

Fig. 1. Example of binary tree of depth $d = 3$ used in our construction.

4 Constructions for TSE Schemes

4.1 Plain TSE

Our first step towards building TSE schemes is to focus on how to achieve Plain TSE. We have to find a way to express time so that the Time Server can extract Time Instant Keys (TIKs) and the sender can specify a decryption time interval. Our approach will make use of a binary tree of depth d, where we denote with $\mathsf{parent}(x)$ and $\mathsf{child}(x)$ the standard notions of parent and child of a node x in a tree. The input T to `Plain.Setup`, which represents the number of allowed time units, will be of the form $T = 2^d$. The root node of the tree is labelled with $\emptyset$ and the non-root nodes are labelled with binary strings of lengths between 1 and d, as illustrated for the case $d = 3$ in Figure 1. Hence each node is associated with a binary string $t_0t_1...t_{l-1}$ of length $l \leq d$. In particular we will have that the leaves of the tree are binary strings of length d labelled from 0...0 (on the left) to 1...1 (on the right). Each leaf will represent a time instant $t = \Sigma_{i=0}^{d-1} t_i 2^{d-1-i}$ between 0 and $T - 1$.

We now define two particular sets of nodes.

- **Path $\mathcal{P}_t$ to t.** Given a time instant $t = \Sigma_{i=0}^{d-1} t_i 2^{d-1-i}$ we construct the following path $\mathcal{P}_t$ in the tree, where the last node is the leaf corresponding to t:

$$\emptyset, t_0, t_0t_1, ..., t_0...t_{d-1}.$$

- **Set $\mathcal{S}_{[t_0,t_1]}$ covering the interval $[t_0, t_1]$.** $\mathcal{S}_{[t_0,t_1]}$ is the minimal set of roots of subtrees that cover leaves representing time instants in $[t_0, t_1]$. We will call this the *cover set* for $[t_0, t_1]$. Such a set is unique and of size at most $2d$. We can compute the labels of the nodes in $\mathcal{S}_{[t_0,t_1]}$ in a particular order by running Algorithm 1 on input $[t_0, t_1]$.

It is easy to see that $\mathcal{P}_t$ and $\mathcal{S}_{[t_0,t_1]}$ intersect in a unique node if and only if $t \in [t_0, t_1]$. This property will ensure that the correctness requirement holds for the Plain TSE scheme that we will construct. The key idea, then, is to view the nodes of the tree as identities and make use of identity-based encryption techniques to instantiate a Plain TSE scheme. Informally, the sender will encrypt under the nodes in the cover set for the Decryption Time Interval (DTI), and the TIK for time t will be the set of private keys associated to the nodes on the path $\mathcal{P}_t$ to t. More formally, we use an Identity-based Encryption (IBE) scheme $I = (\mathtt{Setup}, \mathtt{Key\text{-}Ext}, \mathtt{Enc}, \mathtt{Dec})$ with message space $\mathcal{MSP} = \{0,1\}^l$ to construct $X = (\mathtt{Plain.Setup}, \mathtt{Plain.TIK\text{-}Ext}, \mathtt{Plain.Enc}, \mathtt{Plain.Dec})$, a Plain TSE scheme with the same message space, in the following way:

– $\mathtt{Plain.Setup}(\kappa, T)$. Run `Setup` on input κ to obtain TS-MPK and the master secret key TS-MSK. We set $T = 2^d$, where d is the depth of the tree used in our construction.

– $\mathtt{Plain.TIK\text{-}Ext}$(TS-MPK, TS-MSK, t). Construct $\mathcal{P}_t$ to obtain the list of nodes $\{\emptyset, p_1, ..., p_d\}$ on the path to t. Run `Key-Ext` algorithm on all nodes p in $\mathcal{P}_t$ to obtain a set of private keys $\mathcal{D}_t = \{d_p : p \in \mathcal{P}_t\}$. Return $\mathcal{D}_t$ (we implicitly assume that t can be recovered from this set because $\mathcal{D}_t$ will be broadcast at the particular time t).

Algorithm 1 Compute $\mathcal{S}_{[t_0,t_1]}$.

```
Let L be the binary expansion of t_0.
Let R be the binary expansion of t_1.
Let S = ∅.
while L < R do
   if L ≡ 0 mod 2 then
      L = parent(L)
   else
      S = S ∪ {L}
      L = parent(L) + 1
   end if
   if R ≡ 0 mod 2 then
      S = S ∪ {R}
      R = parent(R) − 1
   else
      R = parent(R)
   end if
end while
if L = R then
   S = S ∪ {L}
end if
return S
```

– `Plain.Enc`(TS-MPK, $m, [t_0, t_1]$). Run Algorithm 1 on input $[t_0, t_1]$ to compute a list of nodes $\mathcal{S}_{[t_0,t_1]}$. For each $s \in \mathcal{S}_{[t_0,t_1]}$ run `Enc`(TS-MPK, m, s) to obtain a list of ciphertexts $\mathcal{CT}_{[t_0,t_1]} = \{c_p : p \in \mathcal{S}_{[t_0,t_1]}\}$. Output $C = (\mathcal{CT}_{[t_0,t_1]}, [t_0, t_1])$.
– `Plain.Dec`(TS-MPK, $C, \mathcal{D}_t$). Here $C = (\mathcal{CT}, [t_0, t_1])$ denotes a list of ciphertexts for the scheme I together with a time interval. If $t \notin [t_0, t_1]$ return $\perp$. Otherwise run Algorithm 1 to generate an ordered list of nodes $\mathcal{S}_{[t_0,t_1]}$ and generate the set $\mathcal{P}_t$; the intersection of these sets is a unique node p. Obtain the key d_p corresponding to p from $\mathcal{D}_t$. Run `Dec`(TS-MPK, c_p, d_p), where $c_p \in \mathcal{CT}$ is in the same position in the list $\mathcal{CT}$ as p is in $\mathcal{S}_{[t_0,t_1]}$, to obtain either a message m or a failure symbol $\perp$. Output the result.

For the above construction, the following result holds.

Theorem 1. *Let I be an IND-CPA secure IBE scheme. Then the Plain TSE scheme X constructed from I as above is IND-CPA secure in the sense of Definition 3, and is correct.*

Proof sketch: The proof of IND-CPA security works in two steps. In the first step, we show that for any IND-CPA attacker $\mathcal{A}$ with non-negligible advantage against the Plain TSE scheme, there is a modified IND-CPA adversary $\mathcal{B}$ having non-negligible advantage against the IBE scheme. This modified adversary specifies in its challenge phase a list of d identities, none of which are allowed to be queried to the ID-based private key extraction oracle at any point in the game, along with a pair of messages m_0, m_1. In return, $\mathcal{B}$ receives from its challenger the encryption of m_b under each of the identities in its list. As usual $\mathcal{B}$'s task is to find b. This reduction is straightforward, relying on the fact that the restriction $t \notin [t_0, t_1]$ imposed on the TIKs k_t that the Plain TSE adversary $\mathcal{A}$ can obtain from its TIK extraction oracle means that $\mathcal{B}$ can handle $\mathcal{A}$'s queries to its TIK extraction oracle by passing them to its ID-based private key extraction oracle. In the second step, we use a standard hybrid argument to reduce the IND-CPA security of the IBE scheme against modified adversaries to its IND-CPA security against normal adversaries (who specify just one identity in the challenge phase). The proof of correctness is straightforward, relying on the fact that $\mathcal{P}_t$ does not intersect $\mathcal{S}_{[t_0,t_1]}$ if $t \notin [t_0, t_1]$. □

We provide a small example to illustrate our construction. Consider $d = 3$ as in Figure 1.

Example: Suppose we wish to decrypt a message that was encrypted using time interval $[2, 6]$. In the tree, these endpoints will correspond to nodes with labels 010 and 110, respectively. We compute $\mathcal{S}_{[2,6]} = \{01, 10, 110\}$. Suppose we obtain the TIK broadcast by TS at time 4 (corresponding to the leaf node labelled 100). This means that we obtain a list of private keys for nodes on the path $\mathcal{P}_4$ from the root to 100. In particular, we have the key corresponding to node 10, the unique intersection of $\mathcal{P}_4$ and $\mathcal{S}_{[2,6]}$. Hence, we are able to decrypt. We observe that for any time t outside of the interval $[2, 6]$, there is no intersection between $\mathcal{P}_t$ and $\mathcal{S}_{[2,6]}$.

In general, ciphertexts in the Plain TSE scheme X consist of up to $2d$ ciphertexts from the IBE scheme I, while private keys consist of at most d private keys from I. The public parameters of X are the same size as those of I. The cost of encryption for the scheme X is up to $2d$ times greater than its cost for the scheme I, while decryption for X costs the same as for I. This compares well with the naive solution of encrypting with a single private key to every time instant in the interval, as it allows for shorter ciphertexts.

A variety of IBE schemes can be used to instantiate the above construction, including Waters' [28] and Gentry's [17] schemes in the standard model, and the Boneh-Franklin scheme [5] and the Sakai-Kasahara scheme (as analysed in [11]) in the ROM. Each of them has various advantages and disadvantages in terms of efficiency and the sizes of public parameters and ciphertexts. For example, Waters' scheme has relatively large public parameters, compact ciphertexts, and depends for its security on the Bilinear Diffie-Hellman Problem, while Gentry's scheme has small public parameters and ciphertexts, but its security depends on a non-standard hardness assumption, the q-Truncated Decisional Augmented Bilinear Diffie-Hellman Exponent (q-TDABDHE) problem.

A potentially more efficient approach would be to use a multi-recipient, single key, ID-based KEM (MR-SK-IBKEM), as defined in [1], which would allow encapsulation of the same key for multiple recipients $id_1, ..., id_n$ in an efficient and secure manner. Using an approach similar to that in [27], we can combine an MR-SK-IBKEM with a (symmetric) Data Encapsulation Mechanism (DEM) to produce an multi-recipient IBE scheme in a standard way [2]; if the underlying KEM and DEM satisfy appropriate security notions (IND-CPA and FG-CPA security, respectively), then the resulting multi-recipient IBE scheme will be IND-CPA secure [2]. This primitive perfectly matches our requirement to be able to encrypt the same message to all nodes in a cover set simultaneously, and it is easy to see how to obtain IND-CPA secure Plain TSE from such a primitive. However, current instantiations for IND-CPA secure MR-SK-IBKEMs are only known in the ROM (see for example [1]). To the best of our knowledge, it remains an open problem to find efficient instantiations that are secure in the standard model. We recall that the scheme in [27] actually solves the *dual* of our problem and can only handle intervals of the type $[t, T-1]$.

4.2 IND-CPA Security

We now wish to address the problem of instantiating IND-CPA secure PK-TSE and ID-TSE schemes. We begin with the PK-TSE case.

Let Π = (`Gen`,`E`,`D`) be a public-key encryption (PKE) scheme with message space $\{0,1\}^l$. We will construct a PK-TSE scheme from a Plain TSE scheme X = (`Plain.Setup`,`Plain.TIK-Ext`,`Plain.Enc`,`Plain.Dec`) with $\mathcal{MSP} = \{0,1\}^l$ and Π in the following way:

– `PK.Setup`(κ, T). Run `Plain.Setup` on the same input to obtain TS-MPK and the master secret key TS-MSK.

– `PK.TIK-Ext`(TS-MPK, TS-MSK, t). Run `Plain.TIK-Ext`(TS-MPK,TS-MSK,t) to obtain k_t, broadcast by TS at time t.

– `PK.KeyGen`(κ). Run `Gen`(κ), which will output a key-pair (pk, sk).

– `PK.Enc`(TS-MPK, $m, [t_0, t_1], pk$). Pick a random $r \in \{0,1\}^l$ and set $m' = m \oplus r$. Then run the algorithm `Plain.Enc`(TS-MPK, $r, [t_0, t_1]$) to obtain c_0 and `E`(pk, m') to get c_1. The ciphertext will be (c_0, c_1).

– `PK.Dec`(TS-MPK, $(c_0, c_1), k_t, sk$). Parse c_0 and c_1. Run `Plain.Dec`(TS-MPK, c_0, k_t) which will output either a message r or a failure symbol $\perp$. Run `D`(sk, c_1) which will output either a message m' or a failure symbol $\perp$. If either of the decryption algorithms returns $\perp$, then output $\perp$; otherwise output $m = r \oplus m'$.

Lemma 1. *Let Π be an IND-CPA secure PKE scheme. Then the PK-TSE scheme, constructed as above, is IND-CPA$_{TS}$ secure.*

Proof. Suppose we have an adversary $\mathcal{A}$ against the PK-TSE scheme with advantage $\epsilon(\kappa)$. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the PKE scheme. The game proceeds as follows.

Setup. The challenger $\mathcal{C}$ runs $\texttt{Gen}(\kappa)$ to obtain a pair (pk, sk). It gives pk to $\mathcal{B}$. $\mathcal{B}$ runs $\texttt{PK.Setup}(\kappa,\text{T})$ to generate (TS-MPK, TS-MSK) and gives (TS-MPK, TS-MSK, pk) to the adversary $\mathcal{A}$.
Challenge. $\mathcal{A}$ outputs two messages m_0 and $m_1 \in \mathcal{MSP}$ and a time interval $[t_0, t_1] \subseteq \mathcal{T}$. $\mathcal{A}$ passes $m_0, m_1, [t_0, t_1]$ to $\mathcal{B}$. $\mathcal{B}$ picks a random $r \in \{0,1\}^l$ and passes $m_0 \oplus r, m_1 \oplus r$ to $\mathcal{C}$. $\mathcal{C}$ picks a random bit b and computes $c'' = \texttt{E}(pk, m_b \oplus r)$. $\mathcal{B}$ runs $\texttt{Plain.Enc}(\text{TS-MPK}, r, [t_0, t_1]) = c'$. Finally, $\mathcal{B}$ passes $c^* = (c', c'')$ to $\mathcal{A}$.
Guess. The adversary outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the IND-CPA$_{TS}$ game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real game and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

We can prove the following result in an analogous way.

Lemma 2. *Let X be an IND-CPA secure Plain TSE scheme. Then the PK-TSE scheme, constructed as above, is IND-CPA$_{CR}$ secure. Moreover, if X is correct, then so is the resulting PK-TSE scheme.*

Hence, the following theorem holds.

Theorem 2. *Let X be an IND-CPA secure Plain TSE scheme and Π be an IND-CPA secure PKE scheme. Then the PK-TSE scheme, constructed as above, is IND-CPA secure. Moreover, if X is correct, then so is the resulting PK-TSE scheme.*

To achieve IND-CPA security in the ID-TSE setting we can adopt an approach similar to the one used above to build a PK-TSE scheme, where instead of a PKE scheme we employ an IBE scheme $I = (\texttt{Setup}, \texttt{Key-Ext}, \texttt{Enc}, \texttt{Dec})$ in the obvious manner. In this setting we obtain an analogous result:

Theorem 3. *Let X be an IND-CPA secure Plain TSE scheme and I be an IND-CPA secure IBE scheme. Then the ID-TSE scheme, constructed analogously to the construction of the PK-TSE scheme above, is IND-CPA secure. Moreover, if X is correct, then so is the resulting ID-TSE scheme.*

In particular, we observe that if I is a *selective-id* IND-CPA secure IBE scheme, then it can be shown that the resulting ID-TSE scheme is also secure in the selective sense.

4.3 IND-CCA Security

We will now address the problem of building IND-CCA$_{TS}$ secure PK-TSE schemes, using an approach similar to that of [8].

Suppose we have a selective-id IND-CPA$_{TS}$ secure ID-TSE scheme $I = (\texttt{TS-Setup}, \texttt{ID-Setup}, \texttt{ID.TIK-Ext}, \texttt{ID.Key-Ext}, \texttt{ID.Enc}, \texttt{ID.Dec})$, with $\mathcal{MSP} = \{0,1\}^l$ and $\mathcal{IDSP} = \{0,1\}^n$. We construct an IND-CCA$_{TS}$ secure PK-TSE scheme $\Gamma = (\texttt{PK.Setup}, \texttt{PK.TIK-Ext}, \texttt{PK.KeyGen}, \texttt{PK.Enc}, \texttt{PK.Dec})$. In the construction, we use a signature scheme $\Sigma = (\texttt{G}, \texttt{Sign}, \texttt{Ver})$, whose generation algorithm $\texttt{G}$ outputs verification keys of length n. We construct the algorithms of Γ as follows:

– $\texttt{PK.Setup}(\kappa, T)$. Run $\texttt{TS-Setup}(\kappa, T)$ to get TS-MPK, TS-MSK.
– $\texttt{PK.TIK-Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$. Run $\texttt{ID.TIK-Ext}(\text{TS-MPK}, \text{TS-MSK}, t)$ to obtain TIK k_t.
– $\texttt{PK.KeyGen}(\kappa)$. Run $\texttt{ID-Setup}(\kappa)$ to get (ID-MPK, ID-MSK), a key-pair.
– $\texttt{PK.Enc}(\text{TS-MPK}, m, [t_0, t_1], \text{ID-MPK})$. Run $\texttt{G}(\kappa)$ and obtain $(vk, sigk)$. Compute $c =$ $\texttt{ID.Enc}(\text{TS-MPK}, \text{ID-MPK}, m, [t_0, t_1], vk)$. Produce $\sigma = \texttt{Sign}(sigk, c)$. The final ciphertext will be $C = (vk, c, \sigma)$.
– $\texttt{PK.Dec}(\text{TS-MPK}, C, k_t, \text{ID-MSK})$. Parse C as (vk, c, σ). Check if $\texttt{Ver}(vk, c, \sigma) = 1$. If not, output $\perp$. Otherwise, run $\texttt{ID.Key-Ext}(\text{ID-MPK}, \text{ID-MSK}, vk)$ to obtain sk_{vk} and decrypt c by running $\texttt{ID.Dec}$ with inputs k_t, sk_{vk}.

Theorem 4. *Let I be a correct, selective-id IND-CPA$_{TS}$ secure ID-TSE scheme and Σ a strongly unforgeable one-time signature scheme. Then Γ, as constructed above, is an IND-CCA$_{TS}$ secure PK-TSE scheme.*

We defer the proof of this theorem to Appendix C. It is also easy to see that the following result holds.

Theorem 5. *Let I be a selective-id IND-CPA$_{TA}$ secure ID-TSE scheme and Σ a strongly unforgeable one-time signature scheme. Then Γ, as constructed above, is an IND-CPA$_{CR}$ secure PK-TSE scheme.*

We can also use a variant of the more complex Boneh-Katz transform [7] to construct PK-TSE schemes that are IND-CCA secure in the standard model. The resulting schemes generally have improved efficiency. Details are omitted.

We discuss the problem of obtaining IND-CCA secure ID-TSE schemes in the following section, where we discuss future work.

5 Extensions and Future Work

Various extensions of our TSE concept are possible. For example, we could consider TSE schemes (in all three settings) that have the property of hiding the decryption time interval of ciphertexts from adversaries. Our current constructions do not offer this. We call such a property DTIC (Decryption Time Interval Confidentiality). We can model the DTIC-IND-CPA/CCA security of a TSE scheme by allowing the adversary to select two messages and *two* time intervals in the challenge phase and requiring him to guess which message was encrypted under which interval. We distinguish between the plain setting, where the decryption time interval is hidden from all users, making successful decryption a kind of *proof of work* (since every user will need to attempt decryption under possibly many keys), and the public-key and identity-based settings, where the decryption time interval is hidden from everyone except the intended recipient. It will be interesting to explore these properties in conjunction with extensions of key-privacy/recipient-anonymity in the public-key and identity-based settings, to obtain enhanced security properties for TSE.

Another relevant problem is that of constructing TSE schemes allowing the capability of *opening* the message outside of the decryption time interval, a useful feature supporting *break the glass* policies. This extension has already been considered in the setting of TRE [20,15]. It would also be interesting to find schemes in which the maximum time value T can be dynamically extended without resetting the parameters of the Time Server, and schemes in which the granularity of times can be adjusted by users (such a consideration was made in [27] for TIBE).

Our focus in this paper has been on achieving IND-CCA security of PK-TSE in the standard model. This leaves the problem of constructing IND-CCA secure ID-TSE schemes, in either the standard model or the ROM. The former, we believe, should be achievable by introducing hierarchical ID-TSE notions and further extending the CHK-style transform to this setting. The latter can be achieved by developing Fujisaki-Okamoto style transforms for the TSE setting. In this paper, we have constructed PK-TSE/ID-TSE schemes in the IND-CPA setting by combining Plain TSE schemes and PKE/IBE schemes. We have then used the CHK technique to obtain CCA security for PK-TSE. It might be worth investigating an alternative approach in which one first obtains an IND-CCA secure Plain TSE scheme, and then applies a Dodis-Katz style construction [16] to combine this with IND-CCA secure PKE/IBE schemes. This may lead to efficiency gains as compared to our CHK-based constructions. It would also be useful to solve the open problem of constructing MR-SK-IBKEMs that are provably secure in the standard model, in order to improve the efficiency of our Plain TSE constructions.

Thinking more broadly, one can envisage the development of the wider concept of Time-Specific Cryptography. This could include, for example, time-specific signatures (where signatures can only be created within certain time intervals). We believe there is much interesting work still to be done in this area.

Acknowledgement

We are grateful to Jason Crampton for suggesting the application of TSE to authentication protocols.

References

1. Manuel Barbosa and Pooya Farshim. Efficient identity-based key encapsulation to multiple parties. In Smart [26], pages 428–441.

2. Kamel Bentahar, Pooya Farshim, John Malone-Lee, and Nigel P. Smart. Generic constructions of identity-based and certificateless KEMs. *J. Cryptology*, 21(2):178–199, 2008.
3. John Bethencourt, Amit Sahai, and Brent Waters. Ciphertext-policy attribute-based encryption. In Pfitzmann and McDaniel [24], pages 321–334.
4. Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical identity based encryption with constant size ciphertext. In Cramer [13], pages 440–456.
5. Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil Pairing. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
6. Dan Boneh, Craig Gentry, and Brent Waters. Collusion resistant broadcast encryption with short ciphertexts and private keys. In Victor Shoup, editor, *CRYPTO 2005*, volume 3621 of *Lecture Notes in Computer Science*, pages 258–275. Springer, 2005.
7. Dan Boneh and Jonathan Katz. Improved efficiency for CCA-secure cryptosystems built using identity-based encryption. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *Lecture Notes in Computer Science*, pages 87–103. Springer, 2005.
8. Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-ciphertext security from identity-based encryption. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 207–222. Springer, 2004.
9. 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, *ICICS 2005*, volume 3783 of *Lecture Notes in Computer Science*, pages 291–303. Springer, 2005.
10. Aldar C.-F. Chan and Ian F. Blake. Scalable, server-passive, user-anonymous timed release cryptography. In *ICDCS 2005*, pages 504–513. IEEE Computer Society, 2005.
11. Liqun Chen and Zhaohui Cheng. Security proof of Sakai-Kasahara's identity-based encryption scheme. In Smart [26], pages 442–459.
12. Sherman S. M. Chow, Volker Roth, and Eleanor G. Rieffel. General certificateless encryption and timed-release encryption. In Rafail Ostrovsky, Roberto De Prisco, and Ivan Visconti, editors, *SCN 2008*, volume 5229 of *Lecture Notes in Computer Science*, pages 126–143. Springer, 2008.
13. 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*. Springer, 2005.
14. Jason Crampton. Trade-offs in cryptographic implementations of temporal access control. In Audun Jøsang, Torleiv Maseng, and Svein J. Knapskog, editors, *NordSec 2009*, volume 5838 of *Lecture Notes in Computer Science*, pages 72–87. Springer, 2009.
15. Alexander W. Dent and Qiang Tang. Revisiting the security model for timed-release encryption with pre-open capability. In Juan A. Garay, Arjen K. Lenstra, Masahiro Mambo, and René Peralta, editors, *ISC 2007*, volume 4779 of *Lecture Notes in Computer Science*, pages 158–174. Springer, 2007.
16. Yevgeniy Dodis and Jonathan Katz. Chosen-ciphertext security of multiple encryption. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *Lecture Notes in Computer Science*, pages 188–209. Springer, 2005.
17. Craig Gentry. Practical identity-based encryption without random oracles. In Serge Vaudenay, editor, *EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 445–464. Springer, 2006.
18. Craig Gentry and Brent Waters. Adaptive security in broadcast encryption systems (with short ciphertexts). In Antoine Joux, editor, *EUROCRYPT 2009*, volume 5479 of *Lecture Notes in Computer Science*, pages 171–188. Springer, 2009.
19. Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM Conference on Computer and Communications Security 2006*, pages 89–98. ACM, 2006.
20. 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 2005*, volume 3650 of *Lecture Notes in Computer Science*, pages 344–358. Springer, 2005.
21. Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In Nigel P. Smart, editor, *EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 146–162. Springer, 2008.
22. Allison Lewko, Tatsuaki Okamoto, Amit Sahai, Katsuyuki Takashima, and Brent Waters. Fully secure functional encryption: Attribute-based encryption and (hierarchical) inner product encryption. Cryptology ePrint Archive, Report 2010/110, 2010. `http://eprint.iacr.org/`.
23. Timothy C. May. Time-release crypto, 1993. manuscript.
24. Birgit Pfitzmann and Patrick McDaniel, editors. *IEEE Symposium on Security and Privacy (S&P 2007), 20-23 May 2007, Oakland, California, USA*. IEEE Computer Society, 2007.
25. Elaine Shi, John Bethencourt, Hubert T.-H. Chan, Dawn Xiaodong Song, and Adrian Perrig. Multi-dimensional range query over encrypted data. In Pfitzmann and McDaniel [24], pages 350–364.
26. Nigel P. Smart, editor. *Cryptography and Coding, 10th IMA International Conference, Cirencester, UK, December 19-21, 2005, Proceedings*, volume 3796 of *Lecture Notes in Computer Science*. Springer, 2005.
27. Mudhakar Srivatsa, Shane Balfe, Kenneth G. Paterson, and Pankaj Rohatgi. Trust management for secure information flows. In Peng Ning, Paul F. Syverson, and Somesh Jha, editors, *ACM Conference on Computer and Communications Security 2008*, pages 175–188. ACM, 2008.
28. Brent Waters. Efficient identity-based encryption without random oracles. In Cramer [13], pages 114–127.

A Additional Definitions and Security Models

Definition 16. *A* Public Key Encryption (PKE) scheme *Π is defined by three algorithms and has associated message space $\mathcal{MSP}$ and ciphertext space $\mathcal{CSP}$. The three algorithms are as follows.*

`Gen`. This algorithm takes as input the security parameter κ and outputs a key-pair (pk, sk).

`E`. This algorithm takes as input a public key pk and a message $m \in \mathcal{MSP}$ and outputs a ciphertext c.

`D`. This algorithm takes as input a private key sk and a ciphertext $c \in \mathcal{CSP}$ and outputs either a message m or a failure symbol $\perp$.

The correctness property *requires that if $c = \texttt{E}(pk, m)$ and sk is the private key corresponding to pk then $\texttt{D}(sk, c) = m$.*

We define a model for IND-CPA security of a PKE scheme.

Definition 17. *Consider the following game.*
Setup. *The challenger $\mathcal{C}$ runs `Gen`(κ) to generate a key-pair (pk, sk) and gives pk to the adversary $\mathcal{A}$.*
Challenge. *$\mathcal{A}$ selects two messages m_0 and $m_1 \in \mathcal{MSP}$ and passes them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit b and computes $c^* = \texttt{E}(pk, m_b)$. c^* is passed to $\mathcal{A}$.*
Guess. *The adversary outputs its guess b' for b.*

$\mathcal{A}$'s advantage in the above game is defined as $Adv_{\mathcal{A}}(\kappa) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 18. *We say that a PKE scheme is* IND-CPA secure *if all polynomial-time adversaries have at most negligible advantage in the above game.*

Definition 19. *An* Identity-Based Encryption (IBE) scheme *I is defined by four algorithms and has associated message space $\mathcal{MSP}$, ciphertext space $\mathcal{CSP}$ and identity space $\mathcal{IDSP}$. The four algorithms are as follows.*

`Setup`. This algorithm takes as input the security parameter κ and outputs a master public key ID-MPK and a master secret key ID-MSK.

`Key-Ext`. This algorithm takes as input ID-MPK, ID-MSK and an $id \in \mathcal{IDSP}$ and outputs the private key sk_{id} corresponding to id.

`Enc`. This algorithm takes as input ID-MPK, a message $m \in \mathcal{MSP}$ and an identity $id \in \mathcal{IDSP}$ and outputs a ciphertext $c \in \mathcal{CSP}$.

`Dec`. This algorithm takes as input a private key sk_{id} and a ciphertext $c \in \mathcal{CSP}$ and outputs either a message m or a failure symbol $\perp$.

The correctness property *requires that if $c = \texttt{Enc}(\text{ID-MPK}, m, id)$ and sk_{id} is the private key corresponding to id then $\texttt{D}(sk_{id}, c) = m$.*

We define a model for IND-CPA security of an IBE scheme.

Definition 20. *Consider the following game.*
Setup. *The challenger $\mathcal{C}$ runs `Setup`(κ) to generate master public key ID-MPK and master secret key ID-MSK and gives ID-MPK to the adversary $\mathcal{A}$.*
Phase 1. *$\mathcal{A}$ can adaptively issue private key extraction queries to an oracle for any identity $id \in \mathcal{IDSP}$. The oracle will respond to each query with sk_{id}.*
Challenge. *$\mathcal{A}$ selects two messages m_0 and $m_1 \in \mathcal{MSP}$ and an $id^* \in \mathcal{IDSP}$ with the restriction that for none of the queries in Phase 1 we have that $id = id^*$. $\mathcal{A}$ passes m_0, m_1, id^* to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit b and computes $c^* = \texttt{Enc}(\text{ID-MPK}, m_b, id^*)$. c^* is passed to $\mathcal{A}$.*
Phase 2. *$\mathcal{A}$ continues to make queries to the private key extraction oracle with the same restriction we have in the Challenge phase.*
Guess. *The adversary outputs its guess b' for b.*

$\mathcal{A}$'s advantage in the above game is defined as $Adv_{\mathcal{A}}(\kappa) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 21. *We say that an IBE scheme is* IND-CPA secure *if all polynomial-time adversaries have at most negligible advantage in the above game.*

Definition 22. *A* signature scheme Σ *is defined by three algorithms and has associated message space* $\mathcal{MSP}$ *and signature space* $\mathcal{SSP}$*. The three algorithms are as follows.*

`G`*. This algorithm takes as input the security parameter* κ *and outputs a signing-verification key pair (sigk,vk).*

`Sign`*. This algorithm takes as input a signing key* $sigk$ *and a message* $m \in \mathcal{MSP}$ *and outputs a signature* $\sigma \in \mathcal{SSP}$*.*

`Ver`*. This algorithm takes as input a verification key* vk*, a message* $m \in \mathcal{MSP}$ *and a signature* $\sigma \in \mathcal{SSP}$ *and outputs a bit* $v \in \{0,1\}$*, where* $v = 1$ *means acceptance and* $v = 0$ *means rejection.*

The correctness property *requires that for every* $(sigk, vk)$ *output by* `G` *and for every message* $m \in \mathcal{MSP}$ *it holds that* $\texttt{Ver}(vk, m, \texttt{Sig}(sigk, m)) = 1$*.*

We define the notion of strong unforgeability under a one-time message attack (SUF-1CMA) for a signature scheme Σ as follows.

Definition 23. *Consider the following game.*
Setup. *The challenger* $\mathcal{C}$ *runs* `G`(κ) *to generate a signing-verification key pair (sigk,vk) and gives* vk *to the adversary* $\mathcal{A}$*.*
Signing Query. $\mathcal{A}$ *selects a message* $m \in \mathcal{MSP}$ *and gives it to* $\mathcal{C}$*.* $\mathcal{C}$ *computes* $\sigma = \texttt{Sign}(sigk, m)$*.* σ *is passed to* $\mathcal{A}$*.*
Forgery. $\mathcal{A}$ *outputs a pair* (m^*, σ^*)*.*

$\mathcal{A}$'s advantage in the above game is defined as $Adv_{\mathcal{A}}(\kappa) = \Pr[\texttt{Ver}(vk, m^*, \sigma^*) = 1 \wedge (m^*, \sigma^*) \neq (m, \sigma)]$.

Definition 24. *We say that a signature scheme is* SUF-1CMA secure *if all polynomial-time adversaries have at most negligible advantage in the above game.*

B Plain TSE from BE

We will show how a Plain TSE scheme can be built from a Broadcast Encryption (BE) scheme in a straightforward way. We begin by giving a formal definition of a public-key broadcast encryption system and a security notion for it, similar to the ones given in [18].

Definition 25. *A* broadcast encryption (BE) scheme *is defined by four algorithms and has associated message space* $\mathcal{MSP}$ *and ciphertext space* $\mathcal{CSP}$*. The four algorithms are as follows.*
`BE.Setup`*. This algorithm takes as input the security parameter* κ*, the number of users of the system* n *and the maximal size* $l \leq n$ *of a broadcast target group. It outputs a master public key BE-MPK and a master secret key BE-MSK.*
`BE.Key-Gen`*. This algorithm takes as input BE-MPK, BE-MSK and an index* $i \in \{1, ..., n\}$ *and outputs the private key* sk_i *for user* i*.*
`BE.Enc`*. This algorithm takes as input BE-MPK, a message* $m \in \mathcal{MSP}$ *and a subset* $S \subseteq \{1, ..., n\}$*, the broadcast target set. If* $|S| \leq l$*, it outputs a ciphertext* $c \in \mathcal{CSP}$*.*
`BE.Dec`*. This algorithm takes as input BE-MPK, a subset* $S \subseteq \{1, ..., n\}$*, an index* $i \in \{1, ..., n\}$*, a private key* sk_i *and a ciphertext* $c \in \mathcal{CSP}$*. It outputs either a message* m *or a failure symbol* $\perp$*.*

The correctness property *requires that if* $c = \texttt{BE.Enc}(BE\text{-}MPK, m, S)$ *and* sk_i *is the private key for* $i \in S$ *then* $\texttt{BE.Dec}(BE\text{-}MPK, S, i, sk_i, c) = m$*.*

Remark 2. We observe that the standard definition of a BE scheme (as per [6,18]) requires the specification of the broadcast target set S as an input to the decryption algorithm.

We now define a model for an adaptively secure BE scheme. This model is slightly different from those used in [6,18] in that it gives the adversary access to a private key extraction oracle also after the Challenge phase.

Definition 26. *Consider the following game.*
Setup. *The challenger $\mathcal{C}$ runs* `BE.Setup`*(κ,n,l) to generate master public key BE-MPK and master secret key BE-MSK and gives BE-MPK to the adversary $\mathcal{A}$.*
Phase 1. *$\mathcal{A}$ can adaptively issue queries to a private key extraction oracle for any index $i \in \{1, ..., n\}$. This oracle will respond by returning $sk_i =$* `BE.Key-Gen`*$(BE\text{-}MPK, BE\text{-}MSK, i)$.*
Challenge. *$\mathcal{A}$ selects two messages m_0 and $m_1 \in \mathcal{MSP}$ and a challenge set $S^* \subseteq \{1, ..., n\}$, such that $i \notin S^*$ for all queries i made in Phase 1. $\mathcal{A}$ passes m_0, m_1, S^* to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit b and computes $c^* =$* `BE.Enc`*$(BE\text{-}MPK, m_b, S^*)$. c^* is passed to $\mathcal{A}$.*
Phase 2. *$\mathcal{A}$ continues to make queries to the private key extraction oracle with the same restriction as in the Challenge phase.*
Guess. *The adversary outputs its guess b' for b.*

$\mathcal{A}$'s advantage in the above game is defined as $Adv_{\mathcal{A}}(\kappa, n, l) = \left|\Pr[b' = b] - \frac{1}{2}\right|$.

Definition 27. *We say that a BE scheme is* adaptively secure *if all polynomial-time adversaries have at most negligible advantage in the above game.*

We now show how to construct a Plain TSE scheme from a BE scheme. We note that by Remark 2 the resulting Plain TSE scheme is inherently prevented from having the DTI confidentiality property. This is because this interval will need to be specified as an input to the decryption algorithm in the Plain TSE scheme, a feature that arises from the construction of this algorithm from the corresponding algorithm of the BE scheme.

Consider a BE scheme $B =$ (`BE.Setup`, `BE.Key-Gen`, `BE.Enc`, `BE.Dec`) with message space $\mathcal{MSP}$. We will use B to construct $X =$ (`Plain.Setup`,`Plain.TIK-Ext`,`Plain.Enc`,`Plain.Dec`), a Plain TSE scheme with the same message space, in the following way:
– `Plain.Setup`(κ, T). Run `BE.Setup` on input κ, $n = T$ and $l = T$ to obtain outputs BE-MPK, BE-MSK. Set TS-MPK = BE-MPK and TS-MSK = BE-MSK.
– `Plain.TIK-Ext`(TS-MPK, TS-MSK, t). Run `BE.Key-Gen` on the same inputs to obtain the TIK k_t for t.
– `Plain.Enc`(TS-MPK, m, $[t_0, t_1]$). Run `BE.Enc`(TS-MPK, m, S), where S is set to the interval $[t_0, t_1]$, to obtain a ciphertext c. c must contain an explicit description of S.
– `Plain.Dec`(TS-MPK, c, k_t). Run `BE.Dec`(TS-MPK, S, t, k_t, c) where S is extracted from c and t is extracted from k_t, to obtain either a message m or a failure symbol $\perp$.

We observe that in addition to not having the DTI confidentiality property, the Plain-TSE scheme constructed from a BE scheme as above also implicitly requires the TIK k_t to contain a description of t. For the above construction, the following result holds.

Theorem 6. *Let B be a fully collusion resistant, adaptively secure BE scheme. Then the Plain TSE scheme X constructed from B as above is IND-CPA secure in the sense of Definition 3, and is correct.*

Proof. Suppose we have an adversary $\mathcal{A}$ against the Plain TSE scheme X with advantage $\epsilon(\kappa)$. We will construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the BE scheme B used in the construction. The game proceeds as follows.
Setup. The challenger $\mathcal{C}$ runs `BE.SetupGen`(κ, T, T) to obtain a pair (BE-MPK, BE-MSK). It gives BE-MPK to $\mathcal{B}$, who passes it on to $\mathcal{A}$.
Phase 1. $\mathcal{A}$ can adaptively issue TIK extraction queries for any time $t \in T$. Such a query is passed on to $\mathcal{B}$, who gives it to $\mathcal{C}$ as private key query for user t. $\mathcal{C}$ will respond to each query with the private key sk_t, which is passed to $\mathcal{B}$, who then forwards it to $\mathcal{A}$ as the TIK for time t.
Challenge. $\mathcal{A}$ selects two messages m_0 and $m_1 \in \mathcal{MSP}$ and a challenge interval $[t_0, t_1] \subseteq T$, such that $t \notin [t_0, t_1]$ for any of the queries t in made in Phase 1. $\mathcal{A}$ passes $m_0, m_1, [t_0, t_1]$ to $\mathcal{B}$, who passes them to $\mathcal{C}$. $\mathcal{C}$ chooses a random bit b and computes $c^* =$ `BE.Enc`(BE-MPK,m_b, $[t_0, t_1]$). c^* is passed to $\mathcal{B}$, who in turn passes it to $\mathcal{A}$.
Phase 2. $\mathcal{A}$ continues to issue TIK extraction queries with the same restriction as in the Challenge phase. These queries are dealt with by $\mathcal{B}$ as in Phase 1.
Guess. The adversary $\mathcal{A}$ outputs its guess b' for b and $\mathcal{B}$ outputs the same guess.

$\mathcal{B}$ perfectly simulates the IND-CPA game for $\mathcal{A}$. Hence $\mathcal{A}$'s advantage is as it would be in the real game and by construction $\mathcal{B}$ wins whenever $\mathcal{A}$ does. □

C Proof of Theorem 4

We recall the statement of Theorem 4:

Theorem 4. *Let I be a correct, selective-id IND-CPA$_{TS}$ secure ID-TSE scheme and Σ a strongly unforgeable one-time signature scheme. Then Γ, as constructed in section 4.3, is an IND-CCA$_{TS}$ secure PK-TSE scheme.*

Proof. Our proof follows closely the proof of [8], with suitable modifications. Given an IND-CCA$_{TS}$ adversary $\mathcal{A}$ against Γ we construct an adversary $\mathcal{B}$ that will interact with $\mathcal{A}$ to break the selective-id IND-CPA$_{TS}$ security of I. Before presenting the game, we note the following important considerations. We start by saying that a ciphertext $C = (vk, c, \sigma)$ is valid if $\texttt{Ver}(vk, c, \sigma) = 1$. Let $C^* = (vk^*, c^*, \sigma^*)$ denote the challenge ciphertext received by $\mathcal{A}$ during the game, and let `Forge` denote the event that in this game $\mathcal{A}$ submits a valid ciphertext $C = (vk^*, c, \sigma)$ to its decryption oracle. It can be shown that $\mathcal{A}$ can be used to break the underlying one-time signature scheme Σ with probability exactly $\Pr_{\mathcal{A},\Gamma}[\texttt{Forge}]$. Since Σ is a strongly unforgeable one-time signature (as defined in Appendix A), we can assume that $\Pr_{\mathcal{A},\Gamma}[\texttt{Forge}]$ is negligible in the security parameter κ.

We now describe the simulation.

Init. $\mathcal{B}$ runs $\texttt{G}(\kappa)$ to get $(vk^*, sigk^*)$. The string vk^* will be the target identity used by $\mathcal{B}$.

Setup. $\mathcal{C}$ runs `TS-Setup`, `ID-Setup` to get (TS-MPK, TS-MSK, ID-MPK, ID-MSK). It gives $\mathcal{B}$ (TS-MPK, TS-MSK, ID-MPK). $\mathcal{B}$ passes the same information to $\mathcal{A}$.

Phase 1. $\mathcal{A}$ can issue queries of the form (C, t) to its Decrypt oracle, where C is of the form (vk, c, σ). $\mathcal{B}$ responds as follows:

- if $\texttt{Ver}(vk, c, \sigma) \neq 1$ then $\mathcal{B}$ returns $\perp$;
- if $\texttt{Ver}(vk, c, \sigma) = 1$ and $vk = vk^*$ then $\mathcal{B}$ aborts and outputs a random bit (event `Forge` just occurred);
- if $\texttt{Ver}(vk, c, \sigma) = 1$ and $vk \neq vk^*$ then $\mathcal{B}$ makes a query vk to its private key extraction oracle to obtain sk_{vk}. $\mathcal{B}$ can compute the TIK k_t as it obtained TS-MSK from its challenger. $\mathcal{B}$ then computes $\texttt{ID.Dec}$(TS-MPK, ID-MPK, k_t, sk_{vk}) and obtains either a message m or failure symbol $\perp$, which is passed to $\mathcal{A}$.

Challenge. $\mathcal{A}$ outputs m_0, m_1 and $[t_0, t_1]$ and passes the tuple $(m_0, m_1, [t_0, t_1])$ to $\mathcal{B}$, who then sends $(m_0, m_1, [t_0, t_1], vk^*)$ to $\mathcal{C}$ as its challenge query. $\mathcal{C}$ picks a random bit b and computes $c^* = \texttt{ID.Enc}$(TS-MPK, ID-MPK, m_b, $[t_0, t_1]$, vk^*). $\mathcal{C}$ gives c^* to $\mathcal{B}$, who computes $\sigma^* = \texttt{Sign}(sigk^*, c^*)$ and returns $C^* = (vk^*, c^*, \sigma^*)$ to $\mathcal{A}$.

Phase 2. $\mathcal{A}$ can continue issuing queries of the form (C, t), where $C = (vk, c, \sigma)$, with the restriction that $(C, t) \neq (C^*, t')$, where $t' \in [t_0, t_1]$. If $\texttt{Ver}(vk, c, \sigma) \neq 1$ then $\mathcal{B}$ returns $\perp$. Otherwise, $\mathcal{B}$ will respond as described in the following two cases:

- Case 1: $C \neq C^*$.
  - if $c \neq c^*$ and $vk = vk^*$ then $\mathcal{B}$ aborts and outputs a random bit (event `Forge` just occurred);
  - if $c = c^*$ and $vk = vk^*$ then $\mathcal{B}$ aborts and outputs a random bit (event `Forge` just occurred);
  - if $c = c^*$ and $vk \neq vk^*$ then $\mathcal{B}$ passes the oracle query vk to its challenger to obtain sk_{vk}. $\mathcal{B}$ then computes the TIK k_t using TS-MSK. $\mathcal{B}$ then computes $\texttt{ID.Dec}$(TS-MPK, ID-MPK, k_t, sk_{vk}) and obtains either a message m or failure symbol $\perp$, which it passes to $\mathcal{A}$.
- Case 2: $C = C^*$ and $t' \notin [t_0, t_1]$.
  In this case, the correctness of the scheme I implies that decryption algorithm `ID.Dec` applied to c^* with key k_t and identity vk^* should output $\perp$, so $\mathcal{B}$ returns $\perp$.

Guess. $\mathcal{A}$ outputs b' as its guess for b. $\mathcal{B}$ outputs the same bit.

We note that $\mathcal{B}$ provides a perfect simulation for $\mathcal{A}$ as well as a legal strategy for attacking scheme I, provided that $\mathcal{B}$ is not forced to abort (a situation that occurs only when the event `Forge` occurs).

In particular $\mathcal{B}$ never queries its challenger for the private key corresponding to the target identity vk^*. We hence have that $\mathcal{B}$ wins if $\mathcal{A}$ does, and this can only happen when the event `Forge` does not occur. In that case, $\mathcal{B}$ is forced to abort and outputs a random bit. We therefore have

$$|\Pr_{\mathcal{B},I}[\mathtt{Succ}] - \tfrac{1}{2}| = |\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \neg\mathtt{Forge}] + \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] - \tfrac{1}{2}|.$$

We observe that the right-hand side of the above equation is negligible since both $\left|\Pr_{\mathcal{B},I}[\mathtt{Succ}] - \frac{1}{2}\right|$ and $\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}]$ are negligible, assuming the security of schemes I and Σ. Finally we have:

$$\begin{aligned}
&|\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ}] - \tfrac{1}{2}| \\
&= |\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \mathtt{Forge}] + \Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \neg\mathtt{Forge}] - \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] + \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] - \tfrac{1}{2}| \\
&\leq |\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \mathtt{Forge}] - \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}]| + |\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \neg\mathtt{Forge}] + \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] - \tfrac{1}{2}| \\
&\leq \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] + |\Pr_{\mathcal{A},\Gamma}[\mathtt{Succ} \wedge \neg\mathtt{Forge}] + \tfrac{1}{2}\Pr_{\mathcal{A},\Gamma}[\mathtt{Forge}] - \tfrac{1}{2}|,
\end{aligned}$$

where on the right-hand side we have two negligible quantities. We have hence proved that $\mathcal{A}$'s advantage in winning the IND-CCA$_{TS}$ game is negligible. □