A New Construction of Time Capsule Signature

Miaomiao Zhang[1], Gongliang Chen[3], Jianhua Li[1,3],
Licheng Wang[2] and Haifeng Qian[2]

[1]Dept. of Electronic Engineering, Shanghai Jiao Tong University
[2]Dept. of Computer Science and Engineering, Shanghai Jiao Tong University
[3]Sch. of Infomation Security Engineering, Shanghai Jiao Tong University

mmzhang_sjtu@sjtu.edu.cn

Abstract. In this paper we introduce a new approach of constructing time capsule signature, which not only captures the basic requirements, but also more straightforward and flexible in contrast to the scheme given by Dodis *et al*[1]. The time capsule signature provides an elegant way to produce a "future signature" that becomes valid from a specific future time t, when a trusted third party (called *Time Server*) publishes some trapdoor information associated with the time t. It also has many other advantages. Our work includes a developed security model of time capsule signature, a novel way of construction based on the bipartite ring signature, which is proven secure in the random oracle model and a concrete realization of the scheme.

1 Introduction

1.1 Motivation

The notion of a digital signature may prove to be one of the most fundamental and useful inventions of modern cryptography. A signature scheme provides a way for each user to sign messages so that the signatures can later be verified by anyone else. More specifically, each user can create a matched pair of private and public keys so that only he can create a signature for a message (using his private key), but anyone can verify the signature for the message (using the signer's public key). The verifier can convince himself that the message contents have not been altered since the message was signed. Also, the signer can not later repudiate having signed the message, since no one but the signer possesses his private key.

In an ordinary signature scheme, the validity of a signature value is determined at the point of signature generation and never changes unless the signer's public key is revoked. Users cannot generate the so-called "future signature" which is not currently valid but becomes valid from a future time t. A possible way to achieve this is signing with a statement like "the signature of message m becomes valid from time t". However, this has several drawbacks. The verifier has to be aware of the current time. And more seriously, in this solution the

signer herself loses control over the validity of the "future signature". Even the real signer cannot make her signature valid until time t, which could be undesirable in certain situations. In another solution, the signer can issue a new but also independent signature of m before time t. However, this may also cause some problem in certain situations. For instance, the verifier can distinguish whether the message m was signed in the "future" or "regular" way, which seems to be unnecessary in most situations. And in case the message m carries some monetary value, the signer needs to make sure that no "double spending" occurs, that is the signer can revoke the original signature, so that it will not become valid at time t.

Therefore, we need a solution where the signer can issue a future signature which should satisfy the following properties:

- After the signature creation, the recipient can make sure that the signature will become valid by time t, even if the signer refuses to cooperate after she produces the future signature.
- The signer can make his future signature valid at any time after the initial creation.
- The resulting signatures are indistinguishable. In other words, the verifier cannot tell whether the signature is validated by the signer earlier, or it became "automatically valid" at time t.

1.2 Related Work

It is crucial to specify the mechanism under which the signature can be "automatically" completed at time t (which we call "hatching" as opposed to "prehatching" which can be done by the signer at any time).

There are two main lines of work related to cryptographic time capsule, depending on whether or not involve a trusted third party.

The first approach, which related to timed-release cryptography, is to ensure that the encryption, commitment or signature can be opened in a brute force way by solving some time-consuming, but computationally feasible problem. For instance, Dwork and Naor [6] used such moderately hard functions in order to deter abuse of resources such as spamming. Bellare and Goldwasser [7, 8] suggested "verifiable time capsules" for key escrowing in order to deter widespread wiretapping. Rivest, Shamir and Wagner [9] suggested "time-lock puzzle", where the goal is to design "inherently sequential puzzles" which are resistant to parallel attacks. This was formally addressed by Boneh and Naor [2], who defined (verifiable) timed commitments. As one of their applications, they get an analog of "timed signature" which is termed "time capsule signature" by Dodis *et al*, where the future signature can either be opened by the signer, or by the recipient-the latter if the recipient solves a moderately hard problem. Then[10, 11] made some advances in this work.

The second approach is based on the trusted third party(TTP). It has two main flavors: optimistic fair exchange of digital signatures, and identity-based future encryption. In the former case, the server needs to resolve all individual

signatures where the signer refused to validate the signature. Representative examples include [3, 4, 13, 5]. In the case of future encryption [14, 15, 12], the main problem addressed was that the sender wants to ensure that the message would remain hidden before the *Time Server* would publish the corresponding trapdoor.

Dodis *et al* introduced the notion of time capsule signatures[1], which is is a "future signature" that becomes valid from a specific future time t, when a trusted third party (the arbitrator which called the *Time Server*) publishes some trapdoor information associated with the time t. In addition, time capsule signature should satisfy the following properties:

- If the signer wants, she can make her time capsule signature effective before the pre-defined time t.
- The recipient of "future signature" can verify right away that the signature will become valid no later than at time t.
- *Time Server* need not contact any user at any time, and in fact does not need to know anything about the PKI employed by the users.
- Signatures completed by the signer before time t are indistinguishable from the ones completed using the *Time Server* at time t.

More specifically, in a time capsule signature scheme, when Alice gives Bob her time capsule signature σ_t for a future time t, Bob can verify that Alice's time capsule signature will become valid from the time t. In addition, if Alice wishes, she can make her time capsule signature effective before the predefined time t. The assumption on *Time Server* is minimal, for the *Time Server* only publishes some information at the beginning of each time period and need not contact any user at any time. They also mentioned that the concept of time capsule signature can be generalized to event capsule signature, where *Event Server* issues the notification information of specific events. The event capsule signature becomes valid if a specific event happens or the signer makes valid before the event occurs.

They also provide a generic construction based on a primitive called identity-based trapdoor hard-to-invert relation (ID-THIR). In brief, ID-THIR is given by a family $\mathcal{R}$ of relations R_{id}, where (1) it is easy to sample a random pair $(c, d) \in R_{id}$ and verify if the pair (c, d) belongs to R_{id}; (2) for each identity id, there exists a trapdoor td_{id}, which allows one to compute a random d corresponding to any given c (The trapdoor td_{id} can be efficiently computed from a single "master trapdoor" $mtd_{\mathcal{R}}$); (3) without the trapdoor, it is hard to find a matching d corresponding to a randomly sampled c, even if one knows many trapdoors corresponding to identities $id'(= id)$.

A generic construction of time capsule signature from ID-THIR is given in [1] and the security of that scheme is based on the properties of the ID-THIR scheme. However, they just give constructions of THIR and do not give any concrete realization of ID-THIR. We find that it is not trivial to concretely construct a ID-THIR.

1.3 Our Contribution

Our contribution lies in developing the security notions of time capsule signature and providing a novel way of construction from an original signature scheme and a bipartite ring signature scheme. Our construction of time capsule signature is very nature, plain and flexible and it is proven secure in the random oracle model if the underlying building blocks(the ordinary signature scheme and the ring signature scheme) are secure. Concrete time capsule signature scheme can be easily realized from our generic construction.

The rest of this paper is organized as follows: In section 2, we recall the components of digital signature scheme and ring signature scheme, and their security definitions. In section 3, we first introduce the definitions of time capsule signature and develop the underlying security model. Then we give our new construction based on a bipartite ring signature and its security proof. Then in section 4, we present a concrete and full time capsule signature scheme from our generic construction and analyze its security. Finally, concluding remarks are made in Section 5.

2 Preliminaries

2.1 Security Notions of Digital Signature

We first review the formal definition of a generic digital signature scheme[18], which we denote by $\mathcal{DS}$.

Definition 1 [Digital Signature Scheme] A digital signature scheme $\mathcal{DS} = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ is defined by the following algorithms:

- on input 1^k, where k is the security parameter, the algorithm $\mathcal{G}$ produces a pair (pk, sk) of matching public and secret keys. Algorithm $\mathcal{G}$ is probabilistic.
- Giving a message m and a pair of matching public and secret keys (pk, sk), $\mathcal{S}$ can produce a signature σ. The signing algorithm might be probabilistic.
- Given a signature σ, a message m and a public key pk, $\mathcal{V}$ tests whether σ is a valid signature of m with respect to pk. In general, the verification algorithm need not be probabilistic.

Formally, Let $\mathcal{O}_{ALG}$ be an oracle simulating the algorithm ALG. Let the attacker F on input the public parameter $param$ have access to $\mathcal{O}_{ALG}$, denoted as $\mathsf{F}^{\mathcal{O}_{ALG}}(param)$. $Query(\mathsf{F}, \mathcal{O}_{ALG})$ is the set of queries F asked to the algorithm ALG simulated by oracle $\mathcal{O}$.

Definition 2 [UF-CMA] Let $\mathcal{DS} = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ be a digital signature scheme, and let A be an attacker assumed to be a probabilistic Turing machine taking a security parameter k as input. Consider the following experiment of running A in an attack on digital signature scheme $\mathcal{DS}$ as the following. Notice that the public key is provided as an input to A.

Experiment $\mathbf{Exp}^{\text{UF-CMA}}_{\mathcal{DS},\,\mathsf{A}}$

Let $(pk, sk) \xleftarrow{R} \mathcal{G}(1^k)$

Let $(m, \sigma) \leftarrow \mathsf{A}^{\mathcal{O}_{\mathcal{S}}}(pk)$
If $\mathcal{V}(m, \sigma) = 1$ and $m \notin Query(\mathsf{A}, \mathcal{O}_{\mathcal{S}})$
Then return 1 else return 0

We define $\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}$ be the probability that experiment $\mathbf{Exp}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}$ returns 1.

$$\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}(k) = Pr[\mathcal{V}(m, \sigma) = 1]$$

Then for any t, q let

$$\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS}}(t, q) = \max_{\mathsf{A}}\{\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}\}$$

where the maximum is over all A such that the execution time of experiment $\mathbf{Exp}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}$ is at most t and the number of oracle queries made by A is at most q. Note that the running time and the number of queries are all polynomial in the security parameter k.

In practice, the queries correspond to messages signed by the legitimate sender, and it would make sense that getting these examples is more expensive than just computing on one's own. That is, we would expect q to be smaller than t.

The scheme $\mathcal{DS}$ is said to be UF-CMA secure if $\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS}}(t, q)$ is negligible in k.

2.2 Security Notions of Ring Signature

The concept of ring signature was formalized by Rivest *et al.* in [17]. According to [16, 20], we give a concise formalized definition of ring signature scheme as follows.

Definition 3 [Ring Signature Scheme] A ring signature scheme, which we denote by $\mathcal{RS}$, is a triple of algorithms $(\mathcal{G}, \mathcal{S}, \mathcal{V})$

- $(S_i, P_i) \leftarrow \mathcal{G}(1^k)$ is a probabilistic polynomial time algorithm which takes security parameter k and outputs private key S_i and corresponding public key P_i of a user i.
- $\sigma \leftarrow \mathcal{S}(m, \langle P_r \rangle, S_s)$ is a probabilistic polynomial time algorithm which takes as inputs a message m, secret key S_s of signer s, and r possible signers' public key set $\langle P_r \rangle$ which includes the one corresponding to S_s, produces a signature σ.
- $1/0 \leftarrow \mathcal{V}(\langle P_r \rangle, m, \sigma)$ is a deterministic polynomial time algorithm which takes a message m, a signature σ and a set of r public keys $\langle P_r \rangle$ as inputs, returns 1 or 0 for accept or reject, respectively.

It is required that for any message m, any (S_s, P_s) generated by $\mathcal{G}(1^k)$ and any $\langle P_r \rangle$ that includes P_s, the equation $\mathcal{V}(\langle P_r \rangle, m, \mathcal{S}(1^k, S_s)) = 1$ satisfies.

The security of ring signature schemes has two aspects: *unforgeability* and *anonymity*.

Definition 4 [Unforgeability of Ring Signature] Let $\mathcal{RS} = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ be a ring signature scheme, and let A be a PPT (probabilistic polynomial-time) adversary taking a security parameter k as input. Consider an experiment of running A in an attack on ring signature scheme $\mathcal{RS}$ as the following.

Experiment $\mathbf{Exp}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}$
Let $(P_i, S_i) \xleftarrow{R} \mathcal{G}(1^k)$
Let $(m, \sigma) \leftarrow \mathsf{A}^{\mathcal{O}_S}(\langle P_r \rangle)$
If $\mathcal{V}(m, \sigma, \langle P_r \rangle) = 1$ and $m \notin Query(\mathsf{A}, \mathcal{O}_S)$
Then return 1 else return 0

Define $\mathbf{Succ}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}$ be the probability that experiment $\mathbf{Exp}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}$ returns 1.

$$\mathbf{Succ}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}(k) = Pr[\mathcal{V}(m, \sigma) = 1]$$

Then for any t, q let

$$\mathbf{Succ}^{\mathrm{UF}}_{\mathcal{RS}}(t, q) = \max_{\mathsf{A}}\{\mathbf{Succ}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}\}$$

where the maximum is over all A such that the execution time of experiment $\mathbf{Exp}^{\mathrm{UF}}_{\mathcal{RS},\,\mathsf{A}}$ is at most t, the number of oracle queries made by A is at most q.

A ring signature scheme is unforgeable if for any PPT adversary A, $\mathbf{Succ}^{\mathrm{UF}}_{\mathcal{RS}}(t, q)$ is negligible. A function ϵ is negligible if for all polynomials π, $\epsilon(k) < 1/\pi(k)$ holds for all sufficiently large k.

Definition 5 [Anonymity of Ring Signature] Let $\mathcal{RS} = (\mathcal{G}, \mathcal{S}, \mathcal{V})$ be a ring signature scheme, and let A be a PPT adversary taking a security parameter k as input. Consider the following experiment of running A in an attack on $\mathcal{RS}$.

Experiment $\mathbf{Exp}^{\mathrm{ANON}}_{\mathcal{RS},\,\mathsf{A}}$
Let $(\langle P_r \rangle, m) \leftarrow \mathsf{A}^{\mathcal{G}}(1^k)$
Let $(i_0, i_1) \leftarrow \mathsf{A}$, where i_0, i_1 are indices that $P_{i_0}, P_{i_1} \in \langle P_r \rangle$
$b \xleftarrow{R} \{0, 1\}$
Let $\sigma \leftarrow \mathcal{S}(S_{i_b}, \langle P_r \rangle, m)$
$\tilde{b} \leftarrow \mathsf{A}^{\mathcal{O}_S}(\langle P_r \rangle, i_0, i_1)$
If $\tilde{b} = b$
Then return 1 else return 0

We define $\mathbf{Succ}^{\mathrm{ANON}}_{\mathcal{RS},\,\mathsf{A}}$ be the probability as follows

$$\mathbf{Succ}^{\mathrm{ANON}}_{\mathcal{RS},\,\mathsf{A}}(k) = |Pr[\tilde{b} = b] - 1/2|$$

Then for any t, q let

$$\mathbf{Succ}^{\mathrm{ANON}}_{\mathcal{RS}}(t, q) = \max_{\mathsf{A}}\{\mathbf{Succ}^{\mathrm{ANON}}_{\mathcal{RS},\,\mathsf{A}}\}$$

where the maximum is over all A such that the execution time of experiment $\mathbf{Exp}^{\mathrm{ANON}}_{\mathcal{RS},\,\mathsf{A}}$ is at most t, the number of oracle queries made by A is at most q.

A ring signature scheme is anonymous if for any PPT adversary A, $\mathbf{Succ}^{\mathrm{ANON}}_{\mathcal{RS}}(t, q)$ is negligible.

3 Time Capsule Signature

3.1 Defination

Definition 6 [**time capsule signature scheme**] A time capsule signature scheme is specified by an 8-tuple of PPT algorithms ($Setup^{TS}$, $Setup^{User}$, $TSign$, $TVerify$, $TRelease$, $Hatch$, $PreHatch$, $Verify$) such that:

- $Setup^{TS}$: This setup algorithm is run by Time Server. It takes a security parameter as input and returns a public/private time release key pair (TPK, TSK).
- $Setup^{User}$: This setup algorithm is run by each user. It takes a security parameter as input and returns the user's public/private key pair (PK, SK).
- $TSign$: The time capsule signature generation algorithm $TSign$ takes as input (m, SK, TPK, t), where t is the specific time from which the signature becomes valid. It outputs a time capsule signature σ'_t.
- $TVerify$: The time capsule signature verification algorithm $TVerify$ takes as input $(m, \sigma'_t, PK, TPK, t)$ and outputs 1 (accept) or 0 (reject).
- $TRelease$: The time release algorithm $TRelease$ is run by Time Server and takes as input (t, TSK). At the beginning of each time period t, Time Server publishes $Z_t = TRelease(t, TSK)$. Note that Time Server dose not contact any user at any time and need not know anything about the users.
- $Hatch$: This algorithm is run by any party and is used to open a valid time capsule signature which became mature. It takes as input $(m, \sigma'_t, PK, TPK, Z_t)$ and returns the hatched signature σ_t.
- $PreHatch$: This algorithm is run by the signer and used to open a valid time capsule signature which is not mature yet. It takes as input $(m, \sigma'_t, SK, TPK, t)$ and returns the pre-hatched signature $\hat{\sigma_t}$.
- $Verify$: This algorithm is used to verify a hatched (or pre-hatched) signature. $Verify$ takes as input $(m, \sigma_t/\hat{\sigma_t}, PK, TPK, t)$ and returns 1 (accept) or 0 (reject).

Correctness property states that

- $TVerify(m, Tsig(m, SK, TPK, t), PK, TPK, t)$
- $Verify(m, \sigma_t/\hat{\sigma_t}, PK, TPK, t)$, where $\sigma_t = Hatch(m, \sigma'_t, PK, TPK, Z_t)$ or $\hat{\sigma_t} = PreHatch(m, \sigma'_t, SK, TPK, t)$.

3.2 Security of Time Capsule Signature

In the time capsule signature scheme, the *Time Server* is assumed to be a trusted third party, which implies that it's unnecessary to consider any dishonest behavior of the *Time Server*. We define the security of time capsule signature as follows.

Definition 7 [**Security of Time Capsule Signature**] The security of time capsule signatures consists of four aspects:

– *Unforgeability of the immature signature.* We require that any PPT adversary A succeeds with at most negligible probability in the following experiment.

Experiment $\mathbf{Exp}^{\text{UF-imTCS}}_{\mathcal{TCS},\ \mathsf{A}}$
Let $(TPK, TSK) \leftarrow Setup^{TS}(1^k)$
Let $(PK, SK) \leftarrow Setup^{User}(1^k)$
$(m, t, \sigma'_t) \leftarrow \mathsf{A}^{\mathcal{O}_{TSign}}(PK, TPK)$
If $TVerify(m, \sigma'_t, PK, TPK, t) = 1$
and $(m, \tilde{t}) \notin Query(\mathsf{A}, \mathcal{O}_{TSign})$ for all $\tilde{t}$
Then return 1 else return 0

where $Query(\mathsf{A}, \mathcal{O}_{Tsign})$ is the set of queries A asked to the time capsule signature generation oracle $\mathcal{O}_{TSign}$. Define $\mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS},\ \mathsf{A}}$ be the probability that experiment $\mathbf{Exp}^{\text{UF-imTCS}}_{\mathcal{TCS},\ \mathsf{A}}$ returns 1.
In other words, no one can forge an immature signature of m.

– *Un-Prehatchable by dishonest users* We require that any PPT adversary B succeeds with at most negligible probability in the following experiment.

Experiment $\mathbf{Exp}^{\text{UPH-TCS}}_{\mathcal{TCS},\ \mathsf{B}}$
Let $(TPK, TSK) \leftarrow Setup^{TS}(1^k)$
Let $(PK, SK) \leftarrow Setup^{User}(1^k)$
$(m, t, \sigma'_t) \leftarrow Tsign(m, SK, TPK, t)$
Let $(m, t, \sigma_t) \leftarrow \mathsf{B}^{\mathcal{O}_{TR}, \mathcal{O}_{PreH}}(PK, TPK)$
If $Verify(m, \sigma_t, PK, TPK, t) = 1$ and
$t \notin Query(\mathsf{B}, \mathcal{O}_{TR})$ and $(m, t, \tilde{\sigma'_t}) \notin Query(\mathsf{B}, \mathcal{O}_{PreH})$
for $\forall \tilde{\sigma'_t}$ such that $TVerify(m, \sigma'_t, PK, TPK, t) = 1$
Then return 1 else return 0

where $Query(\mathsf{B}, \mathcal{O}_{TR})$ is the set of queries of B asked to the time release oracle $\mathcal{O}_{TR}$, and $Query(\mathsf{B}, \mathcal{O}_{PreH})$ is the set of valid queries B asked to the oracle $\mathcal{O}_{PreH}$. In other words, no one should be able to open a pre-mature time capsule signature without help of the signer or Time Server. Notice that adversary B can make any time release query to $\mathcal{O}_{TR}$ except the target time t. Therefore, the above experiment requires strong security guaranteeing both forward and backward secrecy.
Define $\mathbf{Succ}^{\text{UPH-TCS}}_{\mathcal{TCS},\ \mathsf{B}}$ be the probability that experiment $\mathbf{Exp}^{\text{UPH-TCS}}_{\mathcal{TCS},\ \mathsf{B}}$ returns 1.

– *Non-cheating of the signer.* We require that any PPT adversary C succeeds with at most negligible probability in the following experiment.

Experiment $\mathbf{Exp}^{\text{NCH-TCS}}_{\mathcal{TCS},\ \mathsf{C}}$
Let $(TPK, TSK) \leftarrow Setup^{TS}(1^k)$
$(m, t, \sigma'_t, PK) \leftarrow \mathsf{C}^{\mathcal{O}_{TR}}(TPK)$
Let $Z_t \leftarrow TRelease(t, TSK)$
Let $\sigma_t \leftarrow Hatch(m, \sigma'_t, PK, TPK, Z_t)$
If $TVerify(m, \sigma'_t, PK, TPK, t) = 1$ and
$Verify(m, \sigma_t, PK, TPK, t) = 0$
Then return 1 else return 0

Define $\mathbf{Succ}^{\text{NCH-TCS}}_{\mathcal{TCS},\,\mathsf{C}}$ be the probability that experiment $\mathbf{Exp}^{\text{NCH-TCS}}_{\mathcal{TCS},\,\mathsf{C}}$ returns 1.

In other words, adversary A should not be able to produce a time capsule signature σ'_t , where σ'_t looks good to the verifier but cannot be hatched into A's full signature by the Time Server.

– *Indistinguishability.*

It is required that the hatched signature and the prehatched signature are computationally indistinguishable. Let D be any PPT distinguisher. We require that the success probabilisty of D in the following experiment is negligibly close to 1/2.

Experiment $\mathbf{Exp}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}$
Let $(TPK, TSK) \leftarrow Setup^{TS}(1^k)$
Let $(PK, SK) \leftarrow Setup^{User}(1^k)$
Let $(m, t, \sigma'_t) \leftarrow Tsign(m, SK, TPK, t)$
Let $d \xleftarrow{R} \{0, 1\}$
If $d = 0$, $\sigma_t \leftarrow Hatch(\sigma'_t)$
else $\sigma_t \leftarrow PreHatch(\sigma'_t)$
Let $\tilde{d} \leftarrow \mathsf{D}^{\mathcal{O}_{TR}, \mathcal{O}_{Hatch}, \mathcal{O}_{PreH}}(PK, TPK, \sigma'_t)$
If $\tilde{d} = d$
Then Return 1 else reture 0

Define the success probability of D as

$$\mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}} = |Pr[\mathbf{Exp}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}} = 1] - 1/2|$$

For any t, q we define the success probability of distinguishing $\mathcal{TCS}$ signature via

$$\mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS}}(t, q) = \max_{\mathsf{D}}\{\mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}\}$$

Definition 8 [UF-TCS-CMA] A $\mathcal{TCS}$ scheme is sad to be UF-TCS-CMA secure if the *Unforgeability of the immature signature* property and the *Un-Prehatchable by dishonest users* property(mentioned in Definition 7) achieved.

It is obvious

$$\mathbf{Succ}^{\text{UF-TCS-CMA}}_{\mathcal{TCS}} \leq \mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS}} + \mathbf{Succ}^{\text{UPH-TCS}}_{\mathcal{TCS}} \quad (1)$$

Notice that the defininationabove is just an unforgeability notion for the $\mathcal{TCS}$. We give this defininition in order to generalize the unforgery aspect of the time capsule signature, whereas the full description of security is proposed in Definition 7.

3.3 Generic Construction Based on Ring Signature

In this part we give a generic construction of time capsule signature scheme based on bipartite ring signature.

Let $\mathcal{DS}$=(Set, Sig, Ver) be an ordinary signature scheme.

- $Setup^{TS}$: This *Time Server* takes a security parameter k as input and sets a public/private time release key pair $(TPK, TSK) = (SK_{TS}, PK_{TS})$.
- $Setup^{User}$: Each user i sets his public/private key pair (PK, SK)=(SK_i,PK_i) by running $Set(1^k)$.
- $TSign$: To generate a probabilistic time capsule signature on a message m for time t, the signer s first gets the hash value $h = H_0(m||PK_{TS}||t)$ and computes $\sigma'_t = Sig_{SK_s}(h)$ as a time capsule signature.
- $TVerify$: For a given signature σ'_t, the verifier checks whether σ'_t is a valid signature on $h = H_0(m||PK_{TS}||t)$ by running $Ver_{PK_s}(h, \sigma'_t)$.
- $TRelease$: For a given time value t, *Time Server* computes a ring signature relate to user s
  - Determine the symmetric key: The Time Server first computes the hash of his public key, the Signer's ID and time t as the symmetric key k: $k = H_1(PK_{TS}, ID_s, t)$
  - Pick a random glue value: The *Time Server* picks an initialization(glue) value v uniformly at random from $(0, 1)^b$.
  - Pick random value for signer: The *Time Server* picks a random value x_s uniformly from $(0, 1)^b$ and computes $y_s = g_s(x_s)$, where g_s is an extended trapdoor permutation[17] corresponding to s.
  - Solve for y_{TS}: The *Time Server* solves the following ring equation for y_{TS}: $C_{k,v}(y_s, y_{TS}) = v$
  - Invert the *Time Server*'s trapdoor permutation: The *Time Server* uses his trapdoor information in order to invert g_{TS} on y_{TS} to obtain x_{TS}: $x_{TS} = g_{TS}^{-1}(y_{TS})$
  - publish the ring signature: The ring signature part is defined to be the 5-tuple: $Z_t = (PK_s, PK_{TS}; v; x_s, x_{TS})$ and the *Time Server* publishes Z_t.
- $Hatch$: To open an mature time capsule signature σ'_t, a party combine σ'_t with Z_t and returns the hatched signature $\sigma_t = (\sigma'_t, PK_s, PK_{TS}; v; x_s, x_{TS})$.
- $PreHatch$: To open a valid pre-mature time capsule signature σ'_t, the signer should computes a valid ring signature himself:
  - computes the symmetric key: The Signer computes the symmetric key k: $k = H_1(PK_{TS}, ID_s, t)$
  - Pick a random glue value: The Signer picks an initialization(glue) value $\hat{v}$ uniformly at random from $(0, 1)^b$.
  - Pick random value for the *Time Server*: The Signer picks a random value $x_{\hat{TS}}$ uniformly from $(0, 1)^b$ and computes $y_{\hat{TS}} = g_{TS}(x_{\hat{TS}})$
  - Solve for $\hat{y_s}$: The Signer solves the following ring equation for $\hat{y_s}$: $C_{k,v}(\hat{y_s}, y_{\hat{TS}}) = \hat{v}$
  - Invert the trapdoor permutation: The Signer uses his trapdoor information in order to invert g_s on $\hat{y_s}$ to obtain $\hat{x_s}$: $\hat{x_s} = g_s^{-1}(\hat{y_s})$
  - reture a pre-mature signature: The ring signature part generated by the Signer is defined to be the 5-tuple: $(PK_s, PK_{TS}; \hat{v}; \hat{x_s}, x_{\hat{TS}})$ and the pre-mature signature is $\hat{\sigma_t} = (\sigma'_t, PK_s, PK_{TS}; \hat{v}; \hat{x_s}, x_{\hat{TS}})$
- $Verify$: For a given hatched signature σ_t (or pre-hatched signature $\hat{\sigma_t}$). The verifier first check the ring signature part:

– Apply the trapdoor permutation: the verifier computes $y_{TS} = g_{TS}(x_{TS})$ and $y_s = g_s(x_s)$.
– Obtain: the verifier hashes the public key of the *Time Server* and the Signer's ID to compute the symmetric encryption key k: $k = H_1(PK_{TS}, ID_s)$
– Verify the ring equation: the verifier checks whether the following equation is satisfied: $C_{k,v}(y_s, y_{TS}) = v$.

If the ring equation is satisfied, then he verifies that σ'_t is a valid time capsule signature on h by running $Ver_{PK_s}(h, \sigma'_t)$.

The *correctness* property of the scheme are obvious from the properties of the basis digital signature scheme and the ring signature scheme. We now analyze its security.

Lemma 1. The time capsule signature scheme presented above is unforgery if the underlying ordinary signature scheme is UF-CMA secure and ring signature scheme is unforgery. Concretely, we obtain the following bound:

$$\mathbf{Succ}^{\text{UF-TCS-CMA}}_{\mathcal{TCS}}(t_{\text{TCS}}, q_{\text{TCS}}) \leq \mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS}}(t_{\text{DS}}, q_{\text{DS}}) + \mathbf{Succ}^{\text{UF}}_{\mathcal{RS}}(t_{\text{RS}}, q_{\text{RS}}) \tag{2}$$

where $t_{\text{DS}} + t_{\text{RS}} = t_{\text{TCS}} + T_{\mathcal{O}_{\mathcal{S}_{\text{DS}}}} + T_{\mathcal{O}_{\mathcal{S}_{\text{RS}}}}$, and $q_{\text{DS}} + q_{\text{RS}} = q_{\text{TCS}}$. Here, $T_{\mathcal{O}_{\mathcal{S}_{\text{DS}}}}$ and $T_{\mathcal{O}_{\mathcal{S}_{\text{RS}}}}$ denote the running time of the signature generation algorithm simulated by oracle $\mathcal{O}_{\mathcal{S}}$ in the attack games of $\mathcal{DS}$ scheme and $\mathcal{RS}$ scheme respectively.

Proof: Let B denote an attacker that defeats the *unforgeability of the immature signature* in the $\mathcal{TCS}$ scheme. Let A denote an attacker that defeats the UF-CMA security of the $\mathcal{DS}$ scheme.

We show how the view of B in the real attack game of UF-imTCS (Definition 7), which we denote by $\mathbf{G_0}$, can be simulated to obtain a new game $\mathbf{G_1}$ which is related to the ability of the attacker A to defeat the UF-CMA security of the $\mathcal{DS}$ scheme.

Game $\mathbf{G_0}$: As mentioned, this game is identical to the real attack game $\mathbf{Exp}^{\text{UF-imTCS}}_{\mathcal{TCS},\,\mathsf{B}}$ described in Definition 7. We denote by E_0 the event that B outputs a valid message and immature time capsule signature pair as a forgery. We use a similar notation E_1 for Game $\mathbf{G_1}$. Since Game $\mathbf{G_0}$ is the same as the real attack game, we have

$$Pr[E_0] = \mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS},\,\mathsf{B}}$$

Game $\mathbf{G_1}$: First, we replace B's common parameter by A's common parameter, denoted as *cp1*. Then we do the following. Whenever B issues a time capsule signature(an immature signature) generation query m, we intercept it and forward $(m, \cdot)$ as A's signature generation query to A's Challenger to get a corresponding signature σ. We then send this σ to B. If B outputs $(m, \tilde{t}, \tilde{\sigma}_t)$, we intercept it and return it as A's forgery. Note from the simulation that B's view in the real attack game is identical to it's view in Game $\mathbf{G_1}$. Hence we have

$$Pr[E_1] \geq Pr[E_0]$$

By definition of $\Pr[E_1]$ and $\Pr[E_0]$, we obtain

$$\mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\,\mathsf{A}}(k) \geq \mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS},\,\mathsf{B}}(k)$$

Considering the running time and the number of queries, we obtain the following bound

$$\mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS},\ \mathsf{B}}(t_{\text{imTCS}}, q_{\text{imTCS}}) \leq \mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}(t_{\text{DS}}, q_{\text{DS}}) \tag{3}$$

where $t_{\text{DS}} = t_{\text{imTCS}} + T_{\mathcal{O}_{\mathcal{S}_{\text{DS}}}}$, and $q_{\text{DS}} = q_{\text{imTCS}}$. Here, $T_{\mathcal{O}_{\mathcal{S}_{\text{DS}}}}$ denotes the running time of the signature generation algorithm simulating by oracle $\mathcal{O}_{\mathcal{S}}$ in the attack game of $\mathcal{DS}$ scheme.

Then let D denote an attacker that defeats the *Un-Prehatchable by dishonest users* in the $\mathcal{TCS}$ scheme. And C denote an attacker that defeats the *unforgeability* of the $\mathcal{RS}$ scheme.

We show how the view of D in the real attack game of UF-UPH (Definition 7), which we denote by $\mathbf{G_2}$, can be simulated to obtain another new game $\mathbf{G_3}$ related to the ability of the attacker C to defeat the *unforgeability* of the $\mathcal{RS}$ scheme.

Game $\mathbf{G_2}$: As mentioned, this game is identical to the real attack game $\mathbf{Exp}^{\text{UPH}}_{\mathcal{TCS},\ \mathsf{D}}$ described in Definition 7. We denote by E_2 the event that D outputs a valid (m, t, σ_t) triple as a forgery from the given time capsule signature triple (m, t, σ'_t). We use a similar notation E_3 for Game $\mathbf{G_3}$. Since Game $\mathbf{G_2}$ is the same as the real attack game and we have

$$Pr[E_2] = \mathbf{Succ}^{\text{UPH}}_{\mathcal{TCS},\ \mathsf{D}}$$

Game $\mathbf{G_3}$: We first replace D's common parameter by C's common parameter, denoted as *cp2*. Then we do the following. Whenever D issues a time release query t or a PreHatch query (m, t, σ'_t), we intercept it and forward (t, σ'_t) as C's ring signature generation query to C's Challenger to get a corresponding ring signature. Combine this ring signature with σ'_t and get a signature σ_t. We then send this σ_t to D. If D outputs $(m, t, \tilde{\sigma_t})$, we intercept it and parse it to get the ring signature part $\sigma_{\tilde{RS}}$. Then return it as C's forgery. Note from the simulation that D's view in the real attack game is identical to it's view in Game $\mathbf{G_3}$. Hence we have

$$Pr[E_3] \geq Pr[E_2]$$

By definition of $\Pr[E_3]$ and $\Pr[E_2]$, we obtain

$$\mathbf{Succ}^{\text{UF}}_{\mathcal{RS},\ \mathsf{C}}(k) \geq \mathbf{Succ}^{\text{UPH}}_{\mathcal{TCS},\ \mathsf{D}}(k)$$

Considering the running time and the number of queries, we obtain the following bound

$$\mathbf{Succ}^{\text{UPH}}_{\mathcal{TCS},\ \mathsf{D}}(t_{\text{UPH}}, q_{\text{UPH}}) \leq \mathbf{Succ}^{\text{UF}}_{\mathcal{RS},\ \mathsf{C}}(t_{\text{RS}}, q_{\text{RS}}) \tag{4}$$

where $t_{\text{RS}} = t_{\text{UPH}} + T_{\mathcal{O}_{\mathcal{S}_{\text{RS}}}}$, and $q_{\text{RS}} = q_{\text{UPH}}$. Here, $T_{\mathcal{O}_{\mathcal{S}_{\text{RS}}}}$ denotes the running time of the ring signature generation algorithm simulating by oracle $\mathcal{O}_{\mathcal{S}}$ in the ring signature attack game.

Let F be an attacker, which possesses the attack capability of both B and D, defeats the UF-TCS-CMA security of the $\mathcal{TCS}$ scheme.

As we see Equation (1) in Definition 8, $\mathbf{Succ}^{\text{UF-TCS-CMA}}_{\mathcal{TCS}} \leq \mathbf{Succ}^{\text{UF-imTCS}}_{\mathcal{TCS}} + \mathbf{Succ}^{\text{UPH-TCS}}_{\mathcal{TCS}}$. Combine with Equation (2) and (3), we obtain

$$\mathbf{Succ}^{\text{UF-TCS-CMA}}_{\mathcal{TCS},\ \mathsf{F}}(k) \leq \mathbf{Succ}^{\text{UF-CMA}}_{\mathcal{DS},\ \mathsf{A}}(k) + \mathbf{Succ}^{\text{UF}}_{\mathcal{RS},\ \mathsf{C}}(k)$$

Considering the running time and the number of queries, we can obtain the bound. So Equation (2) in the theorem statement holds.

Lemma 2. The time capsule signature scheme presented above is *indistinguishable*(ambiguous) if the underlying bipartite ring signature scheme $\mathcal{RS}_{bi}$ holds the *anonymity*. Concretely, we obtain the following equation:

$$\mathbf{Succ}^{\text{ANON}}_{\mathcal{RS}_{bi},\,\mathsf{A}}(k) \geq \mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}(k) \tag{5}$$

Proof: Let D denote an attacker that defeats the *indistinguishability* in the $\mathcal{TCS}$ scheme and let A denote an attacker that defeats the *anonymity* of the bipartite $\mathcal{RS}_{bi}$ scheme.

We will show how the view of D in the real attack game of IND-TCS (Definition 7), which we denote by $\mathbf{G_0}$, can be simulated to obtain a new game $\mathbf{G_1}$ which is related to the ability of the attacker A to defeat the *anonymity* of the bipartite $\mathcal{RS}_{bi}$ scheme.

Game $\mathbf{G_0}$: this game is identical to the real attack game $\mathbf{Exp}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}$ described in Definition 7. We denote by E_0 the event that D's output $\tilde{d}$ equals to d.We use a similar notation E_1 for Game $\mathbf{G_1}$. Since Game $\mathbf{G_0}$ is the same as the real attack game, we have

$$|Pr[E_0] - 1/2| = \mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}$$

Game $\mathbf{G_1}$: As we know, in this game, adversary A will be provided an oracle for a biparitie ring signature algorithm and the goal of adversary A is to successfully guess b. To do so, A first output a pair of indices (i_0, i_1) where PK_{i_0} and PK_{i_1} are the public keys of the two parties in the bipartite ring, respectively. Then A will run distinguisher D as a subroutine. We replace D's common parameters by A's common parameter, denoted as cp. Then we do the following. Whenever D issues a $Hatch$ query σ'_t to $\mathcal{O}_{Hatch}$(notice that here the oracle $\mathcal{O}_{Hatch}$ will run oracle $\mathcal{O}_{TR}$ first interiorly), we intercept it and forward (σ'_t, t) as A's signature generation query to A's challenger and get a ring signature σ_{RS_0} corresponding to i_0. We then concatenate σ'_t and σ_{RS_0} as σ_t and send σ_t to D. When D issue a $PreHatch$ query of σ'_t to $\mathcal{O}_{PreHatch}$, we intercept it and forward (σ'_t, t) as A's signature generation query to A's challenger and get a ring signature σ_{RS_1} corresponding to i_1. We then concatenate σ'_t and σ_{RS_1} as σ_t and send σ_t to D. If D output $\tilde{d}$, we intercept it and let $\tilde{b} = \tilde{d}$ as A's guess result.

From the simulation that D's view in the real attack game is identical to it's view in Game $\mathbf{G_1}$. Hence we have

$$|Pr[E_1] - 1/2| \geq |Pr[E_0] - 1/2|$$

By definition of $\Pr[E_1]$ and $\Pr[E_0]$, we obtain Equation (5)

$$\mathbf{Succ}^{\text{ANON}}_{\mathcal{RS}_{bi},\,\mathsf{A}}(k) \geq \mathbf{Succ}^{\text{IND-TCS}}_{\mathcal{TCS},\,\mathsf{D}}(k)$$

So we can conclude that our time capsule signature scheme is *indistinguishable* if the underlying bipartite ring signature scheme holds the *anonymity*.

Lemma 3. The cheating behavior of dishonest signer do not exist in our construction.

Proof: Let's recall the experiment $\mathbf{Exp}^{\text{NCH-TCS}}_{\mathcal{TCS},\,\mathsf{C}}$ given in Definition 7. The adversary C(dishonest signer) wins the game means that he successfully produced a time capsule signature σ'_t which looks good to the verifier but can not be hatched into a full signature of C by any recipient using the information released by *Time Sever*.

However, in our construction of $\mathcal{TCS}$ scheme, the security aspect *non-cheating of the signer* follows unconditionally. If a time capsule signature σ'_t satisfies that $TVerify(m, \sigma'_t, PK, TPK, t) = 1$. When *Time Server* releases Z_t, any party can obtain a signature σ_t related to σ'_t. By the *correctness* property of ring signature, the ring signature part of σ_t always holds the ring equation. Therefore, the hatched signature σ_t can passes the verification algorithm $Verify$.

Theorem 1. Our generic construction of time capsule signature from bipartite ring signature is secure.

Proof: According to the above three Lemmas and analysis, it is obvious that those four security aspects mentioned in Definition 7 can be achieved. Therefore our construction of time capsule signature is secure.

4 A Concrete Time Capsule Signature Scheme

4.1 Description of Our Proposed Scheme

Let $\mathsf{Gen}(1^k)$ be the system parameter generation algorithm and $\mathsf{Set}(1^k)$ be the usual RSA key generation algorithm. In our scheme, each party A_i has an RSA public key $P_i = (N_i, e_i)$ which specifies the trapdoor one-way permutation f_i of Z_{N_i}:

$$f_i(x) = x^{e_i} \ (mod\ N_i)$$

Assume that only A_i knows how to compute the inverse permutation f_i^{-1} efficiently, using trapdoor information $d_i = e_i^{-1} \ (mod\ \phi(N_i))$. This is the original Diffie Hellman model[19] for public-key cryptography.

For the trap door permutation f over Z_N, we define a extended trapdoor permutation g over $\{0,1\}^j$ in the following way. For any j-bit input m define nonegative integers q and r so that $m = qN + r$, $0 < r < N$. Then

$$g(m) = \begin{cases} qN + f(r), & \text{if } (q+1)N \le 2^j \\ m, & \text{o.w.} \end{cases}$$

The function g is clearly a permutation over $\{0,1\}^j$, and it is a one-way trapdoor permutation since only someone who know how to invert f can invert g efficiently on more than a negligible fraction of the possible inputs.

Let E be a publicly defined symmetric encryption algorithm such that for any key k of length l, the function E_k is a permutation over j-bit strings.

Define a family of keyed combine functions:

$$C_{k,v}(y_0, y_1) = E_k(y_1 \oplus E_k(y_0 \oplus v)) = v$$

It takes as input a key k, an initialization value v, and arbitrary values y_0, y_1 in $\{0,1\}^j$. Each such combine function uses E_k as a sub-procedure, and produces as output a value z in $\{0,1\}^j$. We can see given any fixed values for k and v, $C_{k,v}$ has the following properties:

1. For any fixed value of y_b, $b \in \{0,1\}$, the function $C_{k,v}$ is a one-to-one mapping from $y_{\bar{b}}$ to the output z.
2. Given j-bit values z and y_b, it is possible to efficiently find a j-bit value for $y_{\bar{b}}$ such that $C_{k,v}(y_b, y_{\bar{b}}) = z$.
3. Give k, v, z, it is infeasible for an adversary to solve the equation

$$C_{k,v}(g_0(x_0), g_1(x_1)) = z$$

for x_0, x_1 if the adversary can't invert any of the trapdoor functions g_0, g_1.

We now describe the full time capsule signature scheme by specifying an 8-tuple of PPT algorithms (*Setup*^{TS}, *Setup*^{User}, *TSign*, *TVerify*, *TRelease*, *Hatch*, *PreHatch*, *Verify*), corresponding to eight phases respectively. To make clear, we list these eight algorithms in the table.

- *Setup Phase of Time Server*: In algorithm *Setup*$^{TS}(1^k)$, $k = k_0 + k_1 = |N_{TS}|$ with 2^{-k_0} and 2^{-k_1} being negligible quantities; H, H_0, H_3 are hash functions satisfying

$$H_0 : \{0,1\}^* \mapsto \{0,1\}^{k_1},\ H : \{0,1\}^{k_1} \mapsto \{0,1\}^{k-k_1-1},\ H_3 : \{0,1\}^* \mapsto \{0,1\}^l$$

  The output bit string from H is split into two sub-bit-strings, one is denoted by H_1 and hash the first k_0 bits, the other is denoted by H_2 and hash the remaining $k - k_1 - k_0 - 1$ bits. l is the key length of symmetric encryption algorithm E_k.
- *Setup Phase of User:* Each user U_i with identity ID_i in the system runs the algorithm *Setup*$^{User}(1^k)$ during this phase.
- *Time capsule signature generation:* To make a future signature, the signer s runs algorithm *TSign*(m, d_s, N_s).
- *Time capsule signature verification:* To check the time capsule signature, the algorithm *TVerify*(m, σ'_t, e_s, N_s) is run by verifier.
- *The time release phase:* At the beginning of each time period t, the *Time Server* runs algorithm *TRelease*$(t, ID_s, ID_{TS}, PK_s, PK_{TS})$ computes and publishes the knowledge for hatching a valid time capsule signature of signer s.
- *Open a mature valid time capsule signature:* The recipient runs algorithm *Hatch*(σ'_t, Z_t).
- *Open an immature valid time capsule signature:* Before time t, the signer can run algorithm *PreHatch*$(t, ID_s, ID_{TS}, PK_s, PK_{TS})$ to open a valid time capsule signature which is not mature yet.
- *Verify a hatched or prehatched signature:* The verifier checks a mature signature by running algorithm *Verify*$(m, \sigma_t, ID_s, ID_{TS}, PK_s, PK_{TS}, t)$.

Algorithm $Setup^{TS}(1^k)$ $(N_{TS},e_{TS},d_{TS},H,H_0,H_3,k_0,k_1)\leftarrow\mathsf{Gen}(1^k)$ $PK_{TS} \leftarrow (N_{TS}, e_{TS})$ $SK_{TS} \leftarrow d_{TS}$ Output $(PK_{TS}, H, H_0, H_3, k_0, k_1)$	Algorithm $Setup^{User}(1^k)$ $(N_i, e_i, d_i) \leftarrow \mathsf{Set}(1^k)$ $PK_i \leftarrow (N_i, e_i)$ $SK_i \leftarrow d_i$ Return PK_i
Algorithm $TSign(m, d_s, N_s)$ $r \xleftarrow{R} \{0,1\}^{k_0}$ $w \leftarrow H_0(m\|ID_s\|t\|r)$ $r^* \leftarrow H_1(w) \oplus r$ $y \leftarrow 0\|w\|r^*\|H_2(w)$ $\sigma'_t \leftarrow y^{d_s} \ (mod\ N_s)$ Return σ'_t	Algorithm $TVerify(m, \sigma'_t, e_s, N_s)$ $y \leftarrow (\sigma'_t)^{e_s} \ (mod\ N_s)$ Parse y as $b\|w\|r^*\|H_2(w)$ $r \leftarrow r^* \oplus H_1(w)$ If $(H_0(m\|ID_s\|t\|r)=w \wedge H_2(w)=r \wedge b=0)$ Return 1 else Return 0
Algorithm $TRelease(t,ID_s,ID_{TS},PK_s,PK_{TS})$ $k \leftarrow H_3(ID_{TS}, ID_s, t)$ $v \xleftarrow{R} \{0,1\}^j$ $x_s \xleftarrow{R} \{0,1\}^j$ $y_s \leftarrow g_s(x_s)$ solve equation $C_{k,v}(y_s, y_{TS}) = v$ for y_{TS} $x_{TS} \leftarrow g_{TS}^{-1}(y_{TS})$ $Z_t \leftarrow (PK_s, PK_{TS}, v, x_s, x_{TS})$ Return Z_t	Algorithm $PreHatch(t,ID_s,ID_{TS},PK_s,PK_{TS})$ $k \leftarrow H_3(ID_{TS}, ID_s, t)$ $v \xleftarrow{R} \{0,1\}^j$ $x_{TS} \xleftarrow{R} \{0,1\}^j$ $y_{TS} \leftarrow g_{TS}(x_{TS})$ solve equation $C_{k,v}(y_s, y_{TS}) = v$ for y_s $x_s \leftarrow g_s^{-1}(y_s)$ $\sigma_t \leftarrow (\sigma'_t, PK_s, PK_{TS}, v, x_s, x_{TS})$ Return σ_t
Algorithm $Hatch(\sigma'_t, Z_t)$ $\sigma_t \leftarrow (\sigma'_t, PK_s, PK_{TS}, v, x_s, x_{TS})$ $= (\sigma'_t; Z_t)$ Return σ_t	Algorithm $Verify(m,\sigma_t,ID_s,ID_{TS},PK_s,PK_{TS},t)$ $y_{TS} \leftarrow g_{TS}(x_{TS})$ $y_s \leftarrow g_s(x_s)$ $k \leftarrow H_3(ID_{TS}, ID_s, t)$ If $C_{k,v}(y_s,y_{TS})=v$ Run $TVerify(m, \sigma'_t, PK_s)$ else Return 0

4.2 Security Analysis

Theorem 2. This concrete time capsule signature scheme given above is a secure time capsule signature scheme.

Proof: As the underlying original digital signature scheme used in the construction are UF-CMA secure and the bipartite ring signature scheme we utilize is unforgeable and anonymous. According to the security analysis and theorems given in the former section, our concrete construction of time capsule signature scheme is secure.

5 Conclusion

Time capsule signature is a useful paradigm, applicable across cryptosystems and cryptographic protocols. In this paper we first recall the concept of time capsule signature and develop its security notions. Then we propose a new construction

of time capsule signature which is proven secure in the random oracle model from bipartite ring signature. This construction removes the deficiencies in the existing scheme based on ID-THIR. Secure and concrete time capsule signature scheme can be easily achieved via our novel way of construction.

Acknowledgment

The author would like to thank Hongsheng Zhou and the anonymous reviewers for their valuable comments on the earlier drafts of the paper.

References

1. Y. Dodis and D. Yum. Time Capsule Signature, Financial Cryptography and Data Security Conference (FC), 2005. , LNCS 3570, pp.57-71, Springer-Verlag, 2005.
2. D. Boneh and M. Naor. Timed commitments. Advances in Cryptology-CRYPTO 2000, LNCS 1880, pp.236-254. Springer-Verlag, 2000.
3. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. In K. Nyberg, editor, Advances in CryptologyEUROCRYPT 1998, LNCS 1403, pp.591-606. Springer-Verlag, 1998.
4. N. Asokan, V. Shoup, and M. Waidner. Optimistic fair exchange of digital signatures. IEEE Journal on Selected Areas in Communication, 18(4), pp.593-610, 2000.
5. Y. Dodis and L.Reyzin. Breaking and repairing optimistic fair exchange from PODC 2003. In Proceedings of the ACM workshop on Digital Rights Management 2003, pp.47-54, 2003.
6. C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In E. Brickell , editor, Advances in CryptologyCRYPTO 1992, LNCS740, Springer-Verlag, pp.139-147,2004.
7. M. Bellare and S. Goldwasser. Encapsulated key escrow. MIT Laborator for Computer Science Technical Report 688, Apr. 1996.
8. M. Bellare and S. Goldwasser. Verifiable partial key escrow. In Proceedings of the 4th ACM Conference on Computer and Communications Security, pp.78-91. 1-4 Apr. 1997.
9. R. Rivest, A. Shamir, and D. Wagner. Time lock puzzles and and timed release cryptography. Technical report, MIT/LCS/TR-684.
10. J. Garay and M. Jakobsson. Timed release of standard digital signatures. In Financial Cryptography 2002, volume 2357 of Lecture Notes in Computer Science, pages 168-182. Springer-Verlag, 11C14 Mar. 2002.
11. J. Garay and C. Pomerance. Timed fair exchange of standard signatures. In Financial Cryptography 2003, volume 2742 of Lecture Notes in Computer Science, pages 190-207. Springer-Verlag, 27C30 Jan. 2003.
12. I. Osipkov, Y. Kim and J. Cheon. New approaches to timed-release cryptography IACR E-print Archive. Available from http://eprint.iacr.org/2004/231/, 2004.
13. D. Boneh, C. Gentry, B. Lynn, and H. Shacham. Aggregate and verifiably encrypted signatures from bilinear maps. In Advances in Cryptology-EUROCRYPT 2003, LNCS 2656, pp416-432. Springer-Verlag, 2003.
14. I. Blake and A. Chan. Scalable, server-passive, user-anonymous timed release public key encryption from bilinear pairing. IACR E-print Archive. Available from http://eprint.iacr.org/2004/211/, 2004.

15. D. Boneh and M. Franklin. Identity based encryption from the Weil pairing. In J. Kilian, editor, Advances in Cryptology-CRYPTO 2001, LNCS 2139, pp.213-229. Springer-Verlag, 2001.
16. Joseph K. Liu and Duncan S. Wong2, On the security models of (threshold) ring signature schemes, the 7th International Conference on Information Security and Cryptology (ICISC 2004), LNCS 3506, Springer-Verlag, 2005, 204 - 217.
17. R. L. Rivest, A. Shamir and Y. Tauman, How to Leak a Secret, Advances in Cryptology-Asiacrypt 2001, LNCS 2248, pp.552-565, Springer-Verlag, 2001.
18. Goldwasser, S. and Bellare, M. Lecture Notes on Cryptography. Summer course on cryptography, MIT, 1996-2001.
19. W. Diffie and M. E. Hellman. New directions in cryptography. IEEE Trans. Inform. Theory, IT-22:644-654, November 1976.
20. Adam Bender, Jonathan Katz and Ruggero Morselli, Ring Signatures: Stronger Definitions, and Constructions without Random Oracles, 3rd Theory of Cryptography Conference(TCC06), LNCS 3876, pp.60-79, Springer-Verlag, 2006
21. Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Maurer [Mau96], pages 399-416. Revised version appears in http://www-cse.ucsd.edu/users/mihir/papers/crypto-papers.html