Practical Time Capsule Signatures in the Standard Model from Bilinear Maps

Benoît Libert* and Jean-Jacques Quisquater

UCL, Microelectronics Laboratory, Crypto Group
Place du Levant, 3, B-1348, Louvain-La-Neuve, Belgium.
{benoit.libert,jean-jacques.quisquater}@uclouvain.be

Abstract. At FC'05, Dodis and Yum introduced a new cryptographic tool called *time capsule signature* (TCS) which allows signers to generate "future signatures" that only become valid from a specific future time t (chosen at signature generation) when a trusted entity (called *Time Server*) discloses some trapdoor information for period t. In addition, time capsule signatures endow signers with the ability to make their signatures valid before the pre-determined time t. Full signatures that were completed by their original issuer should be indistinguishable from those that automatically became valid after the release of the time-specific trapdoor. Time capsule signatures were showed to be generically constructible from another primitive called *identity-based trapdoor hard-to-invert relation* (ID-THIR). The only known instantiations of the latter either rely on the idealized random oracle model or are too inefficient for real-world applications. In this paper, we devise the first efficient ID-THIR (and thus TCS) construction which is secure in the standard model (i.e. without the random oracle heuristic).

Keywords. time capsule signatures, standard model, bilinear maps.

1 Introduction

In 2005, Dodis and Yum introduced the concept of time capsule signatures [17]. Such a primitive allows signers to generate signatures that only become valid from a future moment t when a trusted party (called Time Server) discloses a trapdoor information associated with period t. This is accomplished in such a way that:

- Anyone can directly ascertain that a "future signature" will indeed become effective at time t.
- In a "pre-hatching operation", the legal signer can decide to make her future signature valid at any time before the pre-determined moment t.
- A signature that was not opened by the signer automatically becomes valid (which is called "hatching" as opposed to "pre-hatching") at time t when the Time Server publishes the relevant trapdoor information Z_t allowing signature holders to complete future signatures generated for that period.

* This author acknowledges the DGTRE's First Europe Program of the Walloon Region in Belgium for his financial support.

– The Time Server does not have to interact with any user at any time or know anything about the PKI employed by signers.

Regardless of whether a signature was previously opened by the signer or if it was automatically completed after the release of the trapdoor Z_t at time t, no one can tell how it became valid: "pre-hatched" signatures should be indistinguishable from "hatched" signatures.

Similarly to time release primitives described in [5, 15, 31], time capsule signatures (TCS) follow the server-based approach which allows preparing messages for a definite future and departs from "time-lock puzzle" methods addressing related problems [33, 2, 10, 30, 20, 21]. They imply a minimal assumption on the Time Server that only has to publish some piece of information at the beginning of each time period and never has to contact users.

In [17], Dodis and Yum gave proper security definitions for TCS schemes and showed how to generically construct them using newly defined primitives called *identity-based trapdoor hard-to-invert relations* (ID-THIRs). They also described a generic construction of ID-THIR which yields very efficient implementations in the random oracle model [4] but is much less efficient in the standard model. These results proved the existence of time capsule signatures in the random oracle model assuming the availability of one-way functions and their existence in the standard model if trapdoor one-way permutations exist.

Our contribution. The generic construction of ID-THIR given in [17] relies on non-interactive witness-indistinguishable [18] proofs of knowledge. Before the recent advances of Groth, Ostrovsky and Sahai [26, 27] in NIZK and witness indistinguishable proofs, the best known methods [34] for constructing such proofs in the standard model were very inefficient. Those dramatic improvements were adapted [28] so as to provide constant-size - though impractical - group signatures in the standard model. They could be applied to the present context as well, but resulting implementations would remain too inefficient for practical use. To date, the only practical examples of TCS schemes resort to the random oracle methodology [4] which is known [13] to *only* provide heuristic arguments.

The achievement of this paper is to describe a very simple and efficient identity-based trapdoor hard-to-invert relation which is not generic but is secure in the standard model. It utilizes the Waters signature [36] which is known to be secure in the standard model assuming the hardness of the Diffie-Hellman problem in groups equipped with bilinear maps. More precisely, our ID-THIR turns out to be somehow related to identity-based [35] extensions [12, 32] of Waters signatures. This is not very surprising since the generic ID-THIR of [17] was already making use of proofs of knowledge of signatures (which are nothing but identity-based signatures). The technical difficulty was here to avoid witness indistinguishable proofs. To do so, our implementation takes advantage of tricks which date back to [7] and that were used to prove the security of the signature in [36]. Thanks to the generic transformation of [17], our ID-THIR gives rise to the first practical time capsule signature scheme which is secure in the standard

model (under a well-studied computational assumption).

Organization. In the forthcoming sections, we first recall functional definitions and security models for identity-based trapdoor hard-to-invert relations and time capsule signatures. Section 3 then describes our practical construction of ID-THIR. Its possible optimizations are discussed in section 4 and the resulting concrete TCS scheme is analyzed in section 5.

2 Preliminaries

2.1 Identity-Based Trapdoor Hard-to-Invert Relations

A binary relation R is a subset of $\{0,1\}^* \times \{0,1\}^*$ and the language $\mathcal{L}_R$ is the set of elements α for which there exist β such that $(\alpha, \beta) \in R$. The relation R must be completely specified by a short description D_R. Besides, for all pairs $(\alpha, \beta) \in R$, the length $|\beta|$ of β has to be bounded by a polynomial in $|\alpha|$. Lastly, it should be easy to decide whether a given α lies in $\mathcal{L}_R$.

Definition 1. *An identity-based trapdoor hard-to-invert relation* (ID-THIR) *is a family of binary relations $\mathcal{R} = \{R_{id} | id \in I_{\mathcal{R}}\}$, where $I_{\mathcal{R}}$ is a finite set of indices, that are all trapdoor hard-to-invert relations. Namely, for each $id \in I_{\mathcal{R}}$, sampling a lock/proof pair $(c, d) \in R_{id}$ is easy but finding a proof for a given lock is hard without knowing the specific trapdoor td_{id}. A master trapdoor $\mathsf{mtd}_{\mathcal{R}}$ allows extracting a trapdoor td_{id} for each relation $R_{id} \in \mathcal{R}$. An* ID-THIR *is entirely specified by a 5-uple of algorithms* (Gen, Sample, Check, Extract, Invert) *such that:*

- Gen*: given a security parameter k, this algorithm generates $\mathcal{R} = \{R_{id} | id \in I_{\mathcal{R}}\}$ and returns its description $D_{\mathcal{R}}$ and its master trapdoor $\mathsf{mtd}_{\mathcal{R}}$.*
- Sample*: takes as input $(D_{\mathcal{R}}, id)$ and returns a randomly sampled lock/proof pair $(c, d) \in R_{id}$.*
- Check*: verifies the validity of a lock/proof pair (c, d). It returns 1 (accept) if $(c, d) \in R_{id}$ and 0 (reject) otherwise.*
- Extract*: is used to extract the trapdoor of each relation. Given $id \in I_{\mathcal{R}}$ and the master trapdoor $\mathsf{mtd}_{\mathcal{R}}$, it returns the trapdoor td_{id} for the relation R_{id}.*
- Invert*: allows finding a proof d for a given lock $c \in \mathcal{L}_{R_{id}}$ using the trapdoor td_{id}. If $c \in \mathcal{L}_{R_{id}}$, $\mathsf{Invert}_{\mathsf{td}_{id}}(c)$ outputs a proof d such that $(c, d) \in R_{id}$.*

Let $(c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id)$ and $\tilde{d} \leftarrow \mathsf{Invert}_{\mathsf{td}_{id}}(c)$. The correctness property imposes that $\mathsf{Check}_{D_{\mathcal{R}}, id}(c, d) = \mathsf{Check}_{D_{\mathcal{R}}, id}(c, \tilde{d}) = 1$. The ambiguity is the computational indistinguishability of (c, d) and $(c, \tilde{d})$ even knowing $\mathsf{mtd}_{\mathcal{R}}$. Besides, an ID-THIR *is said one-way if the following probability is negligible for any PPT algorithm $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$:*

$$\Pr[\mathsf{Check}_{D_{\mathcal{R}}, id^\star}(c, \hat{d}) = 1 \ \wedge \ id^\star \notin Query(\mathcal{A}, O_{\mathsf{Extract}}) \mid (D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}}) \leftarrow \mathsf{Gen}(k); \\ (id^\star, st) \leftarrow \mathcal{A}_1^{O_{\mathsf{Extract}}}(D_{\mathcal{R}}); \ (c, d) \leftarrow \mathsf{Sample}_{D_{\mathcal{R}}}(id^\star); \ \hat{d} \leftarrow \mathcal{A}_2^{O_{\mathsf{Extract}}}(D_{\mathcal{R}}, c, st)]$$

where O_{Extract} *is an oracle simulating the trapdoor extraction algorithm* Extract, $Query(\mathcal{A}, O_{\mathsf{Extract}})$ *is the set of queries made by* $\mathcal{A}$ *to the latter oracle and* st *stands for the state information passed by* $\mathcal{A}_1$ *to* $\mathcal{A}_2$*. The soundness property states that the following property is negligible for any algorithm* $\mathcal{B}$*:*

$$\Pr[\mathsf{Check}_{D_{\mathcal{R},id^\star}}(c,\tilde{d}) = 0 \ \wedge \ R_{id^\star} \in \mathcal{R} \ \wedge \ c \in \mathcal{L}_{R_{id^\star}} \mid (D_{\mathcal{R}}, \mathsf{mtd}_{\mathcal{R}}) \leftarrow \mathsf{Gen}(k); \\ (c, id^\star) \leftarrow \mathcal{B}(D_{\mathcal{R}}); \ \mathsf{td}_{id^\star} \leftarrow \mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(id^\star); \ \tilde{d} \leftarrow \mathsf{Invert}_{\mathsf{td}_{id^\star}}(c)]$$

An ID-THIR *is said secure if it meets the above four requirements.*

Intuitively, the one-wayness property captures that it should be computationally infeasible to open a given lock without the trapdoor of the corresponding relation even after having seen trapdoors for polynomially-many other relations. The soundness is the impossibility of coming up with a lock (for some relation) that cannot be opened into a valid lock/proof pair using the relevant trapdoor.

Dodis and Yum showed in [17] that an ID-THIR exists in the random oracle model if a one-way function exists. Their construction relies on the Fiat-Shamir heuristic [19] and non-interactive witness indistinguishable [18] proofs of knowledge. Instead of a Fiat-Shamir like proof, their method can be implemented with non-interactive witness indistinguishable proofs of knowledge (with a common reference string) that do not involve random oracles. However, the best known technique [34] for constructing such proofs uses trapdoor one-way permutations and is very inefficient. Therefore the existence of identity-based trapdoor hard-to-invert relations in the standard model, which requires the existence of trapdoor one-way permutations [17], is currently mainly of theoretical interest.

2.2 Time Capsule Signatures

Definition 2. *A time capsule signature (*TCS*) consists of a* 8*-uple of PPT algorithms* $(\mathsf{Setup}^{\mathsf{TS}}, \mathsf{Setup}^{\mathsf{User}}, \mathsf{TSig}, \mathsf{TVer}, \mathsf{TRelease}, \mathsf{Hatch}, \mathsf{PreHatch}, \mathsf{Ver})$.

- $\mathsf{Setup}^{\mathsf{TS}}$*: is an algorithm run by the Time Server. Given a security parameter* k*, it returns a public/private key pair* $(\mathsf{TPK}, \mathsf{TSK})$.
- $\mathsf{Setup}^{\mathsf{User}}$*: is run by each signer. Given a security parameter* k*, it returns a public/private key pair for the signer* $(\mathsf{PK}, \mathsf{SK})$.
- TSig*: is the time capsule signature generation algorithm. It takes as input* $(m, \mathsf{SK}, \mathsf{TPK}, t)$*, where* t *is the time from which the signature becomes valid. It produces a future signature* σ'_t.
- TVer*: is the time capsule signature verification algorithm. It takes as input a* 5*-uple* $(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t)$ *and returns either* 1 *(accept) or* 0 *(reject).*
- TRelease*: is the time release algorithm run by the Time Server. At the beginning of period* t*, it uses* TSK *to compute and publish* $Z_t = \mathsf{TRelease}(t, \mathsf{TSK})$*. Note that the Time Server never interacts with any user at any time.*
- Hatch*: is run by any party to open a valid time capsule signature that became mature. Given* $(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t)$ *and the time-specific trapdoor* Z_t *as inputs, it returns a hatched signature* σ_t.

$\mathsf{PreHatch}$*: is run by the signer to open a valid time capsule signature which is not mature yet. It takes as input* $(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t)$ *and the signer's private key* SK *as inputs and outputs a pre-hatched signature* σ_t.

Ver*: is used to verify hatched or pre-hatched signatures. Given* $(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t)$*, it returns* 1 *(accept) or* 0 *(reject).*

The correctness imposes that $\mathsf{TVer}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, t) = 1$ *and* $\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) = 1$ *if* $\sigma_t = \mathsf{Hatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, Z_t)$ *or* $\sigma_t = \mathsf{PreHatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{SK}, \mathsf{TPK})$*. Ambiguity requires the distribution of "hatched signatures"* $\sigma_t = \mathsf{Hatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{PK}, \mathsf{TPK}, Z_t)$ *to be computationally indistinguishable from that of "pre-hatched signatures"* $\sigma_t = \mathsf{PreHatch}(m, \mathsf{TSig}(m, \mathsf{SK}, \mathsf{TPK}, t), \mathsf{SK}, \mathsf{TPK})$ *even knowing* TSK.

As explained in [17], the security of time capsule signatures is defined in three aspects: security against the signer, the verifier and the Time Server. In the following notation O_{TSig} is an oracle simulating the time capsule signature generation algorithm TSig, O_{TR} denotes an oracle simulating the time release algorithm $\mathsf{TRelease}$ and O_{PreH} stands for the pre-hatching oracle emulating $\mathsf{PreHatch}$. Given (m, t) as input, O_{TSig} returns a time capsule signature σ'_t generated on behalf of the signer. Oracle O_{PreH} takes (m, t, σ'_t) as input and outputs the signer's pre-hatched signature σ_t.

Security against the signer. This definition means that the signer should be unable to produce a time capsule signature which looks good to the verifier but cannot be hatched into a full signature by the Time Server. More formally, any PPT adversary $\mathcal{A}$ should have negligible advantage in this experiment.

$$\begin{aligned}
\mathsf{Setup}^{\mathsf{TS}}(k) &\rightarrow (\mathsf{TPK}, \mathsf{TSK}) \\
(m, t, \sigma'_t, \mathsf{PK}) &\leftarrow \mathcal{A}^{O_{\mathsf{TR}}}(\mathsf{TPK}) \\
Z_t &\leftarrow \mathsf{TRelease}(t, \mathsf{TSK}) \\
\sigma_t &\leftarrow \mathsf{Hatch}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, Z_t) \\
Adv(\mathcal{A}) &= \Pr[\mathsf{TVer}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t) = 1 \ \wedge \ \mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) = 0]
\end{aligned}$$

Security against the verifier. Informally, the verifier must be unable to open a future signature without the help of the signer or the Time Server. We require any PPT adversary $\mathcal{B}$ to have negligible advantage in the next experiment.

$$\begin{aligned}
\mathsf{Setup}^{\mathsf{TS}}(k) &\rightarrow (\mathsf{TPK}, \mathsf{TSK}) \\
\mathsf{Setup}^{\mathsf{User}}(k) &\rightarrow (\mathsf{PK}, \mathsf{SK}) \\
(m, t, \sigma_t) &\leftarrow \mathcal{B}^{O_{\mathsf{TR}}, O_{\mathsf{TSig}}, O_{\mathsf{PreH}}}(\mathsf{TPK}, \mathsf{PK}) \\
Adv(\mathcal{A}) &= \Pr[\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) = 1 \ \wedge \ t \notin Query(\mathcal{B}, O_{\mathsf{TR}}) \\
&\qquad \wedge \ (m, t, .) \notin Query(\mathcal{B}, O_{\mathsf{PreH}})]
\end{aligned}$$

where $Query(\mathcal{B}, O_{\mathsf{TR}})$ is the set of queries made to the time release oracle O_{TR} and $Query(\mathcal{B}, O_{\mathsf{PreH}})$ denotes the set of *valid* queries to O_{PreH} (i.e. queries (m, t, σ'_t) for which $\mathsf{TVer}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t) = 1$).

Security against the Time Server. Obviously, the Time Server should not be able to produce a valid hatched or pre-hatched signature full signature on a message m without obtaining a time capsule signature on m from the signer. Any PPT adversary $\mathcal{C}$ must have negligible advantage in the following experiment.

$$\begin{aligned}
&\mathsf{Setup}^{\mathsf{TS}^*}(k) \rightarrow (\mathsf{TPK}, \mathsf{TSK}^*)\\
&\mathsf{Setup}^{\mathsf{User}}(k) \rightarrow (\mathsf{PK}, \mathsf{SK})\\
&\quad (m, t, \sigma_t) \leftarrow \mathcal{C}^{O_{\mathsf{TSig}}, O_{\mathsf{PreH}}}(\mathsf{PK}, \mathsf{TPK}, \mathsf{TPK}^*)\\
&\qquad Adv(\mathcal{C}) = \Pr[\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t) = 1 \ \wedge \ (m, .) \notin Query(\mathcal{C}, O_{\mathsf{TSig}})]
\end{aligned}$$

where $\mathsf{Setup}^{\mathsf{TS}^*}$ denotes a run of $\mathsf{Setup}^{\mathsf{TS}}$ by a dishonest Time Server, TSK^* is $\mathcal{C}$'s state after this malicious key generation and $Query(\mathcal{C}, O_{\mathsf{TSig}})$ stands for the set of queries to the time capsule signature oracle O_{TSig}.

2.3 Bilinear Maps

Groups $(\mathbb{G}, \mathbb{G}_T)$ of prime order p are called *bilinear map groups* if there is a mapping $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ with the following properties:

1. bilinearity: $e(g^a, h^b) = e(g, h)^{ab}$ for any $(g, h) \in \mathbb{G} \times \mathbb{G}$ and $a, b \in \mathbb{Z}$;
2. efficient computability for any input pair;
3. non-degeneracy: $e(g, h) \neq 1_{\mathbb{G}_T}$ whenever $g, h \neq 1_{\mathbb{G}}$.

The protocol that we have in mind relies on the intractability of the following well-studied problem in bilinear map groups.

Definition 3. *The* **Computational Diffie-Hellman Problem** *(CDH) in a group $\mathbb{G} = \langle g \rangle$ is to compute g^{ab} given (g^a, g^b). An algorithm (τ, ε)-breaks the CDH assumption if it solves a CDH instance with probability ε in time τ.*

2.4 The Waters Signature

We recall the description of the signature scheme of [36] which is existentially unforgeable in the standard model under the CDH assumption in bilinear map groups. In the description hereafter, messages are assumed to be encoded as bitstrings of length n. In practice however, a collision-resistant hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$ can be applied to sign longer messages.

$\mathsf{Keygen}(k, n)$: choose bilinear map groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^k$. Randomly pick $\alpha \xleftarrow{R} \mathbb{Z}_p^*$, as well as $g, g_2 \xleftarrow{R} \mathbb{G}$ and a vector $\overline{u} = (u', u_1, \ldots, u_n) \in \mathbb{G}^{n+1}$ of random group elements. The public key is $\mathsf{PK} = (n, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, \overline{u}, W)$ with $g_1 = g^\alpha$ and $W = e(g_1, g_2)$. The private key is $\mathsf{SK} = \alpha$.

$\mathsf{Sign}(\mathsf{m}, \alpha)$: parse m as $m_1 \ldots m_n$ with $m_i \in \{0,1\}$ for all $i \in \{1, \ldots, n\}$. A signature of m is produced by picking $r \xleftarrow{R} \mathbb{Z}_p^*$ and setting $\sigma = (\sigma_1, \sigma_2)$ with $\sigma_1 = g_2^\alpha \cdot (u' \cdot \prod_{i=1}^n u_i^{m_i})^r$ and $\sigma_2 = g^r$.

$\mathsf{Verify}(\mathsf{m}, \sigma, \mathsf{PK})$: a purported signature $\sigma = (\sigma_1, \sigma_2)$ on $\mathsf{m} = m_1 \ldots m_n$ is accepted if

$$e(\sigma_1, g) = W \cdot e\big(u' \cdot \prod_{i=1}^n u_i^{m_i}, \sigma_2\big).$$

3 An Efficient ID-THIR in the Standard Model

In this section, we present an identity-based trapdoor hard-to-invert relation based on the Waters signature. More precisely, it uses a 2-level hierarchical extension [22, 29] of the latter independently described in [12, 32] and which is intentionally made existentially (but not universally) forgeable here.

In a nutshell, sampling a random lock/proof pair for some relation R_{id} is done by generating a signature (d_1, d_2, d_3) on some artificial random "message" c in the name of the identity id. The sampling algorithm uses the technique of the simulator in the security proof of [36] to handle signing queries without knowing the private key. Generating a proof for any given lock c is easily achieved using the private key for the identity id.

$\mathsf{Gen}(k, n)$: this algorithm chooses bilinear map groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^k$ and a generator $g \in \mathbb{G}$. It computes $g_1 = g^\alpha$ for a random $\alpha \xleftarrow{R} \mathbb{Z}_p^*$. Next, it chooses $g_2 \xleftarrow{R} \mathbb{G}$, computes $W = e(g_1, g_2)$ and picks a random vector $\overline{u} = (u', u_1, \ldots, u_n) \xleftarrow{R} \mathbb{G}^{n+1}$ which allows defining a function $F : \{0,1\}^n \to \mathbb{G}$ as

$$F(id) = u' \cdot \prod_{j=1}^{n} u_j^{i_j}$$

where $id = i_1 \ldots i_n$ and $i_j \in \{0,1\}$ for all j. For an identity $id \in I_{\mathcal{R}} = \{0,1\}^n$, the relation $\mathcal{R}_{id}$ is defined as the set of pairs $\big(c, (d_1, d_2, d_3)\big) \in \mathbb{G} \times \mathbb{G}^3$ such that

$$e(d_1, g) = W \cdot e(F(id), d_2) \cdot e(c, d_3) \tag{1}$$

The master trapdoor is $\mathsf{mtd}_{\mathcal{R}} = g_2^\alpha$ and the family of relations $\mathcal{R}$ is entirely described by

$$D_{\mathcal{R}} = \{n, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, \overline{u}, W, \mathcal{R}_{id}, I_{\mathcal{R}}\}.$$

$\mathsf{Sample}(D_{\mathcal{R}}, id)$: to generate a random lock/proof pair $\big(c, (d_1, d_2, d_3)\big) \in \mathbb{G} \times \mathbb{G}^3$, this algorithm conducts the following steps.

1. Choose $j_1, j_2 \xleftarrow{R} \mathbb{Z}_p^*$ and compute $c = g_2^{j_1} g^{j_2}$.
2. Pick $r, s \xleftarrow{R} \mathbb{Z}_p^*$ and compute $d_1 = c^s \cdot g_1^{-j_2/j_1} \cdot F(id)^r$.
3. Set $d_2 = g^r$ and $d_3 = g^s \cdot g_1^{-1/j_1}$.

If we define $\tilde{s} = s - \frac{\alpha}{j_1}$, we observe that

$$d_1 = g_2^\alpha \cdot F(id)^r \cdot c^{\tilde{s}}, \quad d_2 = g^r, \quad d_3 = g^{\tilde{s}}. \tag{2}$$

$\mathsf{Check}_{D_{\mathcal{R}}, id}(c, d)$: parse d as (d_1, d_2, d_3). Return 1 if

$$e(d_1, g) = W \cdot e(F(id), d_2) \cdot e(c, d_3)$$

and 0 otherwise.

$\mathsf{Extract}_{\mathsf{mtd}_{\mathcal{R}}}(id)$: given $\mathsf{mdt}_{\mathcal{R}} = g_2^\alpha$, a trapdoor for $id \in \{0,1\}^n$ is extracted by randomly choosing $r \xleftarrow{R} \mathbb{Z}_p^*$ and returning $\mathsf{td}_{id} = (t_1, t_2) = (g_2^\alpha \cdot F(id)^r, g^r)$.

$\mathsf{Invert}_{\mathsf{td}_{id}}(c)$: parse td_{id} as (t_1, t_2). Choose random $r', s \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and return

$$(d_1, d_2, d_3) = (t_1 \cdot F(id)^{r'} \cdot c^s, t_2 \cdot g^{r'}, g^s) = (g_2^{\alpha} \cdot F(id)^{r''} \cdot c^s, g^{r''}, g^s).$$

with $r'' = r + r'$.

We now analyze the four security properties of the above scheme.

Correctness. It is clear that lock/proof pairs $(c, \tilde{d})$ where $\tilde{d} = \mathsf{Invert}_{\mathsf{td}_{id}}(c)$ satisfy equation (1) since $e(t_1, g) = W \cdot e(F(id), t_2)$ for all trapdoors $\mathsf{td}_{id} = (t_1, t_2)$ produced by $\mathsf{Extract}$. From (2), it follows that equation (1) is also satisfied by all pairs (c, d) produced by $\mathsf{Sample}(D_{\mathcal{R}}, id)$. Now, we check that elements $\big(c, (d_1, d_2, d_3)\big)$ generated by Sample are actually distributed according to (2). Indeed, since $c = g_2^{j_1} g^{j_2}$, we have

$$d_1 = c^s \cdot g_1^{-j_2/j_1} \cdot F(id)^r = c^{\tilde{s}} \cdot \big(g_2^{j_1} g^{j_2}\big)^{\alpha/j_1} \cdot g_1^{-j_2/j_1} \cdot F(id)^r = g_2^{\alpha} \cdot c^{\tilde{s}} \cdot F(id)^r$$
$$d_3 = g^s \cdot g_1^{-1/j_1} = g^{\tilde{s}}.$$

The sampling algorithm uses the strategy (borrowed from the Boneh-Boyen framework [7]) of the simulator answering signing queries in the proof of the Waters scheme [36].

Ambiguity. Sampled pairs $\big(c, (d_1, d_2, d_3)\big)$ clearly have exactly the same distribution as pairs $\big(c, (\tilde{d}_1, \tilde{d}_2, \tilde{d}_3)\big)$ when $(\tilde{d}_1, \tilde{d}_2, \tilde{d}_3) = \mathsf{Invert}_{\mathsf{td}_{id}}(c)$.

Soundness. It directly derives from the fact that any given $c \in \mathbb{G}$ can be "signed" using the trapdoor for the relation R_{id} (which is a private key for the identity id in [12, 32]).

One-wayness. The next theorem shows that our ID-THIR is one-way if Waters signatures are existentially unforgeable under chosen-message attacks [24].

Theorem 1. *An attacker breaking the one-wayness property of our* ID-THIR *in the sense of definition 1 implies a chosen-message attacker with the same advantage and running in comparable time for Waters signatures.*

Proof. Let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2)$ be an adversary with advantage ε against the one-wayness property. We construct a forger $\mathcal{F}$ using $\mathcal{A}$ to forge a signature using a challenger $\mathcal{CH}$ answering signing queries.

Algorithm $\mathcal{F}$ first obtains a public key $\mathsf{PK} = (n, \mathbb{G}, \mathbb{G}_T, g, g_1, g_2, \overline{u}, W)$ from its challenger $\mathcal{CH}$ and sends $\mathcal{A}$ an input $D_{\mathcal{R}}$ consisting of PK, $I_{\mathcal{R}} = \{0,1\}^n$ and a description of R_{id} for $id \in I_{\mathcal{R}}$.

Whenever $\mathcal{A}_1$ asks O_{Extract} for the trapdoor of a relation R_{id} for some identity $id \in I_{\mathcal{R}}$, $\mathcal{F}$ asks its challenger $\mathcal{CH}$ for a signature of the message id and relays the answer to $\mathcal{A}_1$. After polynomially-many queries to O_{Extract}, $\mathcal{A}_1$ comes up with an identity $id^\star$ that was never queried to O_{Extract}. At this stage, $\mathcal{F}$ generates a uniformly distributed lock $c = g^{\omega}$ for a random $\omega \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$. In particular c

has the same distribution as locks generated by Sample. On input of c and the state information transmitted by $\mathcal{A}_1$, $\mathcal{A}_2$ issues new queries to O_{Extract} which all trigger a signing query from $\mathcal{F}$ to $\mathcal{CH}$. Eventually, $\mathcal{A}_2$ is expected to output a proof (d_1, d_2, d_3) such that

$$e(d_1, g) = W \cdot e(F(id^\star), d_2) \cdot e(g^\omega, d_3)$$

which can be re-written as

$$e(d_1 \cdot d_3^{-\omega}, g) = W \cdot e(F(id^\star), d_2).$$

Hence, the pair $(\sigma_1 = d_1 \cdot d_3^{-\omega}, \sigma_2 = d_2)$ passes the verification test of Waters signatures. It is thus a valid forgery since $id^\star$ was not queried for signature by $\mathcal{F}$ as it may not have been queried to O_{Extract} by $\mathcal{A}_1$ or $\mathcal{A}_2$ at any time. □

Together with security results of [36], theorem 1 implies the following corollary.

Corollary 1. *Assuming that an adversary $\mathcal{A}$ breaks the one-wayness of our* ID-THIR *with advantage ε when running in time τ and making q_{td} trapdoor queries, there is an algorithm $\mathcal{B}$ that (τ', ε')-breaks the CDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{4q_{\mathsf{td}}(n+1)} \qquad \tau' \leq t + O(q_{\mathsf{td}}\tau_{exp}),$$

τ_{exp} denoting the time complexity of an exponentiation in $\mathbb{G}$.

4 Shorter Public Keys for Small Identity Spaces

The ID-THIR construction of section 3 assumes a space of identities $I_{\mathcal{R}} = \{0,1\}^n$ where n can be as large as 160. In some applications, this space is quite likely to be much smaller. With time capsule signatures for instance, it is reasonable to settle for initializing the scheme in expectation of 2^{30} time periods.

In this case, the function $F : \{0,1\}^n \to \mathbb{G}$ can be replaced with Boneh and Boyen's selective-ID secure "hash" $F(id) = g_2^{H(id)}h$ [7] where $h \in_R \mathbb{G}$ and $H : \{0,1\} \to \mathbb{Z}_p^*$ is a collision-resistant hash function. This modification results in much shorter public parameters as a single group element $h \in \mathbb{G}$ supersedes the vector $\overline{u}$. The resulting ID-THIR remains one-way under the Diffie-Hellman assumption but the proof of one-wayness requires the Diffie-Hellman solver to guess which identity $id^\star$ will be attacked by $\mathcal{A}$ beforehand.

Theorem 2. *If an adversary $\mathcal{A}$ breaks the one-wayness of the modified* ID-THIR *with probability ϵ in time τ, the CDH problem can be (τ', ϵ')-solved where $\tau' \approx \tau$ and $\epsilon' = \epsilon/|I_{\mathcal{R}}|$.*

Proof. We outline an algorithm $\mathcal{B}$ solving a CDH instance (g^a, g^b) using $\mathcal{A}$ as a subroutine. To do so, $\mathcal{B}$ first picks $\rho \xleftarrow{R} \mathbb{Z}_p^*$ and chooses $id^\star \xleftarrow{R} I_{\mathcal{R}}$ as a guess for the identity to be attacked by $\mathcal{A}$. Public parameters are defined as $g_1 = g^a$, $g_2 = g^b$ and $h = g_2^{-I^\star} g^\rho$, where $I^\star = H(id^\star) \in \mathbb{Z}_p^*$, so that $F(id) = g_2^{H(id)-I^\star} g^\rho$.

Trapdoor queries for identities $id \neq id^\star \in I_\mathcal{R}$ can be answered by choosing $s \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and returning

$$(t_1, t_2) = (F(id)^s \cdot g_1^{-\rho/(I-I^\star)}, g^s \cdot g_1^{-1/(I-I^\star)})$$

with $I = H(id) \in \mathbb{Z}_p^*$. The pair (t_1, t_2) has the correct distribution since

$$(t_1, t_2) = (g_2^a \cdot F(id)^{\tilde{s}}, g^{\tilde{s}})$$

with $\tilde{s} = s - a/(I - I^\star)$.

When $\mathcal{A}$ issues her challenge query, $\mathcal{B}$ fails if the target identity is not $id^\star$. Otherwise, it picks a random $\omega \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and responds with the challenge $c = g^\omega$. A successful attacker $\mathcal{A}$ is then expected to output a triple (d_1, d_2, d_3) satisfying

$$e(d_1, g) = W \cdot e(g^\rho, d_2) \cdot e(g^\omega, d_3)$$

which implies $e(d_1 \cdot d_2^{-\rho} \cdot d_3^{-\omega}, g) = e(g_1, g_2)$ and yields the solution $d_1 \cdot d_2^{-\rho} \cdot d_3^{-\omega}$ that $\mathcal{B}$ was after. □

Since $q_{td} < 2^{30}$ is a reasonable upper bound frequently encountered in the literature, the modified scheme should be preferred whenever $|I_\mathcal{R}| < 2^{30}$.

5 Efficient TCS schemes in the Standard Model

The generic construction [17] of secure TCS from any ID-THIR is very simple and does not involve random oracles. It requires an ordinary digital signature scheme $\Sigma = (\mathsf{Keygen}, \mathsf{Sign}, \mathsf{Verify})$ and an ID-THIR $(\mathsf{Gen}, \mathsf{Sample}, \mathsf{Check}, \mathsf{Extract}, \mathsf{Invert})$. The signer generates a key pair $(\mathsf{PK}, \mathsf{SK}) \leftarrow \Sigma.\mathsf{Keygen}(k)$ while the Time Server runs $\mathsf{Gen}(k)$ to produce $(D_\mathcal{R}, \mathsf{mtd}_\mathcal{R})$ and sets $(\mathsf{TPK}, \mathsf{TSK}) = (D_\mathcal{R}, \mathsf{mtd}_\mathcal{R})$.

To produce a time capsule signature on a message m for time t, the signer samples a random lock/proof pair (c, d) for the relation R_t corresponding to the "identity" $t \in I_\mathcal{R}$. The future signature consists of c and the output σ of $\Sigma.\mathsf{Sign}_\mathsf{SK}(m||c||t)$ which can be verified by running $\Sigma.\mathsf{Verify}_\mathsf{PK}(m||c||t, \sigma)$. The signer also remembers d which is used for pre-hatching. The time release algorithm simply uses the master trapdoor $\mathsf{TSK} = \mathsf{mtd}_\mathcal{R}$ to generate a trapdoor $Z_t = \mathsf{td}_{R_t} = \mathsf{Extract}_{\mathsf{mtd}_\mathcal{R}}(t)$ for the "identity" t. Given a future signature $\langle c, \sigma \rangle$, the hatching algorithm uses $Z_t = \mathsf{td}_{R_t}$ to compute a proof $\tilde{d}$ for the lock c. Upon verification of a hatched or pre-hatched signature $\langle (c, d), \sigma \rangle$, the verifier accepts if $\Sigma.\mathsf{Verify}_\mathsf{PK}(m||c||t, \sigma)$ and $\mathsf{Check}_{D_\mathcal{R}, t}(c, d)$ both return 1 and rejects otherwise.

5.1 A Concrete Scheme

The scheme described below is an example of concrete TCS in the standard model. It combines our ID-THIR scheme with Waters signatures. That is why all parties use common public parameters including the description of bilinear map groups $(\mathbb{G}, \mathbb{G}_T)$ of order $p > 2^k$. In practice however, signers are free to choose their own parameters independently of the Time Server: they can use any secure digital signature in the standard model such as Cramer-Shoup [16].

$\mathsf{Setup}^{\mathsf{TS}}(k,n)$: the Time Server chooses a generator $g \in \mathbb{G}$. It computes $g_v = g^{\alpha_v}$ for a random $\alpha_v \xleftarrow{R} \mathbb{Z}_p^*$. Next, it chooses $g'_v \xleftarrow{R} \mathbb{G}$, computes $W_v = e(g_v, g'_v)$ and selects a random vector $\overline{v} = (v', v_1, \ldots, v_n) \xleftarrow{R} \mathbb{G}^{n+1}$ defining a function $F_v : \{0,1\}^n \to \mathbb{G} : t \to F_v(t) = v' \cdot \prod_{j=1}^{n} v_j^{t_j}$ where $t = t_1 \ldots t_n$ and $t_j \in \{0,1\}$ for all j. The Time Server's private key is $\mathsf{TSK} = {g'_v}^{\alpha_v}$ and the public key is

$$\mathsf{TPK} = \{n, \mathbb{G}, \mathbb{G}_T, g, g_v, g'_v, \overline{v}, W_v\}.$$

$\mathsf{Setup}^{\mathsf{User}}(k,n)$: the user picks $\alpha_u \xleftarrow{R} \mathbb{Z}_p^*$, $g'_u \xleftarrow{R} \mathbb{G}$ and a random $(n+1)$-vector $\overline{u} = (u', u_1, \ldots, u_n) \in \mathbb{G}^{n+1}$ which defines the function $F_u : \{0,1\}^n \to \mathbb{G}$ as $F_u(\mathsf{m}) = u' \cdot \prod_{j=1}^{n} u_j^{m_j}$ where $\mathsf{m} = m_1 \ldots m_n$ and $m_j \in \{0,1\}$ for all j. A collision-resistant hash function $H : \{0,1\}^* \to \{0,1\}^n$ is also chosen. The private key is $\mathsf{SK} = {g'_u}^{\alpha_u}$. The public key is $\mathsf{PK} = (n, g, g_u, g'_u, \overline{u}, W_u, H)$ with $g_u = g^{\alpha_u}$ and $W_u = e(g_u, g'_u)$.

$\mathsf{TSig}(m,t)$: the signer first generates a pair $\big(c, (d_1, d_2, d_3)\big) \in \mathbb{G} \times \mathbb{G}^3$ following these steps.

1. Choose $j_1, j_2 \xleftarrow{R} \mathbb{Z}_p^*$ and compute $c = {g'_v}^{j_1} g^{j_2}$.
2. Pick $r, s \xleftarrow{R} \mathbb{Z}_p^*$ and compute $d_1 = c^s \cdot g_v^{-j_2/j_1} \cdot F_v(t)^r$.
3. Set $d_2 = g^r$ and $d_3 = g^s \cdot g_v^{-1/j_1}$.

Then, he computes $\mathsf{m} = H(m||c||t) \in \{0,1\}^n$ and

$$\sigma = (\sigma_1, \sigma_2) = \big({g'_u}^{\alpha_u} \cdot F_u(\mathsf{m})^{r_u}, g^{r_u}\big)$$

for a randomly chosen $r_u \xleftarrow{R} \mathbb{Z}_p^*$. He outputs $\sigma'_t = \langle (\sigma_1, \sigma_2), c \rangle$ and stores the triple (d_1, d_2, d_3) for later use.

$\mathsf{TVer}(m, \sigma'_t, \mathsf{PK}, \mathsf{TPK}, t)$: parse σ'_t as $\langle (\sigma_1, \sigma_2), c \rangle$ and PK as $(n, g, g_u, g'_u, \overline{u}, W_u, H)$. Check that $c \in \mathbb{G}$ and return 0 otherwise. Return 1 if

$$e(\sigma_1, g) = W_u \cdot e(F_u(\mathsf{m}), \sigma_2)$$

with $\mathsf{m} = H(m||c||t) \in \{0,1\}^n$.

$\mathsf{TRelease}(t, \mathsf{TSK})$: given $\mathsf{TSK} = {g'_v}^{\alpha_v}$, the Time Server picks $r_v \xleftarrow{R} \mathbb{Z}_p^*$ and returns $Z_t = ({g'_v}^{\alpha_v} \cdot F_v(t)^{r_v}, g^{r_v})$.

$\mathsf{Hatch}(\sigma'_t, Z_t)$: parse σ'_t as $\langle (\sigma_1, \sigma_2), c \rangle$ and Z_t as $(z_1, z_2) = ({g'_v}^{\alpha_v} \cdot F_v(t)^{r_v}, g^{r_v})$. Pick r'_v, s and compute

$$(\tilde{d_1}, \tilde{d_2}, \tilde{d_3}) = (z_1 \cdot F_v(t)^{r'_v} \cdot c^s, z_2 \cdot g^{r'_v}, g^s) = ({g'_v}^{\alpha_v} \cdot F_v(t)^{r''_v} \cdot c^s, g^{r''_v}, g^s)$$

where $r''_v = r_v + r'_v$. The hatched signature is

$$\sigma_t = \langle (\sigma_1, \sigma_2), c, (\tilde{d_1}, \tilde{d_2}, \tilde{d_3}) \rangle$$

$\mathsf{PreHatch}(\sigma'_t, d)$: parse σ'_t as $\langle (\sigma_1, \sigma_2), c \rangle$ and d as (d_1, d_2, d_3), return the opened signature $\sigma_t = \langle (\sigma_1, \sigma_2), c, (d_1, d_2, d_3) \rangle$.

$\mathsf{Ver}(m, \sigma_t, \mathsf{PK}, \mathsf{TPK}, t)$: parse σ_t as $\langle(\sigma_1, \sigma_2), c, (d_1, d_2, d_3)\rangle$, the signer's public key PK as $(n, g, g_u, g'_u, \overline{u}, W_u, H)$ and TPK as $(n, g, g_v, g'_v, \overline{v}, W_v)$. Return 1 if

$$e(d_1, g) = W_v \cdot e(F_v(t), d_2) \cdot e(c, d_3) \tag{3}$$
$$e(\sigma_1, g) = W_u \cdot e(F_u(\mathsf{m}), \sigma_2) \tag{4}$$

where $\mathsf{m} = H(m||c||t) \in \{0,1\}^n$.

We note that the latter verification algorithm can be optimized as follows. Instead of sequentially verifying relations (3) and (4), the verifier can randomly choose $\beta_1, \beta_2 \stackrel{R}{\leftarrow} \mathbb{Z}_p^*$ and accept the signature if

$$\frac{1}{W_v^{\beta_1} \cdot W_u^{\beta_2}} \cdot \frac{e(g, d_1^{\beta_1} \cdot \sigma_1^{\beta_2})}{e(F_v(t), d_2^{\beta_1}) \cdot e(c, d_3^{\beta_1}) \cdot e(F_u(\mathsf{m}), \sigma_2^{\beta_2})} = 1_{\mathbb{G}_T}.$$

Indeed, if we raise both members of (3) and (4) to the powers β_1 and β_2 respectively, we observe that the above verification test fails with overwhelming probability if either (3) or (4) does not hold. A product of four pairings (which is much faster to compute than a sequence of 4 independent pairings as discussed in [25]) suffices to check both conditions.

As explained in [17], the unconditional security against the signer follows from the correctness and soundness properties of the ID-THIR scheme. Theorem 2 in [17] shows that a successful cheating verifier obtaining a full signature without the help of the Time Server or the signer implies a successful inverter for the underlying ID-THIR scheme. The proof of this fact entails a degradation factor of q_{TSig} which is the number of queries to $\mathsf{O}_{\mathsf{TSig}}$.

Corollary 2. *If a cheating verifier $\mathcal{B}$ has advantage ε within running time τ when making q_{TR} queries to O_{TR} and q_{TSig} queries to O_{TSig}, there is an algorithm that (τ', ε')-breaks the CDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{4q_{\mathsf{TR}}q_{\mathsf{TSig}}(n+1)} \qquad \tau' \leq t + O\big((q_{\mathsf{TR}} + q_{\mathsf{TSig}})\tau_{exp}\big),$$

where τ_{exp} is the time complexity of an exponentiation in $\mathbb{G}$.

It was also proved in [17] that a successful dishonest Time Server implies a chosen-message attacker breaking the underlying signature scheme with the same advantage. Together with results from [36], this yields the following corollary which completes the proof that a secure and efficient time capsule signature exists in the standard model under the Diffie-Hellman assumption.

Corollary 3. *If a cheating Time Server $\mathcal{C}$ has advantage ε within running time τ when making q_{TSig} queries to O_{TSig}, there is an algorithm that (τ', ε')-breaks the CDH assumption where*

$$\varepsilon' \geq \frac{\varepsilon}{4q_{\mathsf{TSig}}(n+1)} \qquad \tau' \leq t + O(q_{\mathsf{TSig}}\tau_{exp}),$$

where τ_{exp} is the same quantity as in corollary 2.

5.2 Efficiency Improvements for Smaller Number of Periods

In section 5.1, the Time Server performs the setup for a large number of time periods. As discussed in section 4, $N < 2^{30}$ is a smaller but quite realistic[1] number of time periods. In this case, the Server's public key can be shortened by replacing the Waters "hash" $F_v(t) = v' \prod_{j=1}^{n} v_j^{t_j}$ with $F_v(t) = g_2^{H(t)} h$, for a random element $h \in_R \mathbb{G}$ and a collision-resistant hash function $H : \{0,1\}^{\lceil \log_2 N \rceil} \rightarrow \mathbb{Z}_p^*$. The degradation factor of corollary 2 becomes $O(N \cdot q_{\mathsf{TSig}})$ instead of $O(q_{\mathsf{TR}} \cdot q_{\mathsf{TSig}})$.

We note that signers are free to implement the scheme with their favourite signing algorithm and they may prefer using short public keys. In this case, they can use the same common public parameters $(\mathbb{G}, \mathbb{G}_T)$ with other pairing-based signatures in the standard model. For instance, combining the selective-message secure signature of [7] at the Time Server with Strong Diffie-Hellman-based signatures [6, 23] at the signer provides an efficient TCS scheme under the Strong Diffie-Hellman assumption. In this case, we have a tight reduction under a stronger assumption in corollary 3.

5.3 Reducing the Public Storage for the Time Server

A shortcoming of time capsule signatures considered in sections 5.1 and 5.2 is that Time Servers have to publish and store a number of group elements which is linear in the number of past time periods at any time. After n periods have passed, the server has to publish a bulletin board with $O(n)$ trapdoors.

To overcome this limitation also present in some time release primitives [31, 5, 15], Boneh et al. [8] proposed to use forward-secure primitives [1, 3, 14] backwards. Roughly said, forward-secure schemes protect the confidentiality or the authenticity of past communications by preventing past (but obviously not future) private keys to be computable from current ones. Hence, to encrypt a message for period t in the future, one can simply encipher it for period $N - t$ using a forward-secure public key encryption scheme [14, 8] prepared for N stages using the tree-like structure of [14]. Thanks to the latter, a private key for period $N - t$ allows anyone to derive keys for stages $N - t + 1, \ldots, N$. In terms of time release primitives, the current private key allows recovering keys for past periods so that the public storage of the server never exceeds $O(\log^2 N)$ group elements.

It is not hard to see that aforementioned tricks apply to our context for a reasonably small number of time periods. At the server, we simply have to replace the selective-message secure signature of Boneh-Boyen [7] by the hierarchical selective-message secure signature suggested by the hierarchical IBE of [8]. It amounts to use the keying technique of a recently proposed forward-secure signature [11] in reverse. To generate a future signature for period t, the signer actually prepares it for period $N - t$. At period t, the Time Server only stores the trapdoor for period t (which is the "forward-secure private key" of period $N - t$) that allows deriving trapdoors for stages $1, \ldots, t - 1$.

[1] For instance, a scheme could be used over more than 2000 years with 2^{30} periods of one minute.

In this case, the security against cheating verifiers relies on a variant of the Diffie-Hellman problem which is to compute $g^{a^{\ell+1}}$ given $(g, g^a, \ldots, g^{a^\ell})$ where $\ell = \log_2 N$.

6 Conclusion

In this paper, we put forth the first practical construction of time capsule signature that provably fits the security definitions of [17] without using the random oracle heuristic. It stems from an efficient example of a recently introduced primitive which is of independent interest and in turn builds on Waters signatures and the Diffie-Hellman assumption.

We note that time capsule signatures with tight reductions remain elusive (even in the random oracle model). Solving this problem would require a new approach for constructing them since the generic construction of ID-THIRs entails a loss of $O(\mathsf{TSig})$ in the security bound against verifiers.

References

1. R. Anderson. Two Remarks on Public Key Cryptology. Invited lecture, *ACM Conference on Computer and Communications Security*, 1997.
2. M. Bellare, S. Goldwasser. Encapsulated key-escrow. In 4^{th} *ACM Conference on Computer and Communications Security*, ACM Press, pages 78–91, 1997.
3. M. Bellare, S. Miner. A Forward-Secure Digital Signature Scheme. In *Crypto'99*, *LNCS* 1666, pp. 431–448. Springer, 1999.
4. M. Bellare, P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *1st ACM Conference on Computer and Communications Security*, pages 62–73, ACM Press, 1993.
5. I. Blake, A.-C.-F. Chan, Scalable, Server-Passive, User-Anonymous Timed Release Public Key Encryption from Bilinear Pairing. In *ICDCS'05*, IEEE Computer Society, pages 504–513, 2005.
6. D. Boneh, X. Boyen. Short signatures without random oracles. In *Eurocrypt'04*, *LNCS* 3027, pages 56–73. Springer, 2004.
7. D. Boneh, X. Boyen. Efficient selective-ID secure identity based encryption without random oracles. In *Eurocrypt'04*, *LNCS* 3027, pp. 223–238. Springer, 2004.
8. D. Boneh, X. Boyen, E.-J. Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In *Eurocrypt'05*, *LNCS* 3494, pp. 440–456. Springer, 2005.
9. D. Boneh, M. Franklin. Identity-based encryption from the Weil pairing. In *Crypto'01*, *LNCS* 2139, pp. 213–229. Springer, 2001.
10. D. Boneh, M. Naor. Timed Commitments. *Advances in Cryptology - Crypto'00*, *LNCS* 1880, pages 236–254, Springer, 2000.
11. X. Boyen, H. Shacham, E. Shen, B. Waters. Forward-Secure Signatures with Untrusted Update. In *ACM CCS'06*, ACM Press, 2006.
12. X. Boyen and B. Waters. Compact Group Signatures Without Random Oracles. In *Eurocrypt'06*, *LNCS* 4004, pages 427–444, Springer, 2006.
13. R. Canetti, O. Goldreich, S. Halevi. The random oracle methodology, revisited. Journal of the ACM 51(4), pages 557–594, 2004.

14. R. Canetti, S. Halevi, J. Katz. A forward secure public key encryption scheme. In *Eurocrypt'03*, *LNCS* 2656, pages 254–271. Springer, 2003.
15. J. H. Cheon, N. Hopper, Y. Kim, I. Osipkov Timed-Release and Key-Insulated Public Key Encryption. In *Financial Cryptography 2006*, to appear in *LNCS* Series. Available from `http://eprint.iacr.org/2004/231`.
16. R. Cramer and V. Shoup. Signature schemes based on the strong rsa assumption. In 7^{th} *ACM Conference on Computer and Communications Security*, pages 46–51. ACM Press, 1999
17. Y. Dodis, D.-H. Yum. Time Capsule Signature. In *Financial Crypto'05*, *LNCS* 3570, pages 57–71, Springer 2005.
18. U. Feige and A. Shamir. Witness indistinguishable and witness hiding protocols. In *STOC'90*, pages 416–426, *ACM Press*, 1990.
19. A. Fiat and A. Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *Crypto'86*, *LNCS* 263, pages 186–194. Springer, 1986.
20. J. Garay, M. Jakobsson, *Timed-Release of Standard Digital Signatures*, In *Financial Crypto'02*, *LNCS* 2357, pages 168–182, Springer, 2002.
21. J. Garay, C. Pomerance, *Timed Fair Exchange of Standard Signatures*, In *Financial Crypto'03*, *LNCS* 2742, pages 190–207, Springer, 2003.
22. C. Gentry, A. Silverberg. Hierarchical ID-based cryptography. In *Asiacrypt'02*, *LNCS* 2501, pages 548–566. Springer, 2002.
23. C. Gentry. Practical Identity-Based Encryption Without Random Oracles. In *Eurocrypt'06*, *LNCS* 4004, pages 445–464. Springer, 2006.
24. S. Goldwasser, S. Micali, R. Rivest. A Digital Signature Scheme Secure Against Adaptive Chosen-Message Attacks. *SIAM J. Comput.* 17(2), pages 281–308, 1988.
25. R. Granger, N. P. Smart. On Computing Products of Pairings. Cryptology ePrint Archive: Report 2006/172, 2006.
26. J. Groth, R. Ostrovsky, A. Sahai. Perfect Non-interactive Zero Knowledge for NP. In *Eurocrypt'06*, *LNCS* 4004, pages 339–358, Springer, 2006.
27. J. Groth, R. Ostrovsky, A. Sahai. Non-interactive Zaps and New Techniques for NIZK. *Crypto'06*, *LNCS* 4117, pages 97–111, 2006.
28. J. Groth. Simulation-Sound NIZK Proofs for a Practical Language and Constant Size Group Signatures. In *Asiacrypt'06*, *LNCS* 4284, pages 444–459, Springer, 2006.
29. E. Kiltz, A. Mityagin, S. Panjwani, B. Raghavan. Append-Only Signatures. In *ICALP'05*, *LNCS* 3580, pages 434–445, Springer, 2005.
30. W. Mao. Timed-Release Cryptography. In *Selected Areas in Cryptography'01*, *LNCS* 2259, pages 342–357, Springer, 2001.
31. M.C. Mont, K. Harrison. M. Sadler, *The HP time vault service: Innovating the way confidential information is disclosed at the right time*, in HP Lab. Report HPL-2002-243, 2002.
32. K. G. Paterson, J. C. N. Schuldt. Efficient Identity-based Signatures Secure in the Standard Model. In *ACISP'06*, *LNCS* 4058, pages 207–222, Springer, 2006.
33. R. Rivest, A. Shamir, D.A. Wagner. Time-lock puzzles and timed-release crypto. MIT LCS Tech. Report MIT/LCS/TR-684, 1996.
34. A. De Santis and G. Persiano. Zero-knowledge proofs of knowledge without interaction. In *FOCS'92*, pages 427–436, 1992.
35. A. Shamir. Identity based cryptosystems and signature schemes. In *Crypto'84*, *LNCS* 196, pages 47–53. Springer, 1984.
36. B. Waters. Efficient Identity-Based Encryption Without Random Oracles. In *Eurocrypt'05*, *LNCS* 3494, pages 114–127. Springer 2005.