Optimally Tight Security Proofs for Hash-Then-Publish Time-Stamping

Ahto Buldas[1,2,3,⋆] and Margus Niitsoo[1,3,⋆⋆]

[1] Cybernetica AS, Akadeemia tee 21, 12618 Tallinn, Estonia
[2] Tallinn University of Technology, Raja 15, 12618 Tallinn, Estonia
[3] University of Tartu, Liivi 2, 50409 Tartu, Estonia

Abstract. We study the security of hash-then-publish time-stamping schemes and concentrate on the tightness of security reductions from the collision-resistance of the underlying hash functions. While the previous security reductions create a quadratic loss in the security in terms of time-success ratio of the adversary being protected against, this paper achieves a notably smaller loss of power 1.5. This is significant for two reasons. Firstly, the reduction is asymptotically optimally tight, as the lower bound of 1.5 on the power was proven recently by the authors in ACISP 2009 and this is the first application for which optimality in this sense can be demonstrated. Secondly, the new reduction is the first one efficient enough to allow meaningful security guarantees to be given for a global-scale time-stamping service based on 256 bit hash functions, which considerably increases the efficiency of possible practical solutions.

1 Introduction

Time stamps are proofs that electronic data was created at certain time. Time stamps support rights protection as well as extending the lifetime of public key digital signatures considering the possible revocation of public-key certificates.

Before 1990, it was believed that the only possible way to achieve secure time-stamping is to use a trusted third party who adds time-readings to electronic data and then signs the data by using a public-key digital signature scheme. Although this scheme has been in use, it does have drawbacks. The assumption of a trusted third party is rather strong and often not feasible in the global corporate scale as nearly everyone has their own interests. Even when such a trusted party could be found, it is generally impossible to guarantee absolute security of the private signature keys. It would therefore be desirable to use time-stamping schemes that are free of secret keys and do not assume ultimate trustworthiness of third parties.

The so-called hash-then-publish time-stamping schemes were first introduced in 1990 by Haber and Stornetta [6] in connection with attempts to eliminate secret-based cryptography and trusted third parties from time-stamping schemes. In such a scheme,

⋆ Supported by the European Regional Development Fund through the Estonian Center of Excellence in Comp. Sci., by Estonian SF grant no. 6944, and by EU FP6-15964: “AEOLUS”.

⋆⋆ Supported by Estonian SF grant no. 6944 and the Tiger University Program of the Estonian Information Technology Foundation.

a collection of N documents is hashed down to a single digest of few dozen bytes that is then published in a widely available medium such as a newspaper. Using Merkle hash trees [9] as a hashing scheme provides a possibility of creating compact certificates (of size $O(\log N)$) for each one of the N documents. To create such a certificate, it is sufficient to store all sibling hash values in the corresponding path in the hash tree from a document to the root of the tree. The sibling hash values are sufficient to re-compute the root hash value from the document and as such they can be used as a *proof of membership*. Based on this idea, Haber and Stornetta then drafted a large-scale time-stamping scheme [1] where a giant Merkle tree is created co-operatively by numerous servers all over the world and the root value is published in newspapers as the hash value of this particular unit of time. In such schemes N is potentially very large.

It might seem obvious that the security of hash-then-publish time-stamping schemes can be reduced to the collision-resistance of the hash function. However, the first correct security proof of such a scheme was published as late as 2004 [5]. It then became evident that the number N of time-stamps explicitly affects the efficiency (security guarantee) of the security proof. In the very first security proof [5] it was shown that if there is an adversary with running time t that is able to backdate a document with probability δ, then there is also a collision-finding adversary that works in time $t' \approx 2t$ and succeeds with probability $\delta' \approx \frac{\delta^2}{N}$. When measuring security in terms of time-success ratio introduced by Luby [8] we have to use $2N \cdot \frac{t}{\delta^2}$-collision resistant hash functions to have a $\frac{t}{\delta}$-secure time-stamping scheme. This means that the hash function must be roughly $\frac{2N}{\delta}$ times more secure against collisions than the time-stamping system constructed from it is against backdating. As N could be very large, the security requirements for the hash function may grow unreasonably large. Indeed, it is mentioned in [5] that such a security proof is practical only for hash functions with 400 or more output bits.

In [4], a more efficient security proof was given, where $\frac{t'}{\delta'} \approx 48\sqrt{N} \cdot \frac{t}{\delta^2}$. This was a considerable improvement because it allowed for much larger values of N. In this paper, we propose a new security reduction, where $\frac{t'}{\delta'} \approx 14\sqrt{N} \cdot \frac{t}{\delta^{1.5}}$, i.e. we get a power 1.5 reduction instead a quadratic one in terms of time-success ratio. This allows us to use shorter hash functions in practical applications while still maintaining good security guarantees. Based on a recently proved separation result [2] we also argue why the exponent 1.5 is the least achievable.

2 Notation

By $x \leftarrow \mathcal{D}$ we mean that x is chosen randomly according to a distribution $\mathcal{D}$. By $\mathbf{E}[X]$ we mean the average of a random variable X. If A is a probabilistic function or a Turing machine, then $x \leftarrow \mathsf{A}(y)$ means that x is chosen according to the output distribution of A on an input y. If $\mathcal{D}_1, \ldots, \mathcal{D}_m$ are distributions and $F(x_1, \ldots, x_m)$ is a predicate, then $\mathsf{Pr}\left[x_1 \leftarrow \mathcal{D}_1, \ldots, x_m \leftarrow \mathcal{D}_m \colon F(x_1, \ldots, x_m)\right]$ is the probability that $F(x_1, \ldots, x_m)$ is true after the ordered assignment of $x_1, \ldots, x_m$. For functions $f, g \colon \mathbb{N} \to \mathbb{R}$, we write $f(k) = O(g(k))$ if there are $c, k_0 \in \mathbb{R}$, so that $f(k) \leq cg(k)$ ($\forall k > k_0$). We write $f(k) = \omega(g(k))$ if $\lim\limits_{k \to \infty} \frac{g(k)}{f(k)} = 0$. If $f(k) = k^{-\omega(1)}$, then f is *negligible*. For every two functions $f(k)$ and $g(k)$, we will write $f \gtrsim g$ iff $f(k) \geq g(k) - k^{-\omega(1)}$. A Turing machine M is *poly-time* if it runs in time $k^{O(1)}$, where k is the input size.

Let $\mathcal{F} = \{\mathcal{F}_k\}_{k\in\mathbb{N}}$ be a function family such that every $h \leftarrow \mathcal{F}_k$ is a function $h\colon \{0,1\}^{\ell(k)} \to \{0,1\}^k$, where $\ell(k) = k^{O(1)}$ and $\ell(k) > k$ for every $k \geq 0$. We say that $\mathcal{F}$ is *collision-free* if for every poly-time (non-uniform) Turing machine A:

$$\Pr\left[h \leftarrow \mathcal{F}_k,\ (x,x')\leftarrow\mathsf{A}(1^k,h)\colon x \neq x',\ h(x) = h(x')\right] = k^{-\omega(1)} \ .$$

3 Hash-then-Publish Time-Stamping

A time-stamping procedure consists of the following two general steps:

1. Client sends a request $x \in \{0,1\}^k$ to Server.
2. Server binds x with a time value t and sends Client a time-certificate c.

Time-stamping protocols process requests in batches $\mathcal{X}_1, \mathcal{X}_2, \mathcal{X}_3 \ldots$ that we call *rounds*. The rounds correspond to time periods of fixed duration (one hour, one day, etc.) After the t-th period, a short commitment $r_t = \mathsf{Com}(\mathcal{X}_t)$ of $\mathcal{X}_t$ is published. A request $x \in \mathcal{X}_t$ precedes another request $x' \in \mathcal{X}_{t'}$ if $t < t'$. The requests of the same batch are considered simultaneous. For this scheme to be efficient there must be an efficient way to prove inclusions $x \in \mathcal{X}_t$, i.e. there is a verification algorithm Ver that on input a request x, a certificate c and a commitment r_t returns true if $x \in \mathcal{X}_t$. On the one hand, it should be easy to create certificates for the members $x \in \mathcal{X}_t$, i.e. there has to be an efficient certificate generation algorithm Cert that outputs a certificate $c = \mathsf{Cert}(x, \mathcal{X}_t)$. On the other hand, for security, it must be infeasible to create such proofs for non-members $y \notin \mathcal{X}_t$, i.e. it is hard to find a certificate c' so that $\mathsf{Ver}(y, c', r_t) = \mathsf{true}$.

Definition 1. *A time-stamping scheme is a triple* $\mathsf{T} = (\mathsf{Com}, \mathsf{Cert}, \mathsf{Ver})$ *of efficient algorithms, where:*

– Com *is a commitment algorithm which, on input a set* $\mathcal{X}$ *of requests, outputs a commitment* $r = \mathsf{Com}(\mathcal{X})$.
– Cert *is a certificate generation algorithm which, on input a set* $\mathcal{X}$ *and an element* $x \in \mathcal{X}$*, generates a certificate* $c = \mathsf{Cert}(x, \mathcal{X})$.
– Ver *is a verification algorithm which, on input a request* x*, a certificate* c *and a commitment* r*, outputs* yes *or* no*, depending on whether* x *is a member of* $\mathcal{X}$ *(the set that corresponds to the commitment* r*). It is assumed that for every set* $\mathcal{X}$ *of requests and every member-request* $x \in \mathcal{X}$ *the following correctness condition holds:*

$$\mathsf{Ver}(x, \mathsf{Cert}(x, \mathcal{X}), \mathsf{Com}(\mathcal{X})) = \mathsf{yes} \ . \qquad (1)$$

3.1 Security Condition for Time-Stamping Schemes

It was shown in [5] that giving a consistent security definition for *hash-then-publish* time-stamping schemes is not an easy task. Intuitively, a time-stamping adversary *back-dates a document that never existed before*, but the "existence" itself is not that easy to capture in formal definitions. In this paper, we use the so-called *entropy-based security* condition [3] that models the "fresh" documents by using high-entropy distributions.

Such approach has been the most common in this line of research. This security condition is inspired by the following attack-scenario with a malicious Server:

1. Server computes a commitment r and publishes it. Server is potentially malicious, so there are no guarantees that r is created by applying Com to a set $\mathcal{X}$ of requests.
2. Alice creates an invention $\mathcal{D}_\mathrm{A} \in \{0,1\}^*$ and protects it by obtaining a time stamp.
3. Some time later, $\mathcal{D}_\mathrm{A}$ is disclosed to the public and Server tries to steal it by showing that the invention was known to Server long before Alice time-stamped it. He creates a slightly modified version $\mathcal{D}'_\mathrm{A}$ of A, i.e. changes the invertor's name, modifies the creation time, and possibly rewords the document in a suitable way.
4. Finally, Server back-dates a hash value x of $\mathcal{D}'_\mathrm{A}$, by finding a certificate c, so that $\mathsf{Ver}(x, c, r) = \mathsf{yes}$. It is shown in [3] that the hash function that computes x from $\mathcal{D}'_\mathrm{A}$ must convert poly-sampleable high entropy input distributions to high entropy output distributions, and this is in fact also a sufficient condition.

Security definitions for time-stamping are usually based on this scenario. However, to our knowledge, there have been no academic discussions whether such a scenario is sufficient for the security level we really expect. One major assumption that has been made here is that before creating and publishing the commitment r, Server has no information about the invention $\mathcal{D}_A$. For example, if Alice obtains a time stamp for $\mathcal{D}_A$ from malicious Server *before* r is published (i.e. steps 1 and 2 are exchanged) then during the computation of r, Server knows the time stamp request x which is partial information about $\mathcal{D}_A$. So, one may imagine that Server tries to extract useful information from x about $\mathcal{D}_A$, create a request x' for a similar document $\mathcal{D}'_A$ that describes the same invention, and then refuse to issue a time stamp for Alice. If such an attack succeeds, Server has the earliest time stamp for Alice's invention. But there are many practical objections against such an attack:

- Time stamp requests only contain a relatively short hash value of $\mathcal{D}_A$ which (in practice) can hardly contain any useful information about the invention.
- It is improbable that all time-stamping servers could be simultaneously corrupted and Alice is usually free to commit to several of them at the same time. This means that malicious servers who try to delay the publishing of r in order to have more time for creating x' based on x will "lose the race" against honest servers who create their time stamps earlier.

So, the assumption that server has no information about $\mathcal{D}_A$ when publishing r is heuristic but still justified in practice and hence it is reasonable to study the security of time-stamping schemes under such assumption.

To formalize such an attack, a two-staged adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ is used. The first stage A_1 computes and outputs r and an advice string a, which contains useful information for the second stage A_2. Note that a may contain all random coins of A_1, which makes all useful information that A_1 gathered available to A_2. After that, the second stage A_2 finds a new x (which is assumed to be a random variable with a sufficient amount of entropy) and a certificate c such that $\mathsf{Ver}(x, c, r) = \mathsf{yes}$. Note that x must be unpredictable because otherwise x could have been pre-computed by A_1 and there would be nothing wrong in proving that x existed before r was computed and published.

Hence, for defining the security of time-stamping schemes, the class of possible adversaries is restricted. Only adversaries that produce unpredictable x are considered [3]. A poly-time adversary $(\mathsf{A}_1, \mathsf{A}_2)$ is *unpredictable* if for every poly-time predictor Π:

$$\mathsf{Pr}\left[(r,a) \leftarrow \mathsf{A}_1(1^k), x' \leftarrow \Pi(r,a), (x,c) \leftarrow \mathsf{A}_2(r,a) \colon x' = x\right] = k^{-\omega(1)} \ . \tag{2}$$

It is reasonable to assume that a contains all internal random coins of A_1 (see [3] for more details). An equivalent definition for the unpredictability of A is that the probability $\mathsf{Pr}\,[\mathsf{Equ}]$ that $\mathsf{A}_2(r,a)$ outputs the same x twice is negligible. We can also say that x should have large (super-logarithmic in k) conditional min entropy $H_\infty(x \mid \mathsf{A}_1(1^k))$.

Definition 2. *A time-stamping scheme is secure if for every unpredictable* $(\mathsf{A}_1, \mathsf{A}_2)$:

$$\mathsf{Pr}\left[(r,a) \leftarrow \mathsf{A}_1(1^k), (x,c) \leftarrow \mathsf{A}_2(r,a) \colon \mathsf{Ver}(x,c,r) = \mathsf{yes}\right] = k^{-\omega(1)} \ . \tag{3}$$

3.2 Hash Tree Time-Stamping Schemes

The commitments r_t are computed as the root hash values of Merkle hash trees [9]. To make the paper more self-contained, we outline the basic facts about hash-chains and how they are used in time-stamping. We use the notation and definitions introduced in [3]. By $()$ we mean an empty list.

Definition 3 (Hash-Chain). *Let* $h\colon \{0,1\}^{2k} \to \{0,1\}^k$ *be a twice-compressing hash function and* $x, y \in \{0,1\}^k$*. By an* h-link *from* x *to* y *we mean a pair* (s,b)*, where* $s \in \{0,1\}^k$ *and* $b \in \{0,1\}$*, such that either* $b = 0$ *and* $y = h(x\|s)$*, or* $b = 1$ *and* $y = h(s\|x)$*. By an* h-chain *from* x *to* y *we mean a (possibly empty) list* $c = ((s_1,b_1), \ldots, (s_\ell,b_\ell))$ *of* h*-links, such that either* $c = ()$ *and* $x = y$*; or (2) there is a sequence* $y_0, y_1, \ldots, y_\ell$ *of* k*-bit strings, such that* $x = y_0$*,* $y = y_\ell$*, and* (s_i, b_i) *is an* h*-link from* y_{i-1} *to* y_i *for every* $i \in \{1, \ldots, \ell\}$*. We denote by* $\mathsf{Chain}_h(x,c) = y$ *the proposition that* c *is an* h*-chain from* x *to* y*. Note that* $\mathsf{Chain}_h(x,()) = x$ *for every* $x \in \{0,1\}^k$*. By the* shape $\rho(c)$ *of* c *we mean the* ℓ*-bit string* $b_1 b_2 \ldots b_\ell$*.*

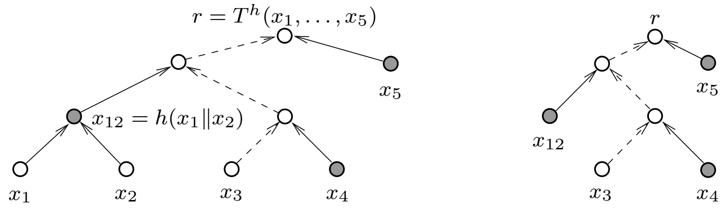

Fig. 1. A hash tree for $\mathcal{X} = \{x_1, \ldots, x_5\}$ and the hash chain $c = ((x_4, 0), (x_{12}, 1), (x_5, 0))$ with shape $\rho(c) = 010$ for x_3

Hash-tree time-stamping schemes use Merkle trees to compute the commitments r_t for batches $\mathcal{X}_t$. The commitment $\mathsf{Com}(\mathcal{X}_t)$ of a batch $\mathcal{X}_t = \{x_1, \ldots, x_N\}$ is $r_t = T^h(x_1, \ldots, x_N) \in \{0,1\}^k$, where T^h is a tree-shaped hashing scheme. A certificate for $x \in \mathcal{X}_t$ is a hash chain c such that $\mathsf{Chain}_h(x, c) = r_t$. The verification procedure $\mathsf{Ver}(x, c, r_t)$ returns yes if $\mathsf{Chain}_h(x, c) = r_t$. In this work, we denote the hash-tree time-stamping scheme by T^h. An example of a hash-tree scheme is depicted in Fig. 1.

Hash-forest time-stamping schemes are obvious generalizations of hash tree schemes. Input for these schemes is a sequence of batches $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_m$ and the commitments are sequences $r = (r_1, r_2, \ldots, r_m)$ of hash values, where every $r_i = \mathsf{Com}(\mathcal{X}_i)$ is computed by using a hash-tree scheme. A certificate for $x \in \mathcal{X}_t$ is a pair $c = (c', t)$ where c' is a hash chain such that $\mathsf{Chain}_h(x, c') = r_t$. The verification procedure $\mathsf{Ver}(x, c, r)$, having as input a request x, a certificate $c = (c', t)$, and a commitment $r = (r_1, r_2, \ldots, r_m)$ returns yes whenever there is $t \in \{1, \ldots, m\}$ and $\mathsf{Chain}_h(x, c) = r_t$. By the *shape* $\rho(c)$ of $c = (c', t)$ we mean the pair $(\rho(c'), t)$.

4 Existing Security Proofs

It was shown in [5] that this scheme cannot be proved secure in a traditional black-box way by assuming only the one-wayness and collision-resistance of h. In [5] they also define a restricted scheme, with a modified verification procedure that uses a set $\mathcal{N}$ of *allowed shapes* with size $|\mathcal{N}| = N$ and the verification procedure Ver was completed with an additional check for $\rho(c) \in \mathcal{N}$. Note that N can be considered as the total capacity of the time-stamping system, i.e. the total number of time-stamps that can be securely issued in the system. All the known security proofs for hash-tree or hash-forest schemes use the following general collision-extraction property:

Definition 4 (Collision-Extraction Property). *If* $\mathsf{Ver}^h(x_1, c_1, r) = \mathsf{Ver}^h(x_2, c_2, r) =$ yes, $\rho(c_1) = \rho(c_2)$, *and* $(x_1, c_1) \neq (x_2, c_2)$, *then the h-calls of* $\mathsf{Ver}^h(x_i, c_i, r)$ *(*$i = 1, 2$*) comprise an h-collision.*

Essentially, this means that given two certificates of the same shape, we can always find a collision. For hash trees or hash forests it is rather easy to see: if two different hash chains c and c' have the same shape and the same root value, there must be an index l such that $c_l \neq c'_l$ but $h(c_l) = h(c'_l)$ which gives the collision that we need. Note that this property also implies that the number of different time-stamp requests per round is limited to N, for otherwise we would have a collision to the hash function we use.

We now proceed to describe the reduction itself. However, in order to give a better intuition to the results we use an iterative process of proving increasingly more precise bounds. All security reductions we illustrate use the following general schema. Having an adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ for a time-stamping scheme T^h with success

$$\delta(k) = \Pr\left[h \leftarrow \mathcal{F}_k, (r, a) \leftarrow \mathsf{A}_1(1^k, h), (x, c) \leftarrow \mathsf{A}_2(r, a) \colon \mathsf{Ver}^h(x, c, r) = \mathsf{yes}\right] \quad . \quad (4)$$

and running time $t = t(k)$, we construct a collision finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ (Fig. 2) with approximate running time $t' \approx m \cdot t$, where m is a reduction-specific parameter and then analyze the success δ' of the collision finder. Although the running time t and the success δ of A depend on the security parameter k, we will use the shorthand notations t and δ instead of $t(k)$ and $\delta(k)$. Let Equ denote the event that $x_i = x_j$ for some

1. Compute $(r,a) \leftarrow \mathsf{A}_1(1^k, h)$.
2. Generate m independent samples: $(x_1, c_1) \leftarrow \mathsf{A}_2(r,a), \ldots, (x_m, c_m) \leftarrow \mathsf{A}_2(r,a)$.
3. Find $x_i \neq x_j$ such that $\mathsf{Ver}^h(x_i, c_i, r) = \mathsf{Ver}^h(x_j, c_j, r) = \mathsf{yes}$ and $\rho(c_i) = \rho(c_j)$.
4. If such a pair was found, use it to extract a collision and output it. Otherwise, output $\perp$.

Fig. 2. Generic collision finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$

$i \neq j$ and $\overline{\mathsf{Equ}}$ denote the opposite event, i.e. that all the x_i-s are different. Considering the collision-extraction property, it would be good if all the successfully back-dated bit-strings were different because then it would be sufficient to find two back-dating certificates of the same shape. Let Coll denote the event that $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ finds a collision for h. A general estimate for the success of the collision finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ is:

$$\mathsf{Pr}[\mathsf{Coll}] \geq \mathsf{Pr}\left[\mathsf{Coll} \cap \overline{\mathsf{Equ}}\right] = \mathsf{Pr}\left[\mathsf{Coll} \mid \overline{\mathsf{Equ}}\right] \cdot (1 - \mathsf{Pr}[\mathsf{Equ}]) \gtrapprox \mathsf{Pr}\left[\mathsf{Coll} \mid \overline{\mathsf{Equ}}\right] \ ,$$

because $\mathsf{Pr}[\mathsf{Equ}] = k^{-\omega(1)}$ due to the unpredictability of $(\mathsf{A}_1, \mathsf{A}_2)$. We can therefore neglect the event Equ in the analysis on the security reductions, i.e. we can just assume that all $x_1, \ldots, x_m$ are different, and use the fact that the success probability of the collision-finder is $\delta' \gtrapprox \mathsf{Pr}\left[\mathsf{Coll} \mid \overline{\mathsf{Equ}}\right]$. Let

$$\begin{aligned}
\mathsf{Pr}[h,r,a] &= \mathsf{Pr}\left[H \leftarrow \mathfrak{F}_k, (R,A) \leftarrow \mathsf{A}_1(1^k) \colon H = h, R = r, A = a\right] \ , \\
\delta^{(n)}_{h,r,a} &= \mathsf{Pr}\left[(x,c) \leftarrow \mathsf{A}_2(r,a) \colon \mathsf{Ver}^h(x,c,r) = \mathsf{yes},\ \rho(c) = n\right] \ , \\
\delta^{(n)} &= \mathop{\mathbf{E}}_{h,r,a}\left[\delta^{(n)}_{h,r,a}\right], \text{and} \qquad (5) \\
\delta_{h,r,a} &= \delta^{(1)}_{h,r,a} + \ldots + \delta^{(N)}_{h,r,a} \ .
\end{aligned}$$

We have $\delta = \sum_{h,r,a} \mathsf{Pr}[h,r,a] \cdot \delta_{h,r,a} = \mathop{\mathbf{E}}_{h,r,a}[\delta_{h,r,a}]$ and $\delta = \delta^{(1)} + \ldots + \delta^{(N)}$. The success probability of the collision finder is:

$$\delta' \gtrapprox \sum_{h,r,a} \mathsf{Pr}[h,r,a] \cdot f(m; \delta^{(1)}_{h,r,a}, \ldots, \delta^{(N)}_{h,r,a}) = \mathop{\mathbf{E}}_{h,r,a}\left[f(m; \delta^{(1)}_{h,r,a}, \ldots, \delta^{(N)}_{h,r,a})\right] \ ,$$

where $f(m; \delta_1, \ldots, \delta_N)$ is a function that computes the probability that $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ made at least two successive A_2-calls (among the total m) with the same certificate shape (Tab. 1). For example, if $N = 1$ (we have only one shape) and $m = 2$ then $f(m, \delta) = \delta^2$ and by the Jensen inequality $\delta' \gtrapprox \mathop{\mathbf{E}}_{h,r,a}\left[\delta^2_{h,r,a}\right] \geq \left(\mathop{\mathbf{E}}_{h,r,a}[\delta_{h,r,a}]\right)^2 = \delta^2$.

4.1 Tightness Measure for Security Reductions

In order to compare the efficiency of adversaries with different running time and success probability, Luby [8] introduced *time-success ratio* $\frac{t}{\delta}$, where t is the running time and δ

Table 1. The success function $f(m; \delta_1, \ldots, \delta_N)$ and its special cases

	$N = 1$	Arbitrary N
$m = 2$	$f(2; \delta) = \delta^2$	$f(2; \delta_1, \ldots, \delta_N) = \delta_1^2 + \ldots + \delta_N^2$
Arbitrary m	$f(m; \delta) =$ $1 - m\delta(1-\delta)^{m-1} - (1-\delta)^m$	$f(m; \delta_1, \ldots, \delta_N) =$ $1 - \sum_{j=0}^{m} \binom{m}{j} j! \sigma_j(\delta_1, \ldots, \delta_N)(1-\delta)^{m-j}$

is the success of the adversary. A cryptographic primitive is said to be S-secure if every adversary has time-success ratio $\frac{t}{\delta} \geq S$. In terms of exact security, this means that the primitive is (t, δ)-secure for every t and δ with $\frac{t}{\delta} \geq S$. Time-success ratio provides a general measure for the tightness of cryptographic reductions. If the time-success ratio $\frac{t'}{\delta'}$ of the constructed adversary (i.e. $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$) is represented as a function $\frac{t'}{\delta'} = F(t, \frac{1}{\delta})$, where t and δ are the running time and the success of the assumed adversary (i.e. $(\mathsf{A}_1, \mathsf{A}_2)$), then the reduction is *tight* if F grows slowly and *loose* if the growth is faster. The reduction is said to be *linear* if $F(a, b) = O(a) \cdot O(b)$, *quadratic* if $F(a, b) = O(a^2) \cdot O(b^2)$, and *polynomial* if $F(a, b) = a^{O(1)} \cdot b^{O(1)}$. The equation $\frac{t'}{\delta'} = F(t, \frac{1}{\delta})$ is also called as the *security loss (formula)* of the reduction.

4.2 Reduction with Quadratic Security Loss

To get a security reduction with quadratic security loss, we take $m = 2$ and use the estimate [1] $f(2; \delta_1, \ldots, \delta_N) \geq N \cdot f(2; \frac{\delta_1 + \ldots + \delta_N}{N})$, and hence by using Jensen inequality

$$\delta' \gtrapprox \mathop{\mathbf{E}}_{h,r,a}\left[f(2; \delta_{h,r,a}^{(1)}, \ldots, \delta_{h,r,a}^{(N)})\right] \geq \mathop{\mathbf{E}}_{h,r,a}\left[N \cdot f\left(2; \frac{\delta_{h,r,a}}{N}\right)\right] = N \cdot \mathop{\mathbf{E}}_{h,r,a}\left[\left(\frac{\delta_{h,r,a}}{N}\right)^2\right] \geq \frac{\delta^2}{N} \ .$$

Such a reduction has the security loss formula $\frac{t'}{\delta'} \approx 2N \cdot \frac{t}{\delta^2}$ and was given in [5].

4.3 Reducing the Power of N

Buldas and Laur [4] used the birthday bound to improve the efficiency of the reduction. Their main idea was to use the collision-finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ with $m = \frac{\sqrt{N}}{\delta}$ instead of $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(2)$. After generating the samples $(x_1, c_1), \ldots, (x_m, c_m)$ and verifying them with Ver_k, the collision finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ has on average $\delta m = \sqrt{N}$ successfully back-dated bit-strings on average. The birthday bound implies that with a probability of roughly $\frac{1}{2}$ we then have two successfully back-dated bit strings with the same shape n.

[1] This holds because $\frac{\delta_1^2 + \ldots + \delta_N^2}{N} \geq \left(\frac{\delta_1 + \ldots + \delta_N}{N}\right)^2$ due to the convexity of the square function.

These can then be used to extract a collision. This idea was made precise in [4] and resulted in a reduction with security loss [2] $\frac{t'}{\delta'} \approx 48\sqrt{N} \cdot \frac{t}{\delta^2}$. Their reduction was the best known for this problem so far.

5 New Reduction

We now establish a power 1.5 reduction by first showing an inefficient reduction and then using combinatorial counting arguments to make it considerably more efficient. Finally, we obtain a reduction with security loss $\frac{t'}{\delta'} = 14\sqrt{N} \cdot \frac{t}{\delta^{1.5}}$. For this, we use $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ with $m = \Theta\left(\sqrt{\frac{N}{\delta}}\right)$. We start from the case $N = 1$ when all certificates have the same shape and we only need two successful A_2 calls to get a collision. If the success of A_2 is δ, the success of $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ is:

$$f(m,\delta) = 1 - m\delta(1-\delta)^{m-1} - (1-\delta)^m \ , \tag{6}$$

where the first negative term is the probability that only one call is successful and the second negative term is the probability that no call was successful. To explain the theoretical obstacles we will meet when going from power 2.0 to power 1.5 reductions, we first show why it is not trivial to construct a linear reduction even for the case $N = 1$.

5.1 Problems with Establishing a Linear Reduction

It might seem that when $N = 1$, it is nearly trivial to construct a linear reduction with security loss $\frac{t'}{\delta'} = c \cdot \frac{t}{\delta}$. One could just take $m = \max\left\{2, \lceil\frac{1}{\delta}\rceil\right\}$, where δ is the success of the back-dating adversary $(\mathsf{A}_1, \mathsf{A}_2)$, and the success δ' of C will be:

$$\delta' \approx f\left(\frac{1}{\delta}, \delta\right) = \begin{cases} 1 - (1-\delta)^{\frac{1-\delta}{\delta}} - (1-\delta)^{\frac{1}{\delta}} & \text{if } \delta < \frac{1}{2} \\ 1 - 2\delta(1-\delta) - (1-\delta)^2 = \delta^2 & \text{if } \delta \geq \frac{1}{2} \end{cases} .$$

It is easy to see that $\lim_{\delta\to 0} f\left(\frac{1}{\delta}, \delta\right) = 1 - 2e^{-1} \approx 0.26424 \geq \frac{1}{4}$ and if the running time of A_2 is t, we seemingly have that the time success ratio of C is $\frac{t'}{\delta'} \approx 4 \cdot \frac{t}{\delta}$.

However, this approach overlooks the fact that h is randomly chosen and therefore the probability δ in (6) depends on particular choices of h and also on the output (r, a) of A_1. This means that the success of C is the mathematical expectation $\underset{h,r,a}{\mathbf{E}}[f(m, \delta_{h,r,a})]$. As f turns out not to be convex, Jensen's inequality cannot be used and the averaging becomes a nontrivial task in which the power of δ necessarily has to increase.

5.2 Tightness Bounds for Security Reductions

It is easy to see that any hash function used in hash-then-publish time-stamping schemes[3] must be *division-resistant* [2], i.e. any poly-time adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ has success:

$$\Pr\left[h \leftarrow \mathfrak{F}_k, (y,a) \leftarrow \mathsf{A}_1(h), x_1 \leftarrow \{0,1\}^k, x_2 \leftarrow \mathsf{A}_2(y,a,x_1) \colon h(x_1\|x_2) = y\right] = k^{-\omega(1)} \ .$$

[2] The larger constant is due to technical reasons and could probably be reduced somewhat.

[3] More precisely, in schemes where the set $\mathcal{N}$ of allowed shapes contains at least one shape that begins with a `0`-bit. In all schemes that are used in practice, this is indeed the case. If for some reasons, all allowed shapes begin with a `1`-bit, then we can show in a similar way that h must satisfy a dual condition with success predicate $h(x_2\|x_1) = y$ instead of $h(x_1\|x_2) = y$.

Indeed, if there is $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$, such that $\textsc{Adv}_k(\mathsf{A}) = \delta$, then we construct $(\mathsf{A}'_1, \mathsf{A}'_2)$ so that A'_1 first calls $(r, a) \leftarrow \mathsf{A}_1$, creates an h-chain $c' = ((s_1, b_1), \ldots, (s_\ell, b_\ell))$ such that $\mathsf{0}b_1 \ldots b_\ell \in \mathcal{N}$, and with output $\mathsf{Chain}_r(c', =)r'$, and outputs (r', a') where $a' = (a, r, c')$. The second stage $\mathsf{A}_2(r', a')$ first generates a random $x \leftarrow \{0{,}1\}^k$, then executes $x_2 \leftarrow \mathsf{A}_2(r, a, x)$ and outputs (x, c), where $c = ((x_2, \mathsf{0}), (s_1, b_1), \ldots (s_\ell, b_\ell))$. It is easy to see that the modified adversary is unpredictable (because x_1 is chosen independent of y and uniformly at random) and breaks the h-based time-stamping scheme in terms of (3) with success δ.

By using oracle separation techniques it has been proved [2] that every black-box security reduction that derives division-resistance from the collision-resistance of the same function is at least a power-1.5 reduction. Hence, power-1.5 black-box reductions are also the best we can get when proving entropy-based security of a hash-then-publish time-stamping scheme from the collision-resistance of the underlying hash function.

5.3 New Reduction: The Case $N = 1$

If $m = \max\{\frac{1}{\sqrt{\delta}}, 2\}$, the success of the generic collision-finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ is:

$$\delta' \gtrapprox \mathop{\mathbf{E}}_{h,r,a}[f(m, \delta_{h,r,a})] = \sum_{h,r,a} \mathsf{Pr}[h, r, a] \cdot f(m, \delta_{h,r,a}) \ . \tag{7}$$

Note that, in general, $\delta' \not\gtrapprox f(m, \mathop{\mathbf{E}}_{h,r,a}[\delta_{h,r,a}]) = f(m, \delta)$ because f is not convex and we cannot apply the Jensen inequality directly. However, $f(m, \delta)$ is convex in the interval $\left[0 \ldots \frac{1}{m-1}\right]$ (Lemma 2 in Appendix A) and lower bounded by the identity function in the interval $\left[\frac{1}{m-1} \ldots 1\right]$ (Lemma 4 in Appendix A). Defining

$$p = \sum_{\substack{h,r,a \\ \delta_{h,r,a} \geq \frac{1}{m-1}}} \mathsf{Pr}[h, r, a] \cdot \delta_{h,r,a} \ ,$$

we estimate the success δ' of the collision-finder as follows:

$$\delta' \gtrapprox \sum_{\substack{h,r,a \\ \delta_{h,r,a} < \frac{1}{m-1}}} \mathsf{Pr}[h, r, a] \cdot f(m, \delta_{h,r,a}) + \sum_{\substack{h,r,a \\ \delta_{h,r,a} \geq \frac{1}{m-1}}} \mathsf{Pr}[h, r, a] \cdot f(m, \delta_{h,r,a}) \geq f(m, \delta - p) + p \ ,$$

where the first sum is lower-bounded by using Lemma 3 of Appendix A. From the observation that $p \geq \frac{\delta}{6}$ or $\delta - p \geq \frac{5\delta}{6}$, and that $f\left(m, \frac{5\delta}{6}\right) \geq \frac{\delta}{6}$ (Appendix B), it follows that $\delta' \gtrapprox \frac{\delta}{6}$. The security loss of the reduction is $\frac{t'}{\delta'} \approx 6 \cdot \frac{t}{\delta^{1.5}}$.

5.4 New Reduction: General Case

We simply use the fact that from $\delta = \delta^{(1)} + \ldots + \delta^{(N)}$ it follows that there is $n \in \mathcal{N}$ such that $\delta^{(n)} \geq \frac{\delta}{N}$. We now take $m = \max\{\frac{1}{\sqrt{\delta^{(n)}}}, 2\}$ and modify the adversary A_2 so that it only outputs (x, c) if $\rho(c) = n$. The success of A is $\delta^{(n)}$ by the defining equation (5). Hence, we reduced the general case to the case $N = 1$ and the success of the collision finder $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ is $\delta' \gtrapprox \frac{\delta^{(n)}}{6} \geq \frac{\delta}{6N}$ and the security loss of the reduction is $\frac{t'}{\delta'} \approx 6 \cdot N^{1.5} \cdot \frac{t}{\delta^{1.5}}$. In the next section, we show that $N^{1.5}$ can actually be reduced to $\sqrt{N}$ which makes our reduction strictly better than the one given in [4].

5.5 New Reduction: Reducing the Power of N

The adversary previously considered only used collisions for the most probable certificate shape. We can get significantly better bounds if we try to take advantage of all possible collisions. We again use $\mathsf{CF}_k^{h,\mathsf{A},\mathsf{T}}(m)$ as our adversary construction. However, we try to bound the success probability δ' of the collision-finder tighter than before. It is clear that the adversary can fail to find a collision only when all the certificates returned by the time-stamping adversary are of different shapes or when two certificates coincide completely. The readers who are not interested in mathematical details of the proof may skip this subsection.

We analyze what happens if the advice a, the hash function h, and the root hash value r for A_1 have been fixed already. Then the probability of all the successfully back-dated certificates having different shapes after m tries is

$$\sum_{k=0}^{m}\binom{m}{k}k!\sigma_k(\delta^{(1)}\ldots\delta^{(N)})(1-\delta)^{m-k}=\sum_{k=0}^{m}\binom{m}{k}\binom{N}{k}k!S_k(\delta^{(1)}\ldots\delta^{(N)})(1-\delta)^{m-k}$$
$$\leq\sum_{k=0}^{m}\binom{m}{k}\binom{N}{k}k!S_1(\delta^{(1)}\ldots\delta^{(N)})^k(1-\delta)^{m-k}=\sum_{k=0}^{m}\binom{m}{k}\frac{N^{\underline{k}}}{N^k}\delta^k(1-\delta)^{m-k}\ , \quad (8)$$

where σ_k is the k-th *elementary symmetric polynomial*, $S_k = \sigma_k/\binom{N}{k}$ and $N^{\underline{k}} = N\cdot(N-1)\cdot\ldots\cdot(N-k+1)$ is the *falling factorial power*. The *MacLaurin's inequality* says that $\sqrt[k]{S_k} \leq \sqrt[l]{S_l}$ whenever $k \geq l$ and $\delta_i \geq 0$. Now note that

$$\delta^k(1-\delta)^{m-k} = \sum_{i=0}^{m-k}(-1)^i\binom{m-k}{i}\delta^{i+k} = \sum_{j=k}^{m}(-1)^{j-k}\binom{m-k}{j-k}\delta^j\ .$$

We plug this into (8), change the order of summation and use $\binom{m}{k}\binom{m-k}{j-k} = \binom{m}{j}\binom{j}{k}$ to get

$$S=\sum_{k=0}^{m}\binom{m}{k}\frac{N^{\underline{k}}}{N^k}\left(\sum_{j=k}^{m}(-1)^{j-k}\binom{m-k}{j-k}\delta^j\right)=\sum_{k=0}^{m}\sum_{j=k}^{m}(-1)^{j+k}\frac{N^{\underline{k}}}{N^k}\binom{m}{k}\binom{m-k}{j-k}\delta^j =$$
$$=\sum_{j=0}^{m}\sum_{k=j}^{m}(-1)^{j+k}\frac{N^{\underline{k}}}{N^k}\binom{m}{j}\binom{j}{k}\delta^j=\sum_{j=0}^{m}(-1)^j\binom{m}{j}\left(\sum_{k=0}^{j}(-1)^k\frac{N^{\underline{k}}}{N^k}\binom{j}{k}\right)\delta^j\ .$$

Computing the first few terms we get $1-\frac{1}{N}\binom{m}{2}\delta^2+\frac{2}{N^2}\binom{m}{3}\delta^3+\frac{3N-6}{N^3}\binom{m}{4}\delta^4+\ldots$. Denote $\phi_n = \sum_{k=0}^{n}(-1)^k\frac{N^{\underline{k}}}{N^k}\binom{n}{k}$ and $\psi_n = \binom{m}{n}|\phi_n|\delta^n$. It turns out that ϕ_n satisfy the recurrence[4] $\phi_{k+1} = \frac{k}{N}(\phi_k - \phi_{k-1})$. Assuming $c_1\sqrt{\frac{N}{\delta}}+1 \leq m \leq c_2\sqrt{\frac{N}{\delta}}$, we get

[4] This recurrence was found using Zeilberger's algorithm [10]. See Appendix C for a proof.

$$\begin{aligned}\psi_{k+1} &= \binom{m}{k+1}\frac{k}{N}\left|\phi_{k-1}-\phi_k\right|\delta^{k+1} \le \binom{m}{k+1}\frac{2k}{N}\max(|\phi_{k-1}|,|\phi_k|)\delta^{k+1} \\ &= \max\left(\frac{2(m-k)(m-k-1)}{(k+1)N}\binom{m}{k-1}|\phi_{k-1}|, \frac{2(m-k)k}{(k+1)N}\binom{m}{k}|\phi_k|\right)\delta^{k+1} \\ &\le \max\left(\frac{c_2^2 N}{N\delta}\psi_{k-1}\delta^2, \frac{2c_2\sqrt{N}}{N\sqrt{\delta}}\psi_k\delta\right) = c_2\sqrt{\delta}\max\left(\frac{2}{\sqrt{N}}\psi_k, c_2\sqrt{\delta}\psi_{k-1}\right) \ .\end{aligned}$$

To simplify further analysis we assume that $N \ge 4$. By noting that $\psi_1 = 0$, we get that $\psi_3 \le c_2\sqrt{\delta}\psi_2$, $\psi_4 \le c_2^2\delta\psi_2$ and in general, $\psi_k \le (c_2\sqrt{\delta})^{k-2}\psi_2$ for all $k \ge 2$ which can be easily verified by induction. Using this, we get a simple bound on the sum of the remaining elements if we assume $c_2\sqrt{\delta} < 1$:

$$\left|\sum_{k=3}^{m}(-1)^k\binom{m}{k}\phi_k\delta^k\right| \le \sum_{k=3}^{m}\psi_k \le \sum_{k=1}^{m-2}(c_2\sqrt{\delta})^k\psi_2 \le \frac{c_2\sqrt{\delta}\psi_2}{1-c_2\sqrt{\delta}} \ .$$

We thus know that the success of the adversary for fixed h, r and a is at least

$$f(N,\delta) \ge \left(1-\frac{c_2\sqrt{\delta}}{1-c_2\sqrt{\delta}}\right)\frac{1}{N}\binom{m}{2}\delta^2 \ge \frac{1-2c_2\sqrt{\delta}}{N(1-c_2\sqrt{\delta})}\frac{c_1^2N}{2\delta}\delta^2 = \frac{c_1^2(1-2c_2\sqrt{\delta})}{2(1-c_2\sqrt{\delta})}\delta \ .$$

We analyze the lower bound described for convexity. Assuming $\frac{c_1}{c_2} = const.$ we can substitute $c_2\sqrt{\delta} = x$ and disregard a constant multiplier to get $x^2\frac{1-2x}{1-x}$ which is easily seen to be convex whenever $x < 1 - \frac{1}{\sqrt[3]{2}} \approx 0.2$. In order to guarantee the convexity of the approximation for f we need to have $c_2\sqrt{\delta} \le 1 - \frac{1}{\sqrt[3]{2}}$ for all possible δ. As $\delta \le 1$, this can easily be achieved by taking $c_2 \le 1 - \frac{1}{\sqrt[3]{2}}$.

Let $\delta_{h,r,a}$ denote the success when h, r and a are fixed and let $\delta = \mathop{\mathbf{E}}_{h,r,a}[\delta_{h,r,a}]$ be the average success. Since f is convex for δ when we fix c_2 as described, we can use the Jensen inequality to get $\bar{f}(N,\delta) = \mathop{\mathbf{E}}_{h,r,a}[f(N,\delta_{h,r,a})] \ge f\left(N, \mathop{\mathbf{E}}_{h,r,a}[\delta_{h,r,a}]\right)$. Thus,

$$\frac{t'}{\delta'} \approx \frac{mt}{\bar{f}(N,\delta)} \le \frac{c_1\sqrt{\frac{N}{\delta}}t}{\frac{c_1^2(1-2c_2\sqrt{\delta})}{2(1-c_2\sqrt{\delta})}\delta} = \frac{2(1-c_2\sqrt{\delta})\sqrt{N}t}{c_1(1-2c_2\sqrt{\delta})\delta^{1.5}} \ .$$

We want to make the bound. Again, assuming $\frac{c_1}{c_2} = const.$ and also that $\delta = N = const.$ we can see that the problem we are facing is equivalent to maximizing $\frac{1-x\sqrt{\delta}}{x(1-2x\sqrt{\delta})}$. The derivative of that function is positive whenever $(\sqrt{\delta}x)^2 - 2\sqrt{\delta}x + 0.5 > 0$. Since $\sqrt{\delta} \le 1$ and $x = c_2 \le 1 - \frac{1}{\sqrt[3]{2}}$ and both are also greater than 0, the derivative is always positive and as such the maximum is achieved when we take $c_2 = 1 - \frac{1}{\sqrt[3]{2}}$.

We now upper bound $\frac{1-c_2\sqrt{\delta}}{(1-2c_2\sqrt{\delta})}$. As the function is strictly increasing for c_2 fixed to $1-\frac{1}{\sqrt[3]{2}}$ and $\sqrt{\delta} \leq 1$, the upper bound is achieved when $\delta = 1$ when the result is $\frac{1}{2-\sqrt[3]{2}} < 1.4$. Taking $c_1 = 0.2$ then gives $\frac{t'}{\delta'} \approx 14\frac{\sqrt{N}t}{\delta^{1.5}}$.

6 Practical Implications

In order to show the practical consequences of the new reduction we will compare three reductions: the reduction given by Buldas and Saarepera in Asiacrypt 2004 [5], the reduction by Buldas and Laur in PKC 2007 [4], and the new reduction given in this article. We study a hypothetic global scale time-stamping system capable of issuing 67 million (about 2^{26}) time stamps per second and with lifetime at least 34 years (about 2^{30} seconds), i.e. we need to take $N = 2^{56}$. Systems of that scale are indeed in practical use. Our security proof is the first practical statement about the security of such systems if a 256 bit hash function (such as SHA2-256) is used. This is because we want the system to be secure against back-dating adversaries with time-success ratio $t/\delta = 2^{64}$. We study adversaries with three different time-success profiles: $(t,\delta) \in \{(1, 2^{-64}), (2^{32}, 2^{-32}), (2^{64}, 1)\}$. For each profile and reduction we compute the necessary output length of the hash function that is used in the time-stamping system considering that the hash function's security is near the birthday barrier, i.e. hash functions of output size k are $2^{k/2}$-secure. The results are presented in Table 2.

Table 2. Efficiency of reductions. The numbers denote hash function's output size in bits.

Reduction	Formula	$t=1, \delta=2^{-64}$	$t=2^{32}, \delta=2^{-32}$	$t=2^{64}, \delta=1$
Asiacrypt 2004	$\frac{t'}{\delta'} \approx 2N\frac{t}{\delta^2}$	370	306	242
PKC 2007	$\frac{t'}{\delta'} \approx 48\sqrt{N}\frac{t}{\delta^2}$	324	260	196
This paper	$\frac{t'}{\delta'} \approx 14\sqrt{N}\frac{t}{\delta^{1.5}}$	256	224	190

We see that a 256-bit hash function is indeed sufficient for such a time-stamping scheme though the previously proposed reductions were incapable of showing this.

It is also interesting to analyze how the hash-function output size k depends on the capacity N and the required security of the time-stamping system against back-dating. We study two levels of security: against 2^{64}-adversaries and against 2^{80}-adversaries. The results are summarized in Table 3. For example, in order to construct a 2^{64}-secure

Table 3. Efficiency of reductions. How hash function output size k depends on the capacity N.

Reduction	Formula	2^{64}-security	2^{80}-security
Asiacrypt 2004	$\frac{t'}{\delta'} \approx 2N\frac{t}{\delta^2}$	$k = 2\log_2 N + 258$	$k = 2\log_2 N + 322$
PKC 2007	$\frac{t'}{\delta'} \approx 48\sqrt{N}\frac{t}{\delta^2}$	$k = \log_2 N + 268$	$k = \log_2 N + 332$
This paper	$\frac{t'}{\delta'} \approx 14\sqrt{N}\frac{t}{\delta^{1.5}}$	$k = \log_2 N + 200$	$k = \log_2 N + 248$

time-stamping system with total capacity $N = 2^{56}$, we need a 256-bit hash function. Unfortunately, for achieving 2^{80}-security with a 256-bit hash function the capacity should be $N \leq 2^8 = 256$, which is clearly insufficient for a global scale time-stamping system. As the reduction we have is asymptotically tight, we have almost no hope of improving the efficiency of the reduction. Hence, in order to draw practical security conclusions about the large time-stamping systems that use a 256-bit hash function, we are forced to use security assumptions stronger than collision-freeness, even if the function is assumed to be collision-free to the birthday barrier.

References

1. Bayer, D., Haber, S., Stornetta, W.-S.: Improving the efficiency and reliability of digital timestamping. In: Sequences II: Methods in Communication, Security, and Computer Science, pp. 329–334. Springer, Heidelberg (1993)
2. Buldas, A., Jürgenson, A., Niitsoo, M.: Efficiency bounds for adversary constructions in black-box reductions. In: Boyd, C., González Nieto, J. (eds.) ACISP 2009. LNCS, vol. 5594, pp. 264–275. Springer, Heidelberg (2009)
3. Buldas, A., Laur, S.: Do broken hash functions affect the security of time-stamping schemes? In: Zhou, J., Yung, M., Bao, F. (eds.) ACNS 2006. LNCS, vol. 3989, pp. 50–65. Springer, Heidelberg (2006)
4. Buldas, A., Laur, S.: Knowledge-binding commitments with applications in time-stamping. In: Okamoto, T., Wang, X. (eds.) PKC 2007. LNCS, vol. 4450, pp. 150–165. Springer, Heidelberg (2007)
5. Buldas, A., Saarepera, M.: On provably secure time-stamping schemes. In: Lee, P.J. (ed.) ASIACRYPT 2004. LNCS, vol. 3329, pp. 500–514. Springer, Heidelberg (2004)
6. Haber, S., Stornetta, W.-S.: How to time-stamp a digital document. Journal of Cryptology 3(2), 99–111 (1991)
7. Haber, S., Stornetta, W.-S.: Secure names for bit-strings. In: ACM Conference on Computer and Communications Security, pp. 28–35 (1997)
8. Luby, M.: Pseudorandomness and Cryptographic Applications. Princeton University Press, Princeton (1996)
9. Merkle, R.C.: Protocols for public-key cryptosystems. In: Proceedings of the 1980 IEEE Symposium on Security and Privacy, pp. 122–134 (1980)
10. Petkovšek, M., Wilf, H.S., Zeilberger, D.: A=B. A.K. Peters, Ltd, Wellesley (1996)
11. Simon, D.: Finding Collisions on a One-Way Street: Can secure hash functions be based on general assumptions? In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 334–345. Springer, Heidelberg (1998)

A Properties of the Success Function *f*

We prove some useful properties of $f(m, x) = 1 - mx(1-x)^{m-1} - (1-x)^m$.

Lemma 1. *If* $m \geq 2$*, then the function* $f(m, x)$ *is increasing in* $[0 \ldots 1]$.

Proof. This follows from the observation that $\frac{d}{dx} f(m, x) = m(m-1)x(1-x)^{m-2}$ is always positive in $x \in [0 \ldots 1]$. □

Lemma 2. *If $m \geq 2$, then the function $f(m,x)$ is convex in $\left[0 \ldots \frac{1}{m-1}\right]$ and concave in $\left[\frac{1}{m-1} \ldots 1\right]$.*

Proof. We use zeroes of the second derivative of $f(m,x)$. The equation

$$\frac{d^2}{dx^2} f(m,x) = -m(m-1)(1-x)^{m-3}[(m-1)x - 1] = 0$$

implies that $x \in \left\{\frac{1}{m-1}, 1\right\}$. It is easy to see by using direct computations that the second derivative is positive if $0 \leq x \leq \frac{1}{m-1}$ and negative if $\frac{1}{m-1} \leq x \leq 1$. □

Lemma 3. *For every $m \geq 2$, for every collection of points $x_1, \ldots, x_n \in \left[0 \ldots \frac{1}{m-1}\right]$ and coefficients $p_1, \ldots, p_n$ so that $\sum_i p_i \leq 1$ we have*

$$\sum_{i=1}^{n} p_i \cdot f(m, x_i) \geq f\left(m, \sum_{i=1}^{n} p_i \cdot x_i\right) \ .$$

Proof. We use the fact that $f(m,0) = 0$, add an artificial term to the sum, and use the convexity of $f(m,x)$ and apply the Jensen's inequality. Let $p_0 = 1 - \sum_i p_i$ and $x_0 = 0$. Then we have:

$$\begin{aligned}\sum_{i=1}^{n} p_i \cdot f(m,x_i) &= p_0 \cdot f(m,x_0) + \sum_{i=1}^{n} p_i \cdot f(m,x_i) \geq f\left(m,\ p_0 \cdot x_0 + \sum_{i=1}^{n} p_i \cdot x_i\right) \\ &= f\left(m, \sum_{i=1}^{n} p_i \cdot x_i\right) \ ,\end{aligned}$$

which proves the claim. □

Lemma 4. *For every $m \geq 2$ and $x \geq \frac{1}{m-1}$ we have $f(m,x) \geq x$.*

Proof. It is sufficient to prove that $f\left(m, \frac{1}{m-1}\right) \geq \frac{1}{m-1}$ for every $m \geq 2$ and then use the fact that $f(m,x)$ is concave in $\left[\frac{1}{m-1} \ldots 1\right]$. Indeed, $f\left(2, \frac{1}{1}\right) = \frac{1}{1}$, $f\left(3, \frac{1}{2}\right) = \frac{1}{2}$, and $f\left(4, \frac{1}{3}\right) = \frac{11}{27} \geq \frac{1}{3}$. If $m \geq 5$ then

$$\begin{aligned} f\left(m, \frac{1}{m-1}\right) &= 1 - \frac{m}{m-1} \cdot \left(1 - \frac{1}{m-1}\right)^{m-1} - \left(1 - \frac{1}{m-1}\right)^{m} \\ &= 1 - \left(\frac{m+1}{m-1} - 1\right) \cdot \left(1 - \frac{1}{m-1}\right)^{m-1} \geq 1 - 2 \cdot \left(1 - \frac{1}{m-1}\right)^{m-1} \\ &= 1 - 2e^{-1} \geq \frac{1}{4} \geq \frac{1}{m-1} \ . \end{aligned}$$

As $f(m,1) = 1$ and f is concave in $\left[\frac{1}{m-1} \ldots 1\right]$, we have $f(m,x) \geq x$, $\forall x \in \left[\frac{1}{m-1} \ldots 1\right]$. □

B Lower Bound for $f(\max\left\{2, \frac{1}{\sqrt{\delta}}\right\}, \frac{5\delta}{6})$

Theorem 1. *For every* $0 \le \delta \le \frac{1}{4}$ *we have* $f(\max\left\{2, \frac{1}{\sqrt{\delta}}\right\}, \frac{5\delta}{6}) \ge \frac{\delta}{6}$.

Lemma 5. *If* $0 \le x \le \frac{1}{m-1}$, *then* $f(m,x) \ge \frac{m(m-1)}{2}x^2 - \frac{m(m-1)(m-2)}{3}x^3$.

Proof. First, we expand $f(m,x)$ as follows:

$$\begin{aligned}
f(m,x) &= 1 - mx(1-x)^{m-1} - (1-x)^m = 1 - mx\sum_{i=0}^{m-1}(-1)^i\binom{m-1}{i}x^i - \sum_{i=0}^{m}(-1)^i\binom{m}{i}x^i \\
&= 1 + \sum_{i=0}^{m-1}(-1)^{i+1}m\binom{m-1}{i}x^{i+1} - \sum_{i=0}^{m}(-1)^i\binom{m}{i}x^i \\
&= 1 + \sum_{i=1}^{m}(-1)^i m\binom{m-1}{i-1}x^i - \sum_{i=0}^{m}(-1)^i\binom{m}{i}x^i = \sum_{i=1}^{m}(-1)^i\left[m\binom{m-1}{i-1} - \binom{m}{i}\right]x^i \\
&= \sum_{i=1}^{m}(-1)^i m\left(1-\frac{1}{i}\right)\binom{m-1}{i-1}x^i = \sum_{i=2}^{m}(-1)^i m\left(1-\frac{1}{i}\right)\binom{m-1}{i-1}x^i
\end{aligned}$$

Obviously, $a_i = m\left(1-\frac{1}{i}\right)\binom{m-1}{i-1} > 0$ and if $x < \frac{1}{m-1}$ and $2 \le i < m$ then

$$\begin{aligned}
\frac{a_i x^i}{a_{i+1}x^{i+1}} &= \frac{1}{x}\frac{\left(1-\frac{1}{i}\right)\binom{m-1}{i-1}}{\left(1-\frac{1}{i+1}\right)\binom{m-1}{i}} = \frac{1}{x}\frac{\frac{i-1}{i}\binom{m-1}{i-1}}{\frac{i}{i+1}\binom{m-1}{i}} = \frac{1}{x}\frac{i^2-1}{i^2}\frac{\binom{m-1}{i-1}}{\binom{m-1}{i}} \\
&= \frac{1}{x}\frac{i^2-1}{i^2}\frac{\frac{(m-1)!}{(i-1)!(m-i)!}}{\frac{(m-1)!}{i!(m-i-1)!}} = \frac{1}{x}\frac{i^2-1}{i^2}\frac{i!(m-i-1)!}{(i-1)!(m-i)!} = \frac{1}{x}\frac{i^2-1}{i^2}\frac{i}{m-i} \\
&= \frac{1}{x}\frac{i^2-1}{i(m-i)} = \frac{1}{x}\cdot\left(1+\frac{1}{i}\right)\frac{i-1}{m-i} \ge \frac{m-1}{m-i}\cdot(i-1) > 1 \ .
\end{aligned}$$

Therefore, the expansion of $f(m,x)$ when $x \le \frac{1}{m-1}$ is an alternating sum of decreasing terms. This means that

$$\begin{aligned}
f(m,x) \ge p(m,x) &= m\left(1-\frac{1}{2}\right)\binom{m-1}{1}x^2 - m\left(1-\frac{1}{3}\right)\binom{m-1}{2}x^3 \\
&= \frac{m(m-1)}{2}x^2 - \frac{m(m-1)(m-2)}{3}x^3 \ .
\end{aligned}$$

□

Lemma 6. *If* $m = \frac{1}{\sqrt{\delta}}$ *and* $0 < \delta < 1$ *then* $\frac{5\delta}{6} \le \frac{1}{m-1}$.

Proof. $\frac{1}{m-1} = \frac{1}{\frac{1}{\sqrt{\delta}}-1} = \frac{\sqrt{\delta}}{1-\sqrt{\delta}} = \frac{\delta}{\sqrt{\delta}-\delta} \ge \frac{\delta}{1} \ge \frac{5\delta}{6}$. □

Lemma 7. *The polynomial* $h(\delta) = \frac{1}{\delta}\cdot p(\frac{1}{\sqrt{\delta}}, \frac{5\delta}{6})$ *is decreasing in* $[0\ldots 1]$.

Proof. As $h(\delta) = \frac{25}{72} - \frac{175}{324}\sqrt{\delta} + \frac{125}{216}\delta - \frac{125}{324}\delta^{3/2}$ and the equation $\frac{d}{dx}h(x) = 0$ has no real solutions and $\lim_{x\to\infty} h(x) = -\infty$ we conclude that $h(x)$ is decreasing in $[0\ldots\infty)$ and $h(\delta)$ is decreasing in $[0\ldots 1]$. □

Therefore, the global minimum of $h(\delta)$ in $[0\ldots 1/4]$ is $h(1/4) = \frac{25}{144} > \frac{1}{6}$. The function $\frac{1}{\delta}\cdot f(\max\left\{2, \frac{1}{\sqrt{\delta}}\right\}, \frac{5\delta}{6})$ is increasing in $[1/4\ldots 1]$ because then $\max\left\{2, \frac{1}{\sqrt{\delta}}\right\} = 2$ and $\frac{1}{\delta}\cdot f(2,\delta) = \delta$ is increasing. Hence, $f(\max\left\{2, \frac{1}{\sqrt{\delta}}\right\}, \frac{5\delta}{6})$ is lower-bounded by $\frac{\delta}{6}$.

C Proof of the Recurrence Relation

Lemma 8. *Stirling numbers of first kind $s(n,m)$ satisfy the following identity ($\forall m,n$):*

$$\sum_{k=0}^{n+1}(-1)^k s(k, m+k-n-1)\binom{n+1}{k} = \sum_{k=0}^{n}(-1)^k k\cdot s(k, m+k-n)\binom{n}{k} \quad . \tag{9}$$

Proof. We use the recurrence relation $s(a,b-1)-s(a+1,b) = a\cdot s(a,b)$ and transform the left hand side sum ℓ of (9) as follows:

$$\begin{aligned}
\ell &= \sum_{k=0}^{n+1}(-1)^k s(k,m+k-n-1)\binom{n+1}{k}\\
&= s(0,m-n-1)\binom{n+1}{0} + \sum_{k=1}^{n}(-1)^k s(k,m+k-n-1)\binom{n+1}{k} +\\
&\qquad\qquad + (-1)^{n+1}s(n+1,m)\binom{n+1}{n+1}\\
&= s(0,m-n-1)\binom{n}{0} + \sum_{k=1}^{n}(-1)^k s(k,m+k-n-1)\left[\binom{n}{k}+\binom{n}{k-1}\right] +\\
&\qquad\qquad + (-1)^{n+1}s(n+1,m)\binom{n}{n}\\
&= s(0,m-n-1)\binom{n}{0} + \sum_{k=1}^{n}(-1)^k s(k,m+k-n-1)\binom{n}{k} +\\
&\qquad + \sum_{k=1}^{n}(-1)^k s(k,m+k-1-n)\binom{n}{k-1} + (-1)^{n+1}s(n+1,m)\binom{n}{n}\\
&= \sum_{k=0}^{n}(-1)^k s(k,m+k-n-1)\binom{n}{k} + \sum_{k=0}^{n}(-1)^{k+1}s(k+1,m+k-n)\binom{n}{k}\\
&= \sum_{k=0}^{n}(-1)^k\left[s(k,m+k-n-1) - s(k+1,m+k-n)\right]\binom{n}{k}\\
&= \sum_{k=0}^{n}(-1)^k k\cdot s(k,m+k-n)\binom{n}{k} \ ,
\end{aligned}$$

which is equal to the right hand side of (9). □

Theorem 2. *The sequence* $\phi_n = \sum_{k=0}^{n}(-1)^k \frac{N^{\underline{k}}}{N^k}\binom{n}{k}$ *satisfies the recurrence relation:*

$$\phi_{n+1} = \frac{n}{N}(\phi_n - \phi_{n-1}) \ .$$

Proof. It is sufficient to show that $A(N) = N^{n+1}\phi_{n+1}$ and $B(N) = nN^n(\phi_n - \phi_{n-1})$ are identical as polynomials with variable N, i.e. all their coefficients coincide. We use the formula $N^{\underline{m}} = \sum_{j=0}^{m} s(m,j) \cdot N^j$, where $s(m,j)$ are Stirling numbers of the first kind. So, we have:

$$A(N) = \sum_{k=0}^{n+1}(-1)^k N^{\underline{k}} N^{n+1-k}\binom{n+1}{k} = \sum_{k=0}^{n+1}\sum_{j=0}^{k}(-1)^k s(k,j)\binom{n+1}{k} N^{n+1+j-k} \ ,$$

from which it follows that the coefficient $\mathsf{coef}_m(A)$ of N^m is:

$$\mathsf{coef}_m(A) = \sum_{k=0}^{n+1}(-1)^k s(k, m+k-n-1)\binom{n+1}{k} \ ,$$

which is equal to the left hand side of identity (9). Similarly, for $B(N)$ we obtain:

$$\begin{aligned}
B(N) &= nN^n(\phi_n - \phi_{n-1}) \\
&= \sum_{k=0}^{n}(-1)^k N^{\underline{k}} N^{n-k} n\binom{n}{k} - \sum_{k=0}^{n-1}(-1)^k N^{\underline{k}} N^{n-k} n\binom{n-1}{k} \\
&= \sum_{k=0}^{n-1}(-1)^k N^{\underline{k}} N^{n-k} n\left[\binom{n}{k} - \binom{n-1}{k}\right] + (-1)^n N^{\underline{n}} N^0 n\binom{n}{n} \\
&= \sum_{k=0}^{n-1}(-1)^k N^{\underline{k}} N^{n-k} n\binom{n-1}{k-1} + (-1)^n N^{\underline{n}} N^0 n\binom{n}{n} \\
&= \sum_{k=0}^{n-1}(-1)^k N^{\underline{k}} N^{n-k} k\binom{n}{k} + (-1)^n N^{\underline{n}} N^0 n\binom{n}{n} \\
&= \sum_{k=0}^{n}(-1)^k N^{\underline{k}} N^{n-k} k\binom{n}{k} = \sum_{k=0}^{n}\sum_{j=0}^{k}(-1)^k k \cdot s(k,j)\binom{n}{k} \cdot N^{n-k+j}
\end{aligned}$$

and

$$\mathsf{coef}_m(B) = \sum_{k=0}^{n}(-1)^k k \cdot s(k, m+k-n)\binom{n}{k} \ ,$$

which coincides with the right hand side of (9). Hence, $\mathsf{coef}_m(A) = \mathsf{coef}_m(B)$ for every $m > 0$, and by Lemma 8 the statement follows. □