Knowledge-Binding Commitments with Applications in Time-Stamping

Ahto Buldas[1,2,3,⋆] and Sven Laur[4,⋆⋆]

[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. Ahto.Buldas@ut.ee
[4] Helsinki University of Technology, Laboratory for Theoretical Computer Science, P.O.Box 5400, FI-02015 TKK, Finland. slaur@tcs.hut.fi

Abstract. We prove in a non-black-box way that every bounded list and set commitment scheme is *knowledge-binding*. This is a new and rather strong security condition, which makes the security definitions for time-stamping much more natural compared to the previous definitions, which assume *unpredictability* of adversaries. As a direct consequence, list and set commitment schemes with partial opening property are sufficient for secure time-stamping if the number of elements has an explicit upper bound N. On the other hand, white-box reductions are in a sense strictly weaker than black-box reductions. Therefore, we also extend and generalize the previously known reductions. The corresponding new reductions are $\Theta(\sqrt{N})$ times more efficient, which is important for global-scale time-stamping schemes where N is very large.

1 Introduction

Commitment schemes are basic building blocks in numerous cryptographic protocols. The most important properties of commitment schemes are binding and hiding. A commitment is hiding if it reveals no information about the committed message and binding if it is impossible to change the committed message afterwards without detection. First such schemes for committing a single bit were proposed by Blum [4] and by Brassard *et al* [5] and were proven secure under the hardness of factoring assumption. Later works have significantly improved their efficiency and weakened the underlying complexity theoretic assumptions, see [14, 10] for further references. Here, we study the so called *partially releasable* commitments, in which one can compute a commitment (also called *digest*) for a list $\mathcal{X} = (x_1, \ldots, x_N)$ of bit-strings, so that it is possible to partially open the commitment for every $x_i \in \mathcal{X}$ without disclosing the other elements of $\mathcal{X}$. For opening x_i it is sufficient to present a decommitment string s_i (also called *certificate*). Achieving the hiding property is somewhat trivial, as one can always add another layer of commitments. Hence, our main emphasis is on the binding property. List commitments [3, 1, 17] that are only binding are known as *one-way accumulators*.

⋆ Partially supported by Estonian SF grant no. 6944, and by EU FP6-15964: "AEOLUS".
⋆⋆ Partially supported by Finnish Academy of Sciences, and by Estonian Doctoral School in Information and Communication Technologies.

In particular, we analyze the security of a *time-stamping* protocol, where clients send their requests $x_1, \ldots, x_N$ to a Time-Stamping Server (TSS) who computes the commitment c and sends the corresponding certificates $s_1, \ldots, s_N$ back to the clients. If c is published in an authentic way then everybody can verify that x_i was generated before c was published. This principle is used in practical time-stamping schemes [12] where c is computed as the root of a hash tree. List commitment schemes were believed to be exactly what one needs for such kind of time-stamping. However, Buldas *et al* [7] pointed out a flaw in the security proof of [12]. By giving a carefully crafted oracle separation they showed that pure collision-resistance is insufficient to prove that the hash tree time-stamping schemes [12] are secure. In other words, either there are collision-resistant functions that are still insecure for time-stamping, or the security of time-stamping schemes follows from currently unknown complexity-theoretic results. The key point of this paradoxical result is that the number of committed elements is potentially unbounded. In Sec. 4, we prove that all list and set commitments, where the cardinality of $\mathcal{X}$ has an explicit bound $|\mathcal{X}| \leq N$, are suitable for time-stamping. The proof is given in the exact security framework and is $\Theta(\sqrt{N})$ times more efficient than the previous reduction [7]. This improvement is especially valuable for global-scale time-stamping schemes in which N is very large.

In Sec. 5, we show that all binding bounded list and set commitments are *knowledge-binding*. This is a new and extremely strong security requirement inspired from the security of time-stamping schemes. Its strength is comparable to the *plaintext awareness* property, which is defined for public key encryption. The knowledge-binding property is also much more intuitive requirement for time-stamping schemes than the previous ones [7, 9], which use unpredictable probability distributions to model the stream of "new documents" sent to a TSS. Roughly, the knowledge-binding property states that for every efficient TSS, it is possible (by observing the commitment procedure) to efficiently extract the list $\mathcal{X}$ of all documents that can be opened by the TSS in the future. The dedicated extractor must know only the internal coin tosses of TSS and some public parameters. Consequently, even if the TSS is malicious, it must *know* the whole list $\mathcal{X}$ before the corresponding commitment is published. This allows to prove the security in the classical *ideal vs real world* comparison framework [11, pp.622–631,697–700].

Moreover, the notion of knowledge-binding commitments can be useful in other cryptographic protocols, because the ability to open a commitment does not change in time and we may skip the proofs of knowledge in the commitment phase. On the other hand, the corresponding security proofs are not black box. This means that once we have an efficient adversary A that breaks the knowledge-binding condition *we know that there exists* an efficient adversary A' that breaks the binding property of the corresponding commitment scheme. However, we may have no efficient ways to construct A'. Therefore, in reality the knowledge-binding property can be violated but the commitment scheme may still be practically binding—the efficient breaking procedure exists but is not known. Black-box security proofs in turn give an efficient procedure for constructing A' from A. In this sense, Theorems 1–4 give substantially stronger security guarantees for a fixed hash function (e.g. SHA-1) than Theorems 5 and 6.

In Sec. 6, we briefly discuss about other possible applications of knowledge-binding such as distributed and fine-grained time-stamping.

Some of the details of this work have been omitted because of space limitations. The missing details will be published in the IACR ePrint Archive.

2 Preliminaries and Notation

We use a non-uniform model of computations, where each algorithm A is specified as an input of a universal multi-tape Turing machine U that first copies the code of A to its working-tape and then starts to interpret it. A is a *t-time algorithm* if U performs at most t elementary operations to interpret the code of A independent of the input of A.

By $x \leftarrow \mathcal{D}$ we mean that x is chosen randomly according to a distribution $\mathcal{D}$. In particular, if A is an algorithm, then $x \leftarrow \mathsf{A}(y)$ means that x is chosen according to the output distribution of A on an input y. Finite sets are identified with the corresponding uniform distributions, e.g., $x \leftarrow \{0,1\}^\ell$ means that x is a uniformly chosen ℓ-bit string. If $\mathcal{D}_1, \ldots, \mathcal{D}_m$ are distributions and $F(x_1, \ldots, x_m)$ is a predicate, then $\Pr[x_1 \leftarrow \mathcal{D}_1, \ldots, x_m \leftarrow \mathcal{D}_m : F(x_1, \ldots, x_m)]$ denotes the probability that $F(x_1, \ldots, x_m)$ is true after the ordered assignment of $x_1, \ldots, x_m$.

By a *cryptographic primitive* $\mathfrak{P}$ we mean a set of computable functions associated with the advantage function $\mathrm{Adv}_{\mathfrak{P}}(\cdot)$, such that for every adversarial algorithm A, the advantage $\mathrm{Adv}_{\mathfrak{P}}(\mathsf{A})$ is a positive real number. Mostly, $\mathrm{Adv}_{\mathfrak{P}}(\mathsf{A})$ is defined as the non-trivial success (scaled probability) in certain game sec that captures the desired properties of $\mathfrak{P}$. A primitive $\mathfrak{P}$ is said to be (t, ε)-secure in terms of sec if $\mathrm{Adv}^{\mathsf{sec}}_{\mathfrak{P}}(\mathsf{A}) \leq \varepsilon$ for every t-time adversary A. For example, by a (t, ε)-secure *collision-resistant hash function* we mean a pair $\mathcal{H} = (\mathsf{Gen}, h)$ of algorithms such that if $\mathsf{pk} \leftarrow \mathsf{Gen}$ is an arbitrary output of the generation function then $h(\mathsf{pk}, \cdot) = h_{\mathsf{pk}}(\cdot)$ is a function of type $\{0,1\}^\ell \to \{0,1\}^m$ where $\ell > m$; and for every t-time adversary A :

$$\mathrm{Adv}^{\mathsf{coll}}_{\mathcal{H}}(\mathsf{A}) = \Pr[\mathsf{pk} \leftarrow \mathsf{Gen}, (x_1, x_2) \leftarrow \mathsf{A}(\mathsf{pk}) : x_1 \neq x_2 \wedge h_{\mathsf{pk}}(x_1) = h_{\mathsf{pk}}(x_2)] \leq \varepsilon \enspace .$$

Time-success ratio. Quite often it is suitable for adversaries to find a trade-off between plausible attacking-time t and the corresponding advantage $\varepsilon(t)$ against $\mathfrak{P}$. If the minimum *time-success ratio* for $\mathfrak{P}$ is $\alpha_{\mathfrak{P}}$, then $\varepsilon(t) \leq \frac{t}{\alpha_{\mathfrak{P}}}$ by definition. Often, we cannot estimate anything else about $\mathfrak{P}$ than $\alpha_{\mathfrak{P}}$. Now, any black- or white-box reduction introduces a *change ratio* $\gamma = \frac{\alpha_1}{\alpha_0}$ where α_0 is the time-success ratio of the basic primitive and α_1 is the ratio of the derived primitive, i.e., we have established a new approximate bound $\varepsilon_1(t) \leq \frac{t}{\gamma\alpha_0}$. Therefore, large values of γ provide better approximating bounds.

Sampling bounds. Our proofs use several standard statistical bounds. Let $X_1, \ldots, X_m$ be identically distributed independent zero-one random variables with $\mu = \Pr[X_i = 1]$ and let $X = \sum_{i=1}^m X_i$. Then for any $0 \leq \theta \leq 1$ the Chernoff bounds [13]

$$\Pr[X \leq (1-\theta)\mu m] \leq e^{-\theta^2 m\mu/2} \enspace , \text{ and } \qquad \Pr[X \geq (1+\theta)\mu m] \leq e^{-\theta^2 m\mu/3} \enspace .$$

We also need a Birthday bound to determine the collision probability. Let $Y_1, \ldots, Y_m$ be identically but arbitrarily distributed independent random variables with possible values $\{1, \ldots, N\}$. Then the probability p that all Y_i-s are different satisfies $p \leq e^{-\frac{m(m-1)}{2N}}$. In particular, if $m \geq 1.5\sqrt{N}$ and $N \geq 9$ then $p \leq \frac{1}{2}$.

3 Partially Releasable Commitment Schemes

Set and list commitments. Most commitment schemes for ℓ-bit strings facilitate only complete disclosure of the committed input. In the context of time-stamping, the complete input can be several gigabytes long whereas we actually need to disclose only a few hundred bits. Therefore, we study commitment schemes that facilitate partial disclosure of inputs. *List commitments* are order-preserving: committed strings are ordered tuples. *Set commitments* in turn do not provide any ordering. Like ordinary commitment schemes, these commitments are specified by four basic algorithms: Gen, Com, Cert and Ver. Initialization algorithm Gen generates public parameters pk. Elements $(m_1, \dots, m_n)$ are committed by computing $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(m_1, \dots, m_n)$, where the commitment c is sent to the receiver and d is kept by the sender for later use. To prove that m_i was indeed used to compute the commitment c, the sender generates a certificate[5] $s \leftarrow \mathsf{Cert}_{\mathsf{pk}}(d, m_i)$ the validity of which can be tested with the Ver algorithm.

The commitment scheme is *functional* if for any $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(m_1, \dots, m_n)$ and $s \leftarrow \mathsf{Cert}_{\mathsf{pk}}(d, m_i)$, the verification result $\mathsf{Ver}_{\mathsf{pk}}(c, n, m_i, s) = \mathsf{true}$ with overwhelming probability. For list commitments, the certificate s contains also the exact location i of the decommitted element, denoted as $\mathsf{loc}(s) = i$. We explicitly assume that a decommitment certificate for a set $\mathcal{X} = \{x_1, \dots, x_r\}$ is a union of the corresponding element certificates $s_1, \dots, s_r$ denoted by $s_1 \cup \ldots \cup s_r$. Consequently, certificates can be freely joined together and split into sub-certificates. For many commitment schemes such lists can further be compressed but this is only an implementation detail.

We omit the formal definition of the hiding property, since we study only the features related to the binding property. The binding property is different for set and list commitments. For list commitments, the binding property is violated if an adversary can open the i-th element in two different ways:

$$\mathrm{Adv}^{\mathsf{bind}}(\mathsf{A}) = \Pr\left[\begin{array}{l}\mathsf{pk} \leftarrow \mathsf{Gen},\ (c, n, x_0, s_0, x_1, s_1) \leftarrow \mathsf{A}(\mathsf{pk}) :\\ x_0 \neq x_1 \wedge \mathsf{loc}(s_0) = \mathsf{loc}(s_1)\\ \wedge\, \mathsf{Ver}_{\mathsf{pk}}(c, n, x_0, s_0) = \mathsf{Ver}_{\mathsf{pk}}(c, n, x_1, s_1) = \mathsf{true}\end{array}\right] , \qquad (1)$$

where the probability is taken over the coin tosses of all relevant algorithms. Since certificates are closed under union and there is no ordering for set commitments, the only way to misbehave is to exceed the size of $\mathcal{X}$:

$$\mathrm{Adv}^{\mathsf{bind}}(\mathsf{A}) = \Pr\left[\begin{array}{l}\mathsf{pk} \leftarrow \mathsf{Gen},\ (c, n, \mathcal{X}, s) \leftarrow \mathsf{A}(\mathsf{pk}) :\\ \mathsf{Ver}_{\mathsf{pk}}(c, n, \mathcal{X}, s) = \mathsf{true} \wedge |\mathcal{X}| > n\end{array}\right] , \qquad (2)$$

where $\mathsf{Ver}_{\mathsf{pk}}(c, n, \mathcal{X}, s)$ first splits $\mathcal{X}$ and s into components and then verifies each component $x_i \in \mathcal{X}$ separately by using the corresponding component-certificate $s_i \in s$. We say that the commitment scheme is (τ, ε)-binding if for all τ-time adversaries $\mathrm{Adv}^{\mathsf{bind}}(\mathsf{A}) \leq \varepsilon$. For unbounded adversaries, we speak about *statistical* ε*-binding*.

Note that set and list commitments must explicitly specify the number n of the committed elements. Indeed, if the certificates do not reveal the size of the commitment,

[5] To be precise, Cert should return a vector of certificates for each location of m_i in the list.

a malicious adversary can just hide some committed elements and receivers can never be sure if the commitment is fully opened. A commitment scheme is N*-bounded* if $\mathsf{Ver}_{\mathsf{pk}}(c, n, x, s) = \mathsf{false}$ for all $n > N$.

List commitment schemes that satisfy only the binding properties are known as *one-way accumulators* [1, 3, 17]. One-way accumulators that in addition to positive statements $x \in \mathcal{X}$ also allow to (compactly) prove negative statements $x \notin \mathcal{X}$ are called *undeniable attesters* [6]. The commonly used binding requirement for one-way accumulators is n*-times collision-freeness* [1], which is equivalent to the binding property of set commitments.

Time-stamping schemes. Time-stamping protocols process documents 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 i-th period, a short commitment c_i of the corresponding batch $\mathcal{X}_i$ is published. A document $x \in \mathcal{X}_i$ precedes document y, if there is $j > 0$ such that $y \in \mathcal{X}_{i+j}$. Obviously, for a fixed commitment c_i there must be an efficient way to prove that $x \in \mathcal{X}_i$. However, for documents $y \notin \mathcal{X}_i$ such proofs must be infeasible to create. Note that c_i can be viewed as a classical set or list commitment to the set $\mathcal{X}_i$ and the corresponding proof of $x \in \mathcal{X}_i$ as a certificate. Therefore, time-stamping schemes share the same functionality and algorithmic description as the set and list commitment schemes. Such a structural similarity is indeed remarkable. Still, careful studies of the security requirements reveal considerable differences between time-stamping and commitment schemes. Different security definitions exist for time-stamping schemes [7–9, 12]. In this paper, we adapt the strongest[6] definition [9] for the non-uniform precise security framework with minor modifications in notations.

Formal definitions of time-stamping schemes do not require that n is explicitly given as an argument to the verification algorithm Ver, but negative results in [7] suggest that time-stamping schemes (at least those without additional third parties) must be bounded, i.e., n has to be at least implicitly specified.

Intuitively, time-stamping schemes must be secure against "back-dating" and this itself raises a subtle issue: How to model the future? Most works [7–9] have taken an approach based on computational entropy. Document generation is modeled as an efficient randomized procedure and the security guarantees are given for document distributions with high enough computational entropy. More formally, an adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ is (τ, δ)-*unpredictable* if for every τ-time predictor Π :

$$\mathrm{Adv}^{\mathsf{upr}}_{\mathsf{A}}(\Pi) = \Pr\left[\begin{array}{l}\omega_1 \leftarrow \Omega, \mathsf{pk} \leftarrow \mathsf{Gen}, \hat{x} \leftarrow \Pi(\mathsf{pk}, \omega_1), \\ (c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1), (x, s) \leftarrow \mathsf{A}_2(\phi) : \hat{x} = x\end{array}\right] \leq \delta \ ,$$

where ω_1 denotes the random coins of A_1 and the probability is taken over the coin tosses of all relevant algorithms. The second stage A_2 of the adversary models an efficient document generation (back-dating) procedure.

[6] There exist stronger security definitions for time-stamping schemes with additional (auditing) parties [8]. The main drawback of those schemes is a large amount of extra communication.

Definition 1 (Entropy based security). *A time-stamping scheme is $(t, \tau, \delta, \varepsilon)$-secure if for every (τ, δ)-unpredictable t-time* A :

$$\mathrm{Adv}^{\mathsf{ts}}(\mathsf{A}) = \Pr\left[\begin{matrix}\omega_1 \leftarrow \Omega, \mathsf{pk} \leftarrow \mathsf{Gen}, (c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1), \\ (x, s) \leftarrow \mathsf{A}_2(\phi) : \mathsf{Ver}_{\mathsf{pk}}(c, n, x, s) = \mathsf{true}\end{matrix}\right] \leq \varepsilon \ . \qquad (3)$$

Here, δ quantifies a trivial advantage. Indeed, consider the next adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$:

- $\mathsf{A}_1(\mathsf{pk}; \omega_1)$ computes $(c, d) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\hat{x})$ and the corresponding valid certificate $s \leftarrow \mathsf{Cert}_{\mathsf{pk}}(c, \hat{x})$ and outputs a tuple $(c, 1, (\hat{x}, s))$.
- $\mathsf{A}_2(\hat{x}, s)$ generates a random x so that $x = \hat{x}$ with probability δ, and outputs (x, s).

For every τ the adversary A is (τ, δ)-unpredictable. However, no matter how the time-stamping scheme is defined, the advantage $\mathrm{Adv}^{\mathsf{ts}}(\mathsf{A})$ of A is at least δ. Hence, it is reasonable to assume that $\delta \ll \varepsilon$. Moreover, as $\log \frac{1}{\delta}$ is an upper bound for the computational Rényi entropy, we implicitly assume that the computational Shannon entropy of the future documents is at least $\log \frac{1}{\delta}$ w.r.t. the time-bound τ.

The biggest drawback of the entropy based definition is non-uniformity. The security definition is natural in the polynomial model but has some flaws when adapted to the exact model. It only offers protection against (τ, δ)-unpredictable adversaries! Hence, it does not exclude extremely successful adversaries that are just *not quite so unpredictable*. In theory, a time-stamping scheme could be protected against (τ, δ)-unpredictable adversaries but still be totally insecure against $(\tau, \delta+\delta^{100})$-unpredictable adversaries. This flaw can be fixed by requiring strong uniformity in the definition:

Definition 2 (Black-box security). *A time-stamping scheme is (t, τ, ε)-secure if there exists a τ-time black-box extractor machine $\mathcal{K}$ such that for every t-time* A :

$$\mathrm{Adv}^{\mathsf{ts}}(\mathsf{A}) = \Pr\left[\begin{matrix}\omega_1 \leftarrow \Omega, \mathsf{pk} \leftarrow \mathsf{Gen}, \hat{\mathcal{X}} \leftarrow \mathcal{K}^{\mathsf{A}(\mathsf{pk};\omega_1,\cdot)}(\mathsf{pk}), \\ (c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1),\ (x, s) \leftarrow \mathsf{A}_2(\phi) : \\ (\mathsf{Ver}_{\mathsf{pk}}(c, n, x, s) = \mathsf{true} \wedge x \notin \hat{\mathcal{X}}) \vee |\hat{\mathcal{X}}| > n\end{matrix}\right] \leq \varepsilon \ , \qquad (4)$$

where ω_1 denotes random coins of A_1 and $\mathcal{K}$ gets a black-box access to $\mathsf{A}_1(\mathsf{pk}; \omega_1)$ and $\mathsf{A}_2(\phi; \cdot)$. The working time of $\mathcal{K}^{\mathsf{A}(\mathsf{pk};\omega_1,\cdot)}$ includes the time needed to execute all oracle calls. For list commitments, we treat $\hat{\mathcal{X}}$ as a list and write $x \in \hat{\mathcal{X}}$ iff $x = \hat{\mathcal{X}}[\mathsf{loc}(s)]$.

Intuitively, we state that malicious time-stamping servers cannot issue valid certificates for unknown documents, as there exists a well known algorithm $\mathcal{K}^{\mathsf{A}(\mathsf{pk};\omega_1,\cdot)}$ for efficiently reconstructing the list of all valid documents $\hat{\mathcal{X}}$. This algorithm can be automatically constructed for every t-time adversary.

It is straightforward to see that (t, τ, ε)-secure time-stamping scheme is always $(t, \tau, \delta, \varepsilon + N\delta)$ secure where $N \geq |\ \mathcal{X}\ |$, as one can use $\mathcal{K}$ in prediction. In Sec. 4, we prove that every binding N-bounded list commitment scheme is also a secure time-stamping scheme. Still, there are quantitative differences between these two notions.

Practical constructions based on hash trees. Merkle trees [15] and count-certified hash trees [16] (described below) constructed from collision-resistant hash functions are binding but not hiding even if the hash function is modeled as a random oracle—a release of an element (a leaf node) also reveals one neighboring element (the sibling leaf node). Nevertheless, if we use Merkle trees to compute a short commitment from hiding and binding commitments, we get binding and hiding list and set commitments.

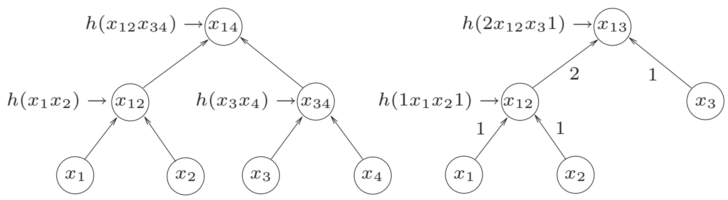

Fig. 1. Merkle hash tree for $\{x_1, x_2, x_3, x_4\}$ and a count-certified hash tree for $\{x_1, x_2, x_3\}$.

A *Merkle hash tree* for a list $\mathcal{X}$ is a binary tree the leaves of which are the elements of $\mathcal{X}$ and each non-leaf node is a hash of its two children (Fig. 1, left). Nodes with a single child can be avoided. Hence, every non-leaf node is assumed to have two children.

A *count-certified hash tree* (Fig. 1, right) is a binary tree which is similar to a Merkle tree, except that its arcs are labeled with *counters* each of which equal to the number of leaves in the corresponding subtree. Each non-leaf vertex v is a hash $h(n_Lx_Lx_Rn_R)$, where n_L and n_R are the counters of the left- and the right subtree respectively. The counter c of the unique outgoing arc of v is the sum $n_v = n_L + n_R$.

Each hash tree can be represented as a commitment function $(c, \mathcal{X}) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\mathcal{X})$, where c is the root hash value of the corresponding tree and pk denotes the public parameters associated with the collision-resistant hash function h. By the certificate $\mathsf{Cert}_{\mathsf{pk}}(\mathcal{X}, x_i)$ for $x_i \in \mathcal{X}$ we mean the smallest amount of data needed to recompute the root hash value. For example, in the Merkle hash tree (Fig. 1, left) the certificate s_2 for x_2 is $s_2 = ((x_1, \_), (\_, x_{34}))$ which represents a sequence of hashing steps starting from the leaf x_2 and ending with the root hash value, whereas $\_$ denotes an *empty slot* which during the verification is filled with the hash of the previous pair. Similarly, in the count-certified hash tree (Fig. 1, right) the certificate for x_2 is $s_2 = ((1, x_1, \_, 1), (2, \_, x_3, 1))$. The verification function $\mathsf{Ver}_{\mathsf{pk}}(c, n, x, s)$ simply recomputes the root hash value by using s and compares it with c. It also checks whether $n \leq N$. The verification algorithm for count-certified trees also recomputes the intermediate counter values to verify the certificate s, in particular if the counter of the root vertex is n.

Collision-Extraction Property. For hash trees with a fixed shape and count-certified hash trees there is a straight and precise reduction of the binding property to the collision-resistance of h because of the following property: If $x_0 \neq x_1$, $\mathsf{Ver}_{\mathsf{pk}}(c, n, x_0, s_0) = \mathsf{Ver}_{\mathsf{pk}}(c, n, x_1, s_1) = \mathsf{true}$, and $\mathsf{loc}(s_0) = \mathsf{loc}(s_1)$, then the internal h-calls of these two verifications comprise a collision for h. Moreover, if the tree is balanced, then the collision can be extracted in $O(|s_0| + |s_1|) = O(\log_2 N)$ time.

1. Execute A_1 in a black-box way and store $(c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1)$.
2. Generate m independent samples $(x_1, s_1) \leftarrow \mathsf{A}_2(\phi), \ldots, (x_m, s_m) \leftarrow \mathsf{A}_2(\phi)$.
3. Output (c, n) and a set of valid pairs $\mathcal{V} = \{(x_i, s_i) : \mathsf{Ver}_{\mathsf{pk}}(c, n, x_i, s_i) = \mathsf{true}\}$.

Fig. 2. Black-box certificate extractor $\mathcal{K}^{\mathsf{A}}_{\mathrm{cert}}(m)$.

4 Bounded Commitments are Sufficient for Time-Stamping

In this section, we prove that bounded commitment schemes with partial opening are sufficient to construct secure time-stamping schemes. The new security reductions use a simple black-box certificate extractor (Fig. 2) and in the proofs we just show that a big enough set of valid decommitments $\mathcal{V}$ allows to break the binding property.

Our proofs do not only generalize the existing ones [7] but are also more efficient. Presented theorems together with the previous separation results [7, 9] provide a clear border between the well studied classical binding properties like collision-freeness and the properties needed for time-stamping. For bounded commitment schemes the binding property implies time-stamping security. Otherwise, these notions are independent—binding properties are not necessary [9] nor sufficient [7].

To clarify the presentation, we have omitted a small $O(N \log N + t)$ term that counts the computational effort needed to manage the list $\mathcal{V}$ of valid decommitments, as the contribution to the total working time is irrelevant for all reasonable values of ε. To be absolutely precise, one has to increase the time-bounds for the binding property by $O(N \log N + t)$ in Theorems 1–4.

Theorem 1 (Entropy based security). *Every $\left(\frac{6t\sqrt{N}}{\varepsilon}, \frac{\varepsilon}{8}\right)$-binding and N-bounded list commitment scheme is also a $\left(t, t, \frac{\varepsilon^3}{432 \cdot N}, \varepsilon\right)$-secure time-stamping scheme for $N \geq 9$.*

Proof. Let $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ be a t-time adversary that violates $\left(t, t, \frac{\varepsilon^3}{432 \cdot N}, \varepsilon\right)$-security promise, i.e., $\mathrm{Adv}^{\mathsf{ts}}(\mathsf{A}) \geq \varepsilon$ and A_2 is sufficiently unpredictable (even for itself):

$$\Pr\left[\mathsf{Coll}\right] := \Pr\left[\begin{array}{l}\mathsf{pk} \leftarrow \mathsf{Gen}, (c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega), \\ (x_0, s_0) \leftarrow \mathsf{A}_2(\phi), (x_1, s_1) \leftarrow \mathsf{A}_2(\phi) : x_0 = x_1\end{array}\right] \leq \frac{\varepsilon^3}{432N} \ .$$

If $m = \frac{6\sqrt{N}}{\varepsilon}$ then the black-box certificate extractor $\mathcal{K}^{\mathsf{A}}_{\mathrm{cert}}(m)$ runs in time $\frac{6t\sqrt{N}}{\varepsilon}$ and provides enough certificates to reveal a double opening. Let Coll^* denote that two equal messages $x_i = x_j$ are produced internally by $\mathcal{K}^{\mathsf{A}}_{\mathrm{cert}}(m)$. Then by the union bound

$$\begin{aligned}\Pr\left[\mathsf{Coll}^*\right] &\leq \sum_{\mathsf{pk}, \omega_1} \Pr\left[\mathsf{pk}, \omega_1\right] \cdot \frac{m(m-1)}{2} \cdot \Pr\left[\mathsf{Coll} | \mathsf{pk}, \omega_1\right] \\ &\leq \frac{m(m-1)}{2} \cdot \Pr\left[\mathsf{Coll}\right] \leq \frac{m^2}{2} \cdot \frac{\varepsilon^3}{432N} \leq \frac{\varepsilon}{24} \ .\end{aligned}$$

Next, we estimate the number of valid document-certificate pairs created by $\mathcal{K}^{\mathsf{A}}_{\mathrm{cert}}(m)$. Let $\varepsilon_{\mathsf{pk}, \omega_1} = \mathrm{Adv}^{\mathsf{ts}}(\mathsf{A} | \mathsf{pk}, \omega_1)$ denote the probability that A is successful for fixed pk

and ω_1. As $\Pr\left[\mathsf{pk} \leftarrow \mathsf{Gen}, \omega_1 \leftarrow \Omega : \varepsilon_{\mathsf{pk},\omega_1} \geq \frac{\varepsilon}{2}\right] \geq \frac{\varepsilon}{2}$, we apply the Chernoff bound for these (pk, ω_1) pairs with $\theta = \frac{1}{2}$ and X_i indicating $(x_i, s_i) \in \mathcal{V}$, and get

$$\Pr\left[|\mathcal{V}| \leq 1.5\sqrt{N} | \varepsilon_{\mathsf{pk},\omega_1} \geq \tfrac{\varepsilon}{2}\right] \leq e^{-\frac{3\sqrt{N}}{8}} < 1/3 \enspace .$$

Since $\mathcal{V}$ consists of identically distributed independent variables, we apply the Birthday bound. If $|\mathcal{V}| \geq 1.5\sqrt{N}$ then $\mathsf{loc}(s_i) = \mathsf{loc}(s_j)$ for some i, j with probability $> \frac{1}{2}$. Let C be an adversary that runs $\mathcal{K}^{\mathsf{A}}_{\text{cert}}(m)$ and then tries to find a double opening in $\mathcal{V}$. Then

$$\mathrm{Adv}^{\mathsf{bind}}(\mathsf{C}) \geq \frac{\varepsilon}{2} \cdot \left(1 - e^{-\frac{3\sqrt{N}}{8}}\right) \cdot \frac{1}{2} - \Pr\left[\mathsf{Coll}^*\right] > \frac{\varepsilon}{6} - \frac{\varepsilon}{24} = \frac{\varepsilon}{8}$$

for $N \geq 9$ and we have obtained a desired contradiction. $\square$

Theorem 2 (Entropy based security). *Every $\left(\frac{4Nt}{\varepsilon}, \frac{\varepsilon}{8}\right)$-binding and N-bounded set commitment scheme is a $\left(t, t, \frac{\varepsilon^3}{64N^2}, \varepsilon\right)$-secure time-stamping scheme for $N \geq 6$.*

Proof. Similarly to the previous proof, let $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ be a t-time adversary that violates a $\left(t, t, \frac{\varepsilon^3}{64N^2}, \varepsilon\right)$-time-stamping security promise. In other words, $\mathrm{Adv}^{\mathsf{ts}}(\mathsf{A}) \geq \varepsilon$ and $\Pr\left[\mathsf{Coll}\right] \leq \frac{\varepsilon^3}{64(N+1)^2}$. Fix $m = \frac{4N}{\varepsilon}$. Then the black-box certificate extractor $\mathsf{C} := \mathcal{K}^{\mathsf{A}}_{\text{cert}}(m)$ then runs in time $\frac{4Nt}{\varepsilon}$. The Chernoff bound with $\theta = \frac{1}{2}$ yields

$$\Pr\left[|\mathcal{V}| \leq N | \varepsilon_{\mathsf{pk},\omega_1} \geq \tfrac{\varepsilon}{2}\right] \leq e^{-\frac{N}{4}} < 1/2 \enspace .$$

Again, $\Pr\left[\mathsf{pk} \leftarrow \mathsf{Gen}, \omega_1 \leftarrow \Omega : \varepsilon_{\mathsf{pk};\omega} \geq \frac{\varepsilon}{2}\right] \geq \frac{\varepsilon}{2}$ and we have obtained a contradiction: $\mathrm{Adv}^{\mathsf{bind}}(\mathsf{C}) \geq \frac{\varepsilon}{2} \cdot \left(1 - e^{-\frac{N}{4}}\right) - \Pr\left[\mathsf{Coll}^*\right] > \frac{\varepsilon}{4} - \frac{m^2}{2} \cdot \frac{\varepsilon^3}{64N^2} = \frac{\varepsilon}{8}$. $\square$

Theorem 3 (Uniform security). *Every $(\frac{2Nt}{\varepsilon}, \frac{\varepsilon}{2})$-binding and N-bounded list commitment scheme is also $(t, \frac{2Nt}{\varepsilon}, \varepsilon)$-black-box secure time-stamping scheme.*

Proof. For the proof we have to fix a canonical black-box extractor machine $\mathcal{K}^{\mathsf{A}}$:

1. First run A_1 and store $(c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1)$ and set $\hat{\mathcal{X}}[i] = \perp$ for $i \in \{1, \ldots, n\}$.
2. Fix $m = \frac{2N}{\varepsilon}$ and for $k \in \{1, \ldots, m\}$ do
   – Compute an independent sample $(x_k, s_k) \leftarrow \mathsf{A}_2(\phi)$.
   – If $\mathsf{Ver}_{\mathsf{pk}}(c, n, x_k, s_k) = \mathsf{true}$ and $\hat{\mathcal{X}}[\mathsf{loc}(s_k)] = \perp$ then set $\hat{\mathcal{X}}[\mathsf{loc}(s_k)] = x_k$.
3. Output the last snapshot of $\hat{\mathcal{X}}$.

Clearly, for every t-time adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$, the extraction algorithm $\mathcal{K}^{\mathsf{A}}$ runs in time $\frac{2Nt}{\varepsilon}$ and the extractor $\mathcal{K}$ is valid for the definition.

For the sake of contradiction, assume that a t-time adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ violates the security promise (4) w.r.t. $\mathcal{K}$. Let a pair (x_k, s_k) be *revealing* if $x_k \neq \hat{\mathcal{X}}[\mathsf{loc}(s_k)]$ in Step 2 of $\mathcal{K}^{\mathsf{A}}$. Then the probability that (x_k, s_k) is revealing must be larger than ε for every $k \in \{1, \ldots, m\}$, since the previous state of $\hat{\mathcal{X}}$ can be viewed as a partial output of $\mathcal{K}^{\mathsf{A}}$. Let X_k be the corresponding zero-one indicator variable, i.e., $X_k = 1$ if (x_k, s_k) is revealing. Then $\varepsilon_k = \mathbf{E}[X_k] > \varepsilon$ and the average of $S_m = \sum_{k=1}^{m} X_k$ is

$$\mathbf{E}[S_m] = \mathbf{E}\left[X_1 + \cdots + X_m\right] = \varepsilon_1 + \cdots \varepsilon_m > m\varepsilon = 2N \enspace .$$

On the other hand, $\mathbf{E}[S_m] \le N + \Pr[S_m > N] \cdot \frac{2N}{\varepsilon}$ and thus $\Pr[S_m > N] > \frac{\varepsilon}{2}$. Therefore, with probability strictly more than $\frac{\varepsilon}{2}$ there are $N+1$ revealing pairs (x_k, s_k) computed by $\mathcal{K}^{\mathsf{A}}$. As the commitment scheme is N-bounded, revealing pairs exist only if $n \le N$. Hence, at least one slot must be overwritten if there are $N+1$ revealing pairs and we have found a double opening with probability strictly more than $\frac{\varepsilon}{2}$. □

Theorem 4 (Uniform security guarantee). *Every* $(\frac{2Nt}{\varepsilon}, \frac{\varepsilon}{2})$*-binding* N*-bounded set commitment scheme is also* $(t, \frac{2Nt}{\varepsilon}, \varepsilon)$*-black-box secure time-stamping scheme.*

Proof. The construction given above is also valid for set commitments. □

Comparison with previous results. Our reductions are not completely novel. A similar proof with a different reduction was given in [7] for hash trees. Therefore, we compare the time-success ratios. Recall that the minimal time-success ratio α implies $\varepsilon(t) \le \frac{t}{\alpha}$ and hence large ratios $\gamma = \frac{\alpha_1}{\alpha_0}$ lead to better security bounds.

In Thm. 1 we constructed a double opener with running time $t_0 \approx \frac{6t\sqrt{N}}{\varepsilon}$ and with advantage $\varepsilon_0 \approx \frac{\varepsilon}{8}$, based on a back-dating adversary with running time t and advantage ε. Thus the change ratio is $\gamma \approx \frac{\varepsilon}{48\sqrt{N}}$ for our reduction. If we adapt the reduction presented in [7] for the exact security model we obtain a ratio $\gamma \approx \frac{\varepsilon}{2N}$, which is significantly smaller for $N \ge 600$. In *global-scale* time-stamping services, N can be very large (say millions or even billions) and our new reduction by far supersedes the previous one [7].

Similarly, one can verify that $\gamma \approx \frac{\varepsilon}{4N}$ for Thm. 3 and Thm. 4 but the security guarantees are much stronger. To break the black-box security an adversary can produce valid document-certificate pairs with low computational Rényi entropy, which makes it impossible to use the birthday paradox. It is easy to see that the extractor must work in time $\Theta(\frac{Nt}{\varepsilon})$ and $\sqrt{N}$ in the denominator is not achievable.

5 All Bounded Commitment Schemes are Knowledge-Binding

Both security definitions for time-stamping (Def. 1,2) are based on heuristic assumptions. Namely, the future is modeled as a *computationally efficient stochastic process*. Such an assumption has two major drawbacks. Firstly, it is philosophically questionable and causes practical problems in the classical framework of secure computations [11]: due to the non-uniform nature of such model, future documents may have arbitrary distributions. Secondly, the success of back-dating adversaries is computed as an average over the distribution of future documents and it might still be easy to “backdate” a fixed document. To overcome these problems, we propose a new security notion where the future is modeled as an advice string that is independent of pk. The independence assumption is essential. Otherwise, no computationally binding commitment scheme can be secure, since the advice may contain explicit double-openings.

Definition 3. *A commitment scheme is (t, τ, ε)-knowledge-binding if for every t-time adversary $\mathsf{A} = (\mathsf{A}_1, \mathsf{A}_2)$ there exist a dedicated τ-time extractor machine $\mathcal{K}_\mathsf{A}$ such that*

$$\mathrm{Adv}^{\mathsf{k\text{-}bind}}(\mathsf{A}) = \max_{\mathsf{adv}} \Pr \left[\begin{array}{l} \mathsf{pk} \leftarrow \mathsf{Gen},\ \omega_1 \leftarrow \Omega,\ \hat{\mathcal{X}} \leftarrow \mathcal{K}_A(\mathsf{pk}; \omega_1), \\ (c, n, \phi) \leftarrow \mathsf{A}_1(\mathsf{pk}; \omega_1), (x, s) \leftarrow \mathsf{A}_2(\phi, \mathsf{adv}) : \\ (\mathsf{Ver}_{\mathsf{pk}}(c, n, x, s) = \mathsf{true} \wedge x \notin \hat{\mathcal{X}}) \vee |\hat{\mathcal{X}}| > n \end{array} \right] \leq \varepsilon \ ,$$

where adv *varies over all advices of length t and the probability is taken over the coins of* Gen, A_1 *and* A_2*. For list commitments, $\hat{\mathcal{X}}$ is a list and write $x \in \hat{\mathcal{X}}$ iff $x = \hat{\mathcal{X}}[\mathsf{loc}(s)]$.*

The new definition explicitly states that there exists an efficient *extraction strategy* $\mathcal{K}_\mathsf{A}$ that is able (by observing the internal computations of the committing algorithm A_1) to predict any bit-string x that is later "back-dated" by A_2. I.e, in some sense x already existed before the commitment and no real back-dating attacks were performed.

But there is an even more intuitive interpretation. When an adversary publishes a commitment c, he implicitly fixes his level of knowledge about the commitment and no future actions can change it. As the level of knowledge does not change in time, a successful opening "proves" that the adversary already "knew" the committed element when the commitment was created. Hence, we can omit proofs of knowledge at the commitment stage and reduce the number of rounds in various protocols. Thus, the new notion is very similar to plaintext-awareness of public-key encryption schemes.

Finally, note that knowledge-binding is a necessary condition for the multi-party security of time-stamping schemes. In the ideal implementation, TSS gives a list $\mathcal{X}$ to a trusted party who will later serve partial release queries $x \in \mathcal{X}$? Hence, there must be an efficient way to extract all documents that TSS can potentially open as a response for any future message that is independent of pk, i.e., the extractor machine $\mathcal{K}_\mathsf{A}$ must exist. To get multi-party security in the malicious model, we must also protect a honest TSS against malicious clients. This can be done in an obvious way by using digital signatures, but due to the space limitations we defer the discussion to follow-up articles.

Clearly, the knowledge-binding property can be established only by using white-box reductions. In other words, we cannot efficiently construct the code of $\mathcal{K}_\mathsf{A}$ given only the code of A, although $\mathcal{K}_\mathsf{A}$ itself is an efficient algorithm. Such reductions provide substantially weaker security guarantees for *fixed hash functions* like SHA-1, since we know *a priori* that efficient collision finders must exist for SHA-1. Therefore, the claims of existence without efficient construction strategies provide no new information. As a result, we can only talk about the security of hash function families, i.e., we have to consider SHA-1 as a "typical" representative of a collision-free hash function family.

The proofs consist of two main steps. First we analyze the behavior of A and construct a dedicated knowledge extractor $\mathcal{K}_\mathsf{A}$. Next we show that $\mathcal{K}_\mathsf{A}$ is efficient and $\mathrm{Adv}^{\mathsf{k\text{-}bind}}(\mathsf{A})$ is sufficiently small. To construct $\mathcal{K}_\mathsf{A}$, we run A on all possible inputs and find suitable triggering messages adv that force A to reveal most of the valid certificates. Next, we construct $\mathcal{K}_\mathsf{A}$ from A and the triggering messages. As the knowledge-binding condition only requires the *existence* of $\mathcal{K}_\mathsf{A}$, the construction time is not an issue.

Theorem 5. *For every $t > 0$ and $\delta > 0$, there exists $\tau = (\frac{N}{\delta} + 1) \cdot O(t)$ such that every (τ, ε)-binding list commitment scheme is $(t, \tau, \varepsilon + \delta)$-knowledge binding.*

Proof. Fix a t-time adversary A and consider a giant status matrix $\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2]$ the rows of which are indexed by public keys pk and random coins ω_1 of A_1, whereas the columns are indexed by t-bit advices adv and random coins ω_2 of A_2. Define

$$\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] = \begin{cases} 0, & \text{if } \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{false} \enspace, \\ \mathsf{loc}(s), & \text{if } \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \enspace, \end{cases}$$

where $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv};\omega_2)$. Note that few columns of W cover most of the rows containing non-zero elements. Namely, Lemma 1 from App. A assures the existence of $\mathcal{I} = \big\{(\mathsf{adv}_1,\omega_2^1),\ldots,(\mathsf{adv}_k,\omega_2^k)\big\}$ such that $|\mathcal{I}| \le \frac{N}{\delta}$ and for any fixed advice-randomness pair (adv,ω_2):

$$\Pr\left[(\mathsf{pk},\omega_1) : 0 \neq \mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] \notin \mathcal{L}[\mathsf{pk},\omega_1] \wedge |\mathcal{L}[\mathsf{pk},\omega_1]| < N\right] \le \delta \enspace, \qquad (5)$$

where $\mathcal{L}[\mathsf{pk},\omega_1] = \{\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] : (\mathsf{adv},\omega_2) \in \mathcal{I}\}$ is a set of revealed locations. Now the construction[7] of $\mathcal{K}_{\mathsf{A}}$ is evident:

1. Given (pk,ω_1) store $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and set $\hat{\mathcal{X}}[i] = \perp$ for $i \in \{1,\ldots,n\}$.
2. For each $(\mathsf{adv},\omega_2) \in \mathcal{I}$ do
   - Compute $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv};\omega_2)$.
   - If $\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true}$ then set $\hat{\mathcal{X}}[\mathsf{loc}(s)] \leftarrow x$.
3. Output the last snapshot of $\hat{\mathcal{X}}$.

To analyze the advantage of $\mathcal{K}_{\mathsf{A}}$, we fix a pair (adv,ω_2). Let $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv};\omega_2)$ as before. For valid decommitment value s, the entry $\hat{\mathcal{X}}[\mathsf{loc}(s)] = \perp$ only if $|\mathcal{L}[\mathsf{pk},\omega_1]| < N$ and thus the inequality (5) given above yields $\Pr\left[(\mathsf{pk},\omega_1) : \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge \hat{\mathcal{X}}[\mathsf{loc}(s)] = \perp\right] \le \delta$. Alternatively, $\mathcal{K}_{\mathsf{A}}$ can fail if $\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true}$ but $\hat{\mathcal{X}}[\mathsf{loc}(s)] \neq x$. However, we can naturally combine A_1, A_2 and $\mathcal{K}_{\mathsf{A}}$ into an adversary B that outputs these double openings and performs $(\frac{N}{\delta}+1)\cdot O(t)$ elementary operations. Consequently, $\mathrm{Adv}^{\mathsf{bind}}(\mathsf{B}) \le \varepsilon$ and thus

$$\Pr\left[(\mathsf{pk},\omega_1) : \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge x \neq \hat{\mathcal{X}}[\mathsf{loc}(s)] \neq \perp\right] \le \varepsilon \enspace .$$

As a result, we have obtained that for any pair (adv,ω_2):

$$\Pr\left[(\mathsf{pk},\omega_1) : \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge x \neq \hat{\mathcal{X}}[\mathsf{loc}(s)]\right] \le \delta + \varepsilon$$

and the claim follows. □

Theorem 6. *For every $t > 0$ and $\delta > 0$, there exists $\tau = (\frac{N}{\delta}+1)\cdot O(t)$ such that every (τ,ε)-binding set commitment scheme is $(t,\tau,\varepsilon+\delta)$-knowledge-binding.*

[7] Note that all elements of the set $\mathcal{I}$ are hardwired as explicit constants into the code of $\mathcal{K}_{\mathsf{A}}$, i.e., $\mathcal{K}_{\mathsf{A}}$ *does not* compute $\mathcal{I}$. As $\mathcal{K}_{\mathsf{A}}$ runs on a universal Turing machine, it must rewind the code of A_2 and thus $\mathcal{K}_{\mathsf{A}}$ performs at most $O(t)$ extra steps to complete the loop of Step 2.

Proof. Fix a t-time adversary A and consider a status matrix $\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2]$ that is indexed identically to the previous proof but the entries are defined differently:

$$\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] = \begin{cases} 0, & \text{if } \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{false} \enspace , \\ x, & \text{if } \mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \enspace , \end{cases}$$

where $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv};\omega_2)$. Then Lemma 1 from App. A assures the existence of $\mathcal{I} = \big\{(\mathsf{adv}_1,\omega_2^1),\ldots,(\mathsf{adv}_k,\omega_2^k)\big\}$ such that $|\mathcal{I}| \leq \frac{N}{\delta}$ and for every fixed advice-randomness pair (adv,ω_2):

$$\Pr\left[(\mathsf{pk},\omega_1) : 0 \neq \mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] \notin \mathcal{L}[\mathsf{pk},\omega_1] \wedge |\mathcal{L}[\mathsf{pk},\omega_1]| < N\right] \leq \delta \enspace , \qquad (6)$$

where $\mathcal{L}[\mathsf{pk},\omega_1] = \{\mathsf{W}[\mathsf{pk},\omega_1;\mathsf{adv},\omega_2] : (\mathsf{adv},\omega_2) \in \mathcal{I}\}$ is a set of revealed elements. Now the construction of $\mathcal{K}_{\mathsf{A}}$ is straightforward:

1. Given (pk,ω_1) store $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and set $\hat{\mathcal{X}} \leftarrow \emptyset$.
2. For each $(\mathsf{adv},\omega_2) \in \mathcal{I}$ do
   - Compute $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv};\omega_2)$.
   - If $\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true}$ then add x to $\hat{\mathcal{X}}$.
3. Output the last snapshot of $\hat{\mathcal{X}}$.

To analyze the advantage of $\mathcal{K}_{\mathsf{A}}$, fix (adv,ω_2). Let $(c,n,\phi) \leftarrow \mathsf{A}_1(\mathsf{pk};\omega_1)$ and $(x,s) \leftarrow \mathsf{A}_2(\phi,\mathsf{adv},\omega_2)$ as before. As $\hat{\mathcal{X}}[\mathsf{pk},\omega_1] = \mathcal{L}[\mathsf{pk},\omega_1]$ by the construction (see Lemma 1), the inequality (6) yields $\Pr\left[\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge x \notin \hat{\mathcal{X}} \wedge |\hat{\mathcal{X}}| < n \leq N\right] \leq \delta$. The extractor $\mathcal{K}_{\mathsf{A}}$ can also fail when $\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true}$ but $x \notin \hat{\mathcal{X}}$ and $|\hat{\mathcal{X}}| \geq n$. Again, we can naturally combine A_1, A_2 and $\mathcal{K}_{\mathsf{A}}$ into an adversary B with running-time $(\frac{N}{\delta}+1)\cdot O(t)$ that runs all algorithms and extracts all valid openings. Consequently, the restriction $\mathrm{Adv}^{\mathsf{bind}}(\mathsf{B}) \leq \varepsilon$ yields $\Pr\left[\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge x \notin \hat{\mathcal{X}} \wedge |\hat{\mathcal{X}}| \geq n\right] \leq \varepsilon$ and we have obtained that for any pair (adv,ω_2):

$$\Pr\left[\mathsf{Ver}_{\mathsf{pk}}(c,n,x,s) = \mathsf{true} \wedge x \notin \hat{\mathcal{X}}\right] \leq \delta + \varepsilon$$

and the claim follows. □

Efficiency of the new reduction. Again, we compute time-success ratios to compare the efficiency of the new white-box reduction to the previous black-box ones. To have a fair comparison we take $\delta \approx \varepsilon$. Then Theorems 5 and 6 provide attacks against the binding property with parameters $t_0 \approx (\frac{N}{\delta}+1)t$ and $\varepsilon_0 = \varepsilon$, provided that there exist a t-time adversary achieving $\varepsilon + \delta$ success. As a result, we obtain a change ratio $\gamma = \frac{\alpha_1}{\alpha_0} \approx (\frac{N}{\delta}+1)^{-1} \cdot \frac{\varepsilon}{\varepsilon+\delta} \approx \frac{\varepsilon}{2N}$, which is better than the change ratio $\gamma \approx \frac{\varepsilon}{4N}$ provided by Thm. 3 and Thm. 4. The difference is not essential rather it comes from slightly loose success bounds in Thm. 3 and Thm. 4.

6 Applications of Knowledge-Binding Commitments

Here, we briefly describe how knowledge-binding count-certified hash trees can be used and why knowledge-binding property is important. Knowledge-binding property can be viewed as an indifference against outside advices. Similar to the plaintext-awareness, the knowledge-binding property allows one to combine commitments with other cryptographic primitives without a fear of unwanted interference. Such interference often makes it hard or impossible to prove the security of new constructions. If the secret or public parameters of other primitives are independent of the commitment parameters pk, then the rest of the protocol can be interpreted as an external advice. Hence, one can use the standard hybrid argument technique even if the primitives are used concurrently.

Distributed and fine-grain time-stamping. Knowledge-binding commitments give rise to a secure time-stamping service where a central time-stamping authority (TSS) computes and publishes the round commitment (c, n) and distributes the respective certificates s_i to the clients. But such service is susceptible to denial-of-service attacks. Hence, it is more natural to consider a distributed service where k independent servers compute sub-commitments (c_i, n_i) and at the end of the round the master commitment (c, n) is compiled. Therefore, it is advantageous to use knowledge-binding commitments that facilitate fast merging of sub-commitments and mostly local certificate computations. Count-certified hash trees have the following important property: every root node (c_i, n_i) of a hash subtree forms a correct commitment. Moreover, given two root nodes (c_L, n_L) and (c_R, n_R) it is straightforward to compute the commitment of the merged tree and update the corresponding certificates.

In a way, a set commitment scheme provides a really coarse-grain time-stamping service. It is impossible to order the events inside the round $\mathcal{X}$. List commitment provides only a partial solution, as clients have to trust that the TSS orders documents correctly in a single round. Tree-shaped list commitments that preserve knowledge-binding w.r.t. the root of each subtree allow also fine-grained time-stamping even if the TSS acts maliciously. Essentially, TSS has to send to a Client all root commitments (c_i, n_i) of all preceding computations, then the Client has strong guarantees that after submitting his query the TSS cannot insert any messages in the prefix of the list without getting caught. Hence, count-certified hash trees could be used for fine-grain time-stamping.

Non-malleable partially releasable commitments. To show that knowledge-binding commitments have other applications outside of the domain of time-stamping, we give a construction of partially releasable non-malleable commitments form non-malleable string commitments and knowledge-binding commitments. It is just an informal example, we do not formalize the claim due to the lack of space.

Recall that a commitment scheme is non-malleable if given a commitment c it is infeasible to construct a new commitment $c' \neq c$ such that after seeing a certificate s for x it is infeasible to output a valid certificate s' for x' such that x and x' are related. Let $\mathcal{L} = \{c_1, \ldots, c_n\}$ be a list of non-malleable commitments for $x_1, \ldots, x_n$ and $(C, D) \leftarrow \mathsf{Com}_{\mathsf{pk}}(\mathcal{L})$ is computed by using a knowledge-binding commitment scheme. Then the resulting commitment scheme is non-malleable. From the knowledge-binding property it follows that after seeing a proof that c_i was computed by using x_i, adversary's ability to output certificates (c, s) such that $\Pr[\mathsf{Ver}(C, n, c, s) = \mathsf{true}]$ does not increase. Hence, the adversary knows all valid commitment-certificate pairs (c_i, s_i) essentially before any commitment is opened. Therefore, non-malleability directly follows from the non-malleability of the lower-level commitment.

References

1. N. Barić and B. Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Proc. of *EUROCRYPT'97*, *LNCS 1233*, pages 480–494, 1997.
2. D. Bayer, S. Haber, and W.-S. Stornetta. Improving the efficiency and reliability of digital time-stamping. In *Sequences II: Methods in Communication, Security, and Computer Science*, pages 329-334, Springer-Verlag, New York 1993.
3. J. Benaloh and M. de Mare. One-way accumulators: a decentralized alternative to digital signatures. In Proc. of *EUROCRYPT'93*, *LNCS 765*, pages 274–285, 1994.
4. M. Blum. Coin flipping by telephone: a protocol for solving impossible problems. In Proc. of *CompCon*, pages 133–137, 1982.
5. G. Brassard, D. Chaum, and C. Crépeau. Minimum disclosure proofs of knowledge. *JCSS*, vol.37, pages 156–189, 1988.
6. A. Buldas, P. Laud, H. Lipmaa. Eliminating counterevidence with applications to accountable certificate management. *Journal of Computer Security*, 10(3), pages 273–296, 2002.
7. A. Buldas and M. Saarepera. On provably secure time-stamping schemes. In Proc. of *ASIACRYPT 2004*, *LNCS 3329*, pages 500–514, 2004.
8. A. Buldas, P. Laud, M. Saarepera, and J. Willemson. Universally composable time-stamping schemes with audit. In *ISC05*, *LNCS 3650*, pages 359–373, 2005.
9. A. Buldas, S. Laur. Do broken hash functions affect the security of time-stamping schemes? In Proc. of *ACNS'06*, *LNCS 3989*, pages 50–65, 2006.
10. I. Damgård. Commitment schemes and zero knowledge protocols. In *Lectures on Data Security: modern cryptology in theory and prectice*, *LNCS 1561*, pages 63–86, 1999.
11. O. Goldreich. *Foundations of Cryptography II: Basic Applications*, Cambridge University Press, 2004.
12. S. Haber and W.-S. Stornetta. Secure Names for Bit-Strings. In Proc. of *ACM Conference on Computer and Communications Security*, pages 28–35, 1997.
13. T. Hagerup and C. Rüb. A Guided Tour of Chernoff Bounds. *Information Processing Letters*, 33, pages 305–308, 1990.
14. S. Halevi and S. Micali. Practical and provably-secure commitment schemes from collision-free hashing. In *CRYPTO'96*, *LNCS 1109*, pages 201–215, 1996.
15. R. C. Merkle. Protocols for public-key cryptosystems. *Proceedings of the 1980 IEEE Symposium on Security and Privacy*, pages 122–134, 1980.
16. G. Nuckolls, C. U. Martel, and S. G. Stubblebine. Certifying Data from Multiple Sources. In Proc. of the *DBSec 2003*, pages 47–60, 2003.
17. K. Nyberg. Fast accumulated hashing. In Proc. of *FSE'96*, *LNCS 1039*, pages 83–87, 1996.

A Combinatorial Extraction Lemma

Consider a finite matrix $\mathsf{W}[r;c]$ the rows of which are indexed by $r \in \mathcal{R}$ and the columns are indexed by $c \in \mathcal{C}$. Moreover, assume that a certain probability measure $\Pr[\cdot]$ is defined over the row indices $\mathcal{R}$. Then it is straightforward to state and prove a combinatorial lemma that we used for proving the knowledge-binding property.

Lemma 1. *For any $\delta > 0$ and $N \in \mathbb{N}$, there exist a set of column indices $\emptyset \subseteq \mathcal{I} \subseteq \mathcal{C}$ such that $0 \leq |\mathcal{I}| \leq \frac{N}{\delta}$ and for every column $c \in \mathcal{C}$:*

$$\Pr[r \leftarrow \mathcal{R} : \mathsf{W}[r;c] \neq 0 \wedge \mathsf{W}[r;c] \notin \mathcal{L}[r] \wedge |\mathcal{L}[r]| < N] \leq \delta \enspace ,$$

where $\mathcal{L}[r] = \{\mathsf{W}[r,c] : c \in \mathcal{I}\} \setminus \{0\}$ is the set of nonzero elements revealed by $\mathcal{I}$.

Proof. Consider following iterative procedure:

1. Set $\mathcal{I} = \emptyset$ and initialise row counters $\mathsf{cnt}[r] = N$ for $r \in \mathcal{R}$.
2. While exists $c \in \mathcal{C}$ such that $\Pr[r : \mathsf{W}[r;c] \neq 0] \geq \delta$ do
   (a) Choose c such that $\Pr[r : \mathsf{W}[r;c] \neq 0] \geq \delta$ and insert c into $\mathcal{I}$.
   (b) For each row $r \in \mathcal{R}$ such that $\mathsf{W}[r;c] \neq 0$ do
      – Store $w \leftarrow \mathsf{W}[r;c]$.
      – Remove w entries from the row.
        If $\mathsf{W}[r;c'] = w$ then $\mathsf{W}[r,c'] \leftarrow 0$ for $c' \in \mathcal{C}$.
      – Decrease counter $\mathsf{cnt}[r] \leftarrow \mathsf{cnt}[r] - 1$.
   (c) Zero all rows where $\mathsf{cnt}[r] = 0$.
      – If $\mathsf{cnt}[r] = 0$, set $\mathsf{W}[r;c'] \leftarrow 0$ for $c' \in \mathcal{C}$.

Let $\mathcal{N} = \{r : \exists \mathsf{W}[r;c] \neq 0\}$ denote nonzero rows and $\mathcal{N}_{\mathrm{old}}, \mathcal{N}_{\mathrm{new}}$ denote the value of $\mathcal{N}$ before and after update at Step 2. Let

$$\mu[\mathcal{N}] = \sum_{r \in \mathcal{N}} \Pr[r]\, \mathsf{cnt}[r]$$

be the average counter value. Then by the construction $\mu[\mathcal{N}_{new}] \leq \mu[\mathcal{N}_{old}] - \delta$ after a single iteration of Step 2. As initially $\mu[\mathcal{N}] \leq N$, then after $\lfloor N/\delta \rfloor$ iterations $\Pr[\mathcal{N}] \leq \mu[\mathcal{N}] < \delta$. Note that the algorithm nullifies the elements $\mathsf{W}[r,c']$ only if they already belong to $\mathcal{L}[r]$ or $|\mathcal{L}[r]| \geq N$. In the end, each column c contains at most a δ-fraction of elements that satisfy the predicate $\mathsf{W}[r;c] \neq 0 \wedge \mathsf{W}[r;c] \notin \mathcal{L}[r] \wedge |\mathcal{L}[r]| < N$ and the claim follows. Note that $\mathcal{I}$ can be empty. $\square$

$$
\begin{array}{cc}
\mathcal{I} = \emptyset & \mathcal{L} \\
\boxed{\begin{array}{ccccc} 1 & 2 & 0 & 1 & 1 \\ 1 & 0 & 3 & 0 & 2 \\ 2 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 2 \end{array}} & \begin{array}{c} \emptyset \\ \emptyset \\ \emptyset \\ \emptyset \\ \emptyset \end{array}
\end{array}
\Rightarrow
\begin{array}{cc}
\mathcal{I} = \{1\} & \mathcal{L} \\
\boxed{\begin{array}{ccccc} \mathbf{0} & 2 & 0 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & 3 & 0 & 2 \\ \mathbf{0} & 0 & 1 & \mathbf{0} & 3 \\ \mathbf{0} & 0 & 0 & 1 & 0 \\ \mathbf{0} & 0 & 0 & 0 & 2 \end{array}} & \begin{array}{c} \{1\} \\ \{1\} \\ \{2\} \\ \emptyset \\ \emptyset \end{array}
\end{array}
\Rightarrow
\begin{array}{cc}
\mathcal{I} = \{1,3\} & \mathcal{L} \\
\boxed{\begin{array}{ccccc} \mathbf{0} & 2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & 0 & \mathbf{0} & 1 & 0 \\ \mathbf{0} & 0 & \mathbf{0} & 0 & 2 \end{array}} & \begin{array}{c} \{1\} \\ \{1,3\} \\ \{2,1\} \\ \emptyset \\ \emptyset \end{array}
\end{array}
\quad
\begin{array}{cc}
\mathcal{I} = \{1,3\} & \mathcal{L} \\
\boxed{\begin{array}{ccccc} \mathbf{1} & \underline{2} & \mathbf{0} & \mathbf{1} & \mathbf{1} \\ \mathbf{1} & 0 & \mathbf{3} & 0 & 2 \\ \mathbf{2} & 0 & \mathbf{1} & \mathbf{2} & 3 \\ \mathbf{0} & 0 & \mathbf{0} & \underline{1} & 0 \\ \mathbf{0} & 0 & \mathbf{0} & 0 & \underline{2} \end{array}} & \begin{array}{c} \{1\} \\ \{1,3\} \\ \{1,2\} \\ \emptyset \\ \emptyset \end{array}
\end{array}
$$

Fig. 3. Illustration of Lemma 1. The first three sub-figures show how the columns are selected for the uniform distribution over the rows and for parameter values $N = 2$, $\delta = 0.3$, boldface symbols denote the changed values. The last sub-figure shows the final result. Boldface symbols denote the revealed entries. Underlined symbols denote the entries that satisfy the predicate.