DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS
19

SECURE AND EFFICIENT TIME-STAMPING SYSTEMS

HELGER LIPMAA

TARTU 1999

DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS
19

DISSERTATIONES MATHEMATICAE UNIVERSITATIS TARTUENSIS
19

SECURE AND EFFICIENT TIME-STAMPING SYSTEMS

HELGER LIPMAA

TARTU 1999

Department of Mathematics, University of Tartu, Estonia

Dissertation is accepted for the commencement of the degree of Doctor of Philosophy (PhD) on April 30, 1999 by the Council of the Department of Mathematics, University of Tartu.

Commencement will take place on June 30, 1999.

Publication of this dissertation is granted by the governmental financial support to PhD students.

Tartu Ülikooli Kirjastuse trükikoda
Tiigi 78, 50410 Tartu
Tellimus nr. 381

CONTENTS

List of Figures

LIST OF ORIGINAL PUBLICATIONS

1. Ahto Buldas, Peeter Laud, Helger Lipmaa, and Jan Villemson. Time-stamping with binary linking schemes. In Hugo Krawczyk, editor, *Advances on Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 486–501, Santa Barbara, USA, August 1998. Springer-Verlag.

2. Ahto Buldas, Helger Lipmaa, and Berry Schoenmakers. Optimally efficient accountable time-stamping. Submitted, May 1999.

3. Helger Lipmaa. IDEA: A cipher for multimedia architectures? In Stafford Tavares and Henk Meijer, editors, *Selected Areas in Cryptography '98*, volume 1556 of *Lecture Notes in Computer Science*, pages 248–263, Kingston, Canada, 17–18 August 1998. Springer-Verlag.

INTRODUCTION

Because of the increasing importance of the international business and communication, it has become necessary to forward signed documents to a long distance. Traditional mail system is too slow for the information society needs. Radio, TV, Fax and Computer Networks made it possible to send information from one point to another with the greatest possible speed — the speed of light. Of course, the received electronic document cannot just be assumed to be authentic because:

- electronic documents can be easily copied and modified without detection;
- unlike hand-written characters, digitally encoded characters have not the individuality;
- the signature of an electronic document is not physically connected to the content of the document.

Due to these disadvantages electronic documents are rarely considered to have any legal force.

Digital signature [DH76, RSA78, Pfi96] is a cryptographic technique that enables to protect digital information (represented as a bit-stream) from undesirable modification. Digital signatures are widely used to protect data in secure e-mail systems. Digital signature can effectively substitute the hand-written signature in the electronic environment. In many countries the laws and regulations have been adopted which equalize the use and functions of digital signature to handwritten signature. However, non of these countries have any experience of using digitally signed data as an evidence in the court.

The legal use of electronic records is increasingly important. One of the reasons, as it has been said above, is that electronic documents enable to communicate much faster. Another reason is that the archives are full of old paper documents which have a legal importance but are used very rarely. Saving the space in archives is urgently needed. This helps to understand why the initiative to form the Committee of Law of Electron Documents in Estonia became from the archivists.

It became clear that the most important step towards the legal use of electronic documents is to enable the legal regulation of digital signatures. Before issuing a law on digital signature, definite understanding of the technical details necessary to support the law is inevitable. Thereby, the intensive cooperation of lawyers, archivists and data security specialists is needed.

Security techniques used in electronic documents have been developed keeping an eye on secure messaging systems. However, the secure maintenance of electronic documents with a long lifetime is a bit more complicated task. Numerous problems in this area were not solved yet. Some of these have been regarded below from the viewpoint of data security specialists.

Private Keys

A problem related to the use of digital signatures is the management of the private signature keys. If somebody else, except the owner, gains access to the private key, he or she will be able to forge the owner's signatures on electronic documents. At that point even the value of legitimately signed documents can be called into question. Moreover, if the signer of a particularly important document (for example, a loan agreement) later wishes to repudiate his or her signature, he or she can dishonestly report the compromise of his or her private key.

Therefore, the verifier of a digitally signed document should be able to ascertain when the document was actually signed. Digital time-stamping is a solution to this problem.

Time-Stamping

Most of the time-stamping systems use a trusted third party called Time-Stamping Authority (TSA). The time-stamp is a digital attestation of the TSA that an identified electronic document, subscribed with a digital signature, has been presented to TSA at a certain time. *Time-stamping is a set of techniques enabling one to ascertain whether an electronic document was created or signed at a certain time.*

The real importance of time-stamping becomes clear when there is a need for a legal use of electronic documents with a long lifetime. Without time-stamping we neither can trust signed documents when the cryptographic primitives used for signing have become unreliable nor solve the cases when the signer himself repudiates the signing, claiming that he has accidentally lost his signature key. During the last years, especially in the context of legal regulation of using digital signatures, the organizational and legal aspects of time-stamping itself have become the subject of world-wide attention. In addition to defining the responsibilities of the owner of the signature, duties and responsibilities of the third party (Time-Stamping Authority, TSA) must be stated as well. Hence, there is an increasing interest in time-stamping systems where the need to trust the TSA is minimized. In order to make users liable only for their own mistakes, there has to be a possibility to ascertain the offender.

Unlike physical objects, digital documents do not comprise the seal of time. Thus, the association of an electronic document uniquely with a certain moment of time is very complicated, if not impossible. Even by the theory of relativity, no absolute time exists. The best we can achieve with time-stamping is the relative temporal authentication (RTA) based on the complexity-theoretic assumption on the existence of collision-resistant hash functions. RTA enables the verifier given two time-stamped documents to verify which of the two was created earlier.

Some ten years ago time-stamping was considered to be an uninteresting area since the only known time-stamping method employed completely trusted third party — the Time-Stamping Authority. Whatever the TSA said the clients had to believe. More people became interested in this field after the seminal publication [HS90] of Haber and Stornetta, where it was shown that the trust to the TSA can be creatly reduced by using so called *linking schemes*. Several papers improving the original schemes were published in early nineties. The surge of papers dried soon, since it seemt that everything attainable has been attained. Again, the area was considered to be uninterested. Until [BLLV98] was published.

Main Contributions

Out contributions to the field are manifold. At first, the paper [BLLV98] by Buldas, Laud, Lipmaa and Villemson was the first scientific paper ever that explicitly treated the (extremely strong) security requirements of legally applicable time-stamping. Starting from the cognition that absolute temporal authentication is impossible, this paper presented a new set of protocols that enables one to prove the relative temporal authentication, and to detect and demonstrate the frauds by a trusted third party (i.e., the proposed time-stamping system was *accountable*). A number of previously proposed time-stamping systems was examined an discarded. It was also shown that the linear linking scheme of Haber and Stornetta [HS90, HS91] can be used to achieve accountability, but the resulting system would be plainly impractical. A new system (based on "binary linking schemes") was proposed that was accountable and practical.

The paper [BL98] by Buldas and Laud formalized the "anti-monotonicity" requirement that these binary linking schemes had to satisfy and defined a new anti-monotone binary linking scheme (shortly, AM scheme), where the certificate sizes were tightly optimal in the class of AM schemes.

Further, the paper [BLS99] of Buldas, Lipmaa and Schoenmakers showed that the anti-monotonicity requirement is unnecessary for accountability. A new family of graphs ("threaded authentication trees") was proposed, and it was proven that this family is tightly optimal (in certificate size) in the class of all acyclic digraphs. In fact, the term "accountable time-stamping" was the first time proposed by this paper, This paper also specified the security requirements, by explicitly listing the security preconditions, under which the accountability can be achieved. For first time ever, also the publication protocol between the time-stamping authority and the publishing authority was discussed. It was shown that the publishing process does not have to be blindly trusted. Indeed, it is possible to establish which party is guilty in wrong publications (the publication protocol is not discussed in this thesis, however).

Apart from the intra-round optimizations, the paper [BLLV98] also defined *accumulated time-stamping*, where the cumulative *round stamps* are connected with each other in a similar manner as the stamps of one round. Such two-layered approach enables efficient verification of the one-way dependency between stamps issued in different rounds even when the TSA does not have a copy of stamps from intermediate rounds. Hence, the storage requirements to the TSA decrease drasticly. A manuscript [Lip99] of the author of this thesis formalizes and simplifies accumulated time-stamping.

Briefly, our objective was to elaborate a secure and efficient time-stamping system. This objective was obtained. Moreover, new system is tightly optimal under some reasonable assumptions. We would like to stress that the optimal efficiency is a hard thing to attain. While our systems are optimal in space complexity, they may not be optimal in time complexity. Optimality in time would also depend on usage of fast cryptographic primitives, and fast implementations of these. Nobody knows what is the lower bound of the time complexity of collision-resistant hash functions. Moreover, as it was exemplified in [Lip98], rapidity of cryptographic primitives intrinsicly depends on the hardware used and on the programming skills of the implementer.

Outline of Thesis

In Sect. 1 we give a brief short survey of the cryptographic prerequisites. In Sect. 2 the time-stamping solutions proposed to date are analyzed. Sect. 3 clarifies the security objectives of time-stamping by giving essential requirements to the time-stamping systems. In Sect. 4 the protocols of the new time-stamping system are described using the linear linking scheme. In Sect. 5 anti-monotone schemes are introduced and a scheme with logarithmic verifying time is presented. Finally, in Sect. 6, optimal schemes are investigated. A new scheme that is tightly optimal in the general case is proposed.

1 PREREQUISITES

1.1 Collision-Resistant Hash Functions

Definition 1 (Collision-resistant hash function) A *collision-resistant hash function* ([MOV96, Sect. 9.2]) is a function H which has the properties of

- *compression* — H maps an input x of arbitrary finite bit-length, to an output $H(x)$ of fixed bit-length χ;
- *ease of computation* — given H and an input x, $H(x)$ is easy to compute;
- *preimage resistance* — for essentially all pre-specified outputs, it is computationally infeasible to find any input which hashes to that output, i.e., to find any preimage x' such that $H(x') = y$ when given any y for which a corresponding input is not known;
- *second-preimage resistance* — it is computationally infeasible to find any second input which has the same output as any specified input, i.e, given x, to find a 2nd-preimage $x' \neq x$ such that $H(x) = H(x')$, and
- *collision resistance* — it is computationally infeasible to find any two distinct inputs x, x' which hash to the same output, i.e., such that $H(x) = H(x')$. (Note that here there is free choice of both inputs.)

While the existence of collision-resistant hash functions (CRHF) is still an important open problem, there exists a plethora of fast candidate hash functions, including SHA-1 [NIS94] and RIPEMD-160 [DBP96]. For a recent overview of the literature about the connections between collision-resistant hash functions and other cryptographic primitives, we direct readers to [Sim98].

1.2 Digital Signatures

Another cryptographic primitive we use are the *digital signatures* [DH76, RSA78]. In what follows, let $\mathrm{sig}_A(M)$ be A's signature on the message M. Since the time-stamping protocols require the principals to sign arbitrary data, the used signature scheme should be secure against adaptive chosen message attack [GMR88]. The most efficient and secure signature schemes [CS98, GHR99] known at the current moment base on the recently proposed *Strong RSA Assumption* [BP97].

1.3 Authentication Graphs

Let $G = (V, E)$ be a directed acyclic graph (DAG), which we assume to be topologically sorted (i.e., for each edge $(v, w) \in E$ we have $v < w$), where $V = \{1, \dots, n(G) = |V|\}$. Furthermore, we assume that vertex $|V|$ is the unique sink of G, which we sometimes call the *root* of G. The set of sources of G is denoted by $\mathcal{S}(G)$. We say that G is *simply connected* if there is a directed path between any two vertices v and w with $v < w$ (or equivalently, if $(v, v+1) \in E$, for all v except the root).

As a general notation, let $\vec{L}_N := (L_{n_1}, \dots, L_{n_k})$, where $n_1, \dots, n_k$ are the elements of N in strictly increasing order. Recall that h denotes a CRHF. As a generalization of Merkle's authentication trees [Mer80], we introduce *authentication graphs G* as labeled DAGs with the labels assigned in the following manner. Each source v of G is labeled by a label $L_v = H_v$, where H_v is a given string specific to source v. The label of a non-source vertex v is computed as a function of the labels of its proper ancestors: $L_v = h(\vec{L}_{E^{-1}(v)})$.

We say that $v \in V$ is *computable* from $W \subseteq V$ if either

1. $v \in W$ or
2. $E^{-1}(v) \neq \emptyset$ and all $w \in E^{-1}(v)$ are computable from W.

We say that $W_1 \subseteq V$ is *computable* from $W_2 \subseteq V$ if all $v \in W_1$ are computable from W_2. For any path $\mathcal{P} \subseteq V(G) \setminus \mathcal{S}(G)$, we say that the set $\mathrm{AP}_G(\mathcal{P}) := E^{-1}(\mathcal{P}) \setminus \mathcal{P}$ is the *authenticator* of $\mathcal{P}$. Clearly, $\mathcal{P}$ is computable from $\mathrm{AP}_G(\mathcal{P})$. A vertex v computable from $\mathcal{P}$ is *consistent* with v if the subgraph of G induced by $\mathcal{P} \cup \{v\}$ is a subgraph of some authentication graph (i.e., if there are no internal inconsistencies in calculation the L_v's of the joint graph).

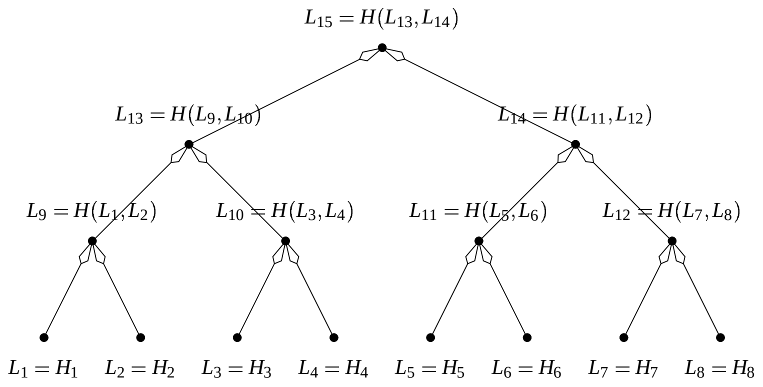


Figure 1: Merkle's authentication tree V_3.

Let $d(v,w)$ denote the distance from vertex v to vertex w, that is, length of the shortest path from v to w, and let $d_{pt}(G) = \max_{v\in V}(d(1,v) + d(v,|V|))$. We say that G is *dense* if $d_{pt}(G) = (\log|V|)^{O(1)}$.

1.4 Merkle's Authentication Tree

As an example, let us look at the *Merkle's authentication tree* $V_d = (V,E)$ [Mer80] with 2^d sources (V_3 is represented by Figure 1). Trivially, V_d is an authentication graph. V_d is not simply connected, but it is dense ($d_{pt}(V_d) = d = \log(n(V_d) + 1)$). Let $\mathcal{P} = (3,10,13,15)$ be a path in V_3. Corresponding authenticator is $\mathrm{AP}_{V_3}(\mathcal{P}) = (9,4,14)$.

2 EXISTING TIME-STAMPING SYSTEMS

2.1 Simple Solutions

Let us at first describe a "naïve" solution, "digital safety-deposit box" [HS91, Sect. 3]. Whenever a client has a document to be time-stamped, he or she transmits the document to a time-stamping authority (TSA). The authority records the date and time the document was received and retains a copy of the document for safe-keeping. If the integrity of the client's document is ever challenged, it can be compared to the copy stored by the TSA. If they are identical, this is evidence that the document has not been tampered with after the date contained in the TSA records. This procedure does in fact meet the central requirements for the time-stamping of a digital document. However, this approach raises several concerns:

Privacy This method compromises the privacy of the document in two ways: a third party could eavesdrop while the document is being transmitted, and after transmission it is available indefinitely to the TSA itself. Thus the client has to worry not only about the security of documents it keeps under its direct control, but also about the security of its documents at the TSA.

Bandwidth and storage Both the amounts of time required to send a document for time-stamping and the amount of storage required at the TSA depend on the length of the document to be time-stamped. Thus the time and expense required to time-stamp a large document might be prohibitive.

Incompetence The TSA copy of the document could be corrupted in transmission to the TSA, it could be incorrectly time-stamped when it arrives at the TSA, or it could become corrupted or lost altogether at any time while it is stored at the TSA. Any of these occurrences would invalidate the client's time-stamping claim.

Trust The fundamental problem remains: nothing in this scheme prevents the TSA from colluding with a client in order to claim to have time-stamped a document for a date and time different from the actual one.

The next simple time-stamping protocol, discussed by Haber and Stornetta [HS91, Sect. 4], addresses the first three concerns listed above. The final issue, trust, will be handled in the subsequent sections. By this protocol, the TSA appends the current time t to the submitted document X, signs the composite document (t, X) and returns the two values t and $s = \mathrm{sig}_{\mathrm{TSA}}(t, X)$ to the client. The weaknesses of this scheme are the unreliability of old time-stamps after a possible leakage of the signature key of the TSA and the impossibility of verifying whether s was issued actually at time t stated in the time-stamp, implying that *the TSA has to be unconditionally trusted.* Because of these drawbacks it has been widely accepted that a secure time-stamping system cannot rely solely on keys or on any other secret information. Two recent overviews of the existing time-stamping solutions are [MQ97] and [Jus98a].

2.2 Linear Linking Scheme (LLS)

In order to diminish the need for trust, the users may demand that the TSA links all time-stamps together into a chain using a collision-resistant hash function H as was proposed in [HS91, Sect. 5.1, variant 1] (Figure 2). In this case the time-stamp for the n-th submitted document H_n is

$$s = \mathrm{sig}_{TSA}(n, t_n, \mathrm{ID}_n, H_n, L_n) \ ,$$

where t_n is the current time, ID_n is the identifier of the submitter and L_n is the linking information defined by the recursive equation

$$L_n := (t_{n-1}, \mathrm{ID}_{n-1}, H_{n-1}, H(L_{n-1})) \ .$$

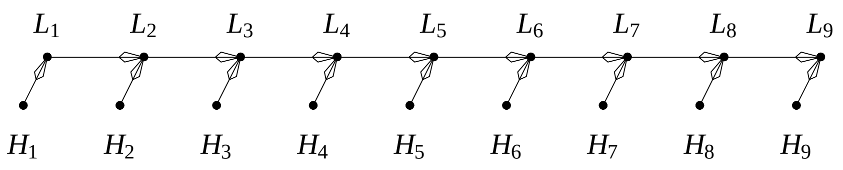


Figure 2: *Linear linking scheme* with 9 elements.

There are several complications with the practical implementation of this scheme. Most important complication is that the number of steps needed to verify the one-way relationship between two stamps is linear with respect to the number of stamps between them. Hence, a single verification may be as costly as it was to create the whole chain. This solution has impractical trust and broadcast requirements, as it was pointed out already in [BdM91]. A modification was proposed in [HS91, Sect. 5.1, variant 2] (Figure 3) where every time-stamp is linked with $k > 1$ time-stamps directly preceding it. This variation decreases the requirements for broadcast by increasing the space needed to store individual stamps.

2.3 Tree-Like Schemes

Two similar schemes based on Merkle's authnetication trees [Mer80] have been proposed [BdM91, BHS92]. In the Haber-Stornetta scheme [BHS92, HS97], the

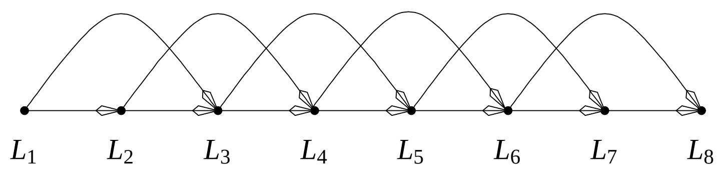


Figure 3: A modification of linear linking scheme, where every time-stamp is linked with 2 time-stamps directly preceding it.

time-stamping procedure is divided into *rounds*. The time-stamp R_r for round r is a cumulative hash of the time-stamp R_{r-1} for round $r-1$ and of all the documents submitted to the TSA during the round r. After the end of the r-th round a binary tree T_r is built.

Every participant P_i who wants to time-stamp at least one document in this round, submits to the TSA a hash $y_{r,i}$ which is a hash of R_{r-1} and of all the documents he wants to time-stamp in this round. The sources of T_r are labeled by different $y_{r,i}$. Each inner node k of T_r is recursively labeled by $H_k := H(H_{k_\mathsf{L}}, H_{k_\mathsf{R}})$ [HS97], where k_L and k_R are correspondingly the left and the right proper ancestor of k, and H is a collision-resistant hash function. The TSA has to store only the time-stamps R_r for rounds (Figure 4). All the remaining information, required to verify whether a certain document was time-stamped during a fixed round, is included into the individual time-stamp of the document.

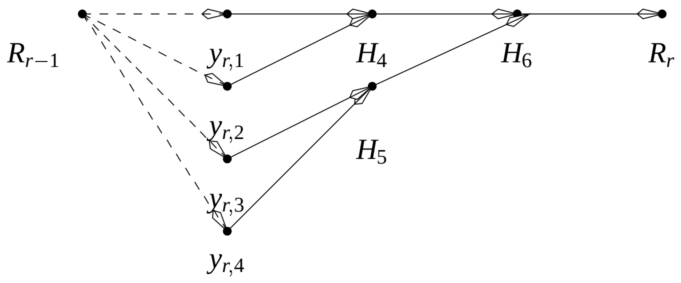


Figure 4: An example of the time-stamp for round r by the schemes presented in [BdM91].

For example, the individual time-stamp for $y_{r,3}$ is $[r;(y_{r,4},\mathsf{L}),(H_4,\mathsf{R})]$. The verifying procedure of the time-stamp of $y_{r,3}$ consists of verifying the equality $R_r = H(H(H_4,H(y_{r,3},y_{r,4})),R_{r-1})$. Here, the size of a single time-stamp is logarithmic with respect to the number of participants submitting their documents to the TSA for the current round.

The Haber–Stornetta tree scheme [BHS92, HS97] (Figure 5) differs slightly from the Benaloh–de Mare scheme [BdM91]. Here, the time-stamp R_n for the n-th round is linked directly to R_{n-1}, enabling the verifier to check one-way dependencies between R_i without examining the individual time-stamps of the submitted documents. This is impossible in the Benaloh–de Mare scheme. However, in the Haber–Stornetta scheme the individual time-stamps in the n-th round are not linked to the time-stamp R_{n-1} for previous round.

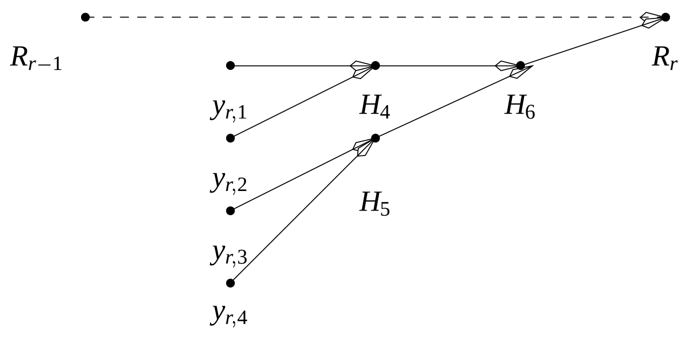


Figure 5: An example of the time-stamp for round r by the schemes presented in [BHS92].

These schemes are feasible but provide the RTA for the documents issued during the same round only if we unconditionally trust the TSA to maintain the order of time-stamps in T_r. Therefore, this method either increases the need for trust or otherwise limits the maximum temporal duration of rounds to the insignificant units of time. However, if the number of submitted documents during a round is too small, the expenses of time-stamping a single document may become unreasonably large (Sect. 3.5.4).

3 SECURITY OBJECTIVES

In the following we give a definition of time-stamping systems applicable in legal situations. Later we will justify our approach and compare it to older ones.

A *time-stamping system* consists of a set of principals with the *time-stamping authority* (*TSA*) together with a triple $(\mathbf{S}, \mathbf{V}, \mathbf{A})$ of protocols. The *stamping protocol* **S** allows each participant to post a message. The *verification algorithm* **V** is used by a principal having two stamps to verify the temporal order between those time-stamps. The *audit protocol* **A** is used by a principal to verify whether the TSA carries out his duties. Additionally, no principal (in particular, TSA) should be able to produce fake time-stamps without being caught (this notion will be formalized in Sect. 3.3).

A time-stamping system has to be able to handle time-stamps which are anonymous and do not reveal any information about the content of the stamped data. The TSA is not required to identify the initiators of time-stamping requests. Our notion of time-stamping system differs from the one given in, e.g., [BdM91] by several important aspects. Below we motivate the differences.

3.1 Relative Temporal Authentication

The main security objective of time-stamping is *temporal authentication* (a phrase coined probably by Michael Just, [Jus98b]) — ability to prove that a certain document has been created at a certain moment of time. Although the creation of a digital data item is an observable event in the physical world, the moment of its creation cannot be ascertained by observing the data itself (moreover, even by the theory of relativity, no such thing as the absolute time exists). The best one can do is to check the relative temporal order between some auxiliary data (i.e., prove the *relative temporal authentication, RTA*) using one-way dependencies defining the arrow of time, analogous to the way in which the growth of entropy defines the arrow of time in the physical world [Haw88, Chap. 9].

All (but the most simplest) time-stamping systems that we describe use some authentication graph G built in a black-box manner upon a collision-resistant one-way hash function H. In such cases one says that n is *one-way dependent* of m if there is a G-path from m to n. The intuition behind this is the following "rough" derivation rule:

> If $H(X)$ and X are known to a principal A at a moment t, then someone (possibly A himself) used X to compute $H(X)$ at a moment prior to t.

All proposed time-stamping systems make sense only under the hypothesis of the existence of collision free one-way hash functions.

Vertices of G are also sometimes referred as *linking items*. In the context of time-stamping H_n (referred to also as *data items*) denotes the (hash of the) n-th time-stamped document. The *time certificate* $\mathrm{Cert}(n)$ *of* H_n is equal to $(n, \vec{L}_N)$, for some $N \subseteq \{1, \dots, n(G)\}$ (exact definition will depend on the graph G and will be given later). To achieve RTA it is necessary for some $L_{n'}$ in $\mathrm{Cert}(n)$ to be one-way dependent of some element $L_{m'}$ in $\mathrm{Cert}(m)$, whenever $m < n$.

Note that a one-way relationship between L_n and L_m does not prove that in the moment of creating H_n the bit-string H_m did *not* exist. All we know is that H_n did exist at the moment of creating L_m.

3.2 Discussion: Absolute Time

The stamp of H_n does not contain any absolute time t_n whereas it could not be taken for granted that the value t_n indeed represents the submission time of H_n. The only way for a principal to associate a stamp with a certain moment of time is to stamp a *nonce* at this moment. By a nonce we mean a sufficiently long (say, k-bit, $k \geq 160$) random bit-string, such that the probability it has been already time-stamped is negligible (easily achieved if the bit-string is generated using a cryptographically strong pseudo-random number generator).

In order to verify the absolute creating time of a document time-stamped by another principal, the verifier has to compare the time-stamp with the time-stamps of nonces generated by the verifier herself. In this solution there are neither supplementary duties to the TSA nor to the other principals. The use of nonces illustrates the similarity between time-stamping and ordinary authentication protocols, where nonces are used to prevent the possible reuse of old messages from previous communications (a reader interested more in authentication protocols, is directed to, e.g., [BR93]).

By using relative temporal authentication it is possible to determine not only the submitting time of the signature but also the time of signing the document. Before signing a document X the principal A generates a nonce R and stamps it. He then includes the stamp $L(R)$ of R to the document (Figure 6), signs it and obtains the stamp $L(X)$ of the signature $X' = \mathrm{sig}_A(L(R), X)$. From the view-point of the TSA these two stamping events are identical (he need not be aware whether he is stamping a nonce or meaningful data). For the verification of the document X, the verifier has to compare both these tamps with the stamps trusted by her. As there are one-way dependencies between $L(R)$, X' and $L(X')$, the verifier may conclude that the signature was created in the time-frame between the moments of issuance of $L(R)$ and of $L(X')$ respectively. If these moments are close enough, the signing time can be ascertained with necessary precision.

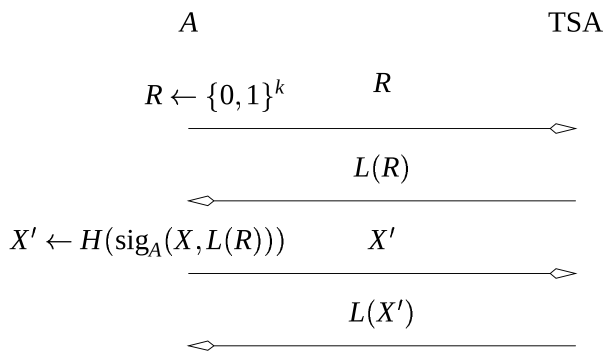


Figure 6: Double-stamping a signature.

3.3 Accountable Time-Stamping

A time-stamping system must have properties enabling users to verify whether an arbitrary time certificate is correct or not. (As we already noted, possession of two documents with corresponding time certificates is not enough to prove the RTA between the *documents* because everyone is able to produce fake chains of stamps.)

A time-stamping system should allow

1. To determine whether the time certificates possessed by an individual have been tampered with; and

2. In the case of tampering, to determine whether the certificates were tampered by the TSA or tampered after the issuing (generally by unknown means).

In the second case, there is no-one to bring an action against.

As a general principle, the principals interested in legal use of stamps should themselves verify their correctness immediately after the issuing (using signatures and other techniques discussed later) because if the signature of the TSA becomes unreliable, the signed time-stamps cannot be used as an evidence.

In order to increase the trustworthiness of the time-stamping services it should be possible for the clients to periodically inspect the TSA. Also, in the case when the TSA is not guilty he should have a mechanism to prove his innocence, i.e., that he has not issued a certain time-stamp during a certain round.

Additionally, the TSA must publish regularly, in an authenticated manner, the time-stamps for rounds [BdM91] in mass media. If the time-stamping protocol includes (by using collision-resistant one-way hash functions)

1. the message digest of any time-stamp issued during the r-th round into the time-stamp for r-th round, and

2. the message digest of the time-stamp for round $r-1$ into any time-stamp issued during the r-th round,

it will be intractable for anyone to undetectably forge a time-stamp. The forgery detection procedures should be simple. Forgeries should be determinable either during the stamping protocol (when the time-stamp, signed by the TSA, fails to be correct) or later when it is unable to establish the temporal order between two otherwise correct time-stamps (see Sect. 4 for details).

Definition 2 A time-stamping system is *accountable*, if (on some reasonable security assumptions) it makes the trusted third parties accountable for their actions by enabling a principal to detect and later prove to the judge any frauds. Moreover, if a party has honestly followed the protocol but will still be accused in forgeries, she can explicitly disprove any false accusations.

3.4 Security Preconditions

In the current subsection we postulate some security assumptions. It can be shown that the later proposed time-stamping systems are accountable if the following — usually tacitly assumed in the crypto context — conditions hold:

Liveliness. The TSA responds to the clients instantaneously (i.e., no *Denial of Service* attacks). "Denial of Service" attacks are extremely powerful against the time-stamping systems (indeed, the whole concept of a TSI would become meaningless if DoS attacks against it were easy to employ) and should therefore be considered very seriously.

Competence of honest parties. The party willing to prove misbehaviour of other parties is honest and competent. In particular, she should verify the signatures and not accept if the verification fails.

Key secrecy. The signing keys of TSA do not compromise until the end of the next round. Otherwise, the owner of the leaked key holds full responsibility for any damage resulting from the key compromise. It is reasonable to assume that the TSA is competent to keep his keys uncompromised during a short time-frame. Timely reporting of the key compromise decreases the influence of the key leakage to one round.

Validity of cryptographic assumptions. None of the used cryptographic primitives (hash function, signature scheme) is broken during a round. Note that key secrecy implies in general (but not always) that the used signature scheme is secure.

3.5 Feasibility Requirements

3.5.1 Connectivity

Sect. 4 shows how to modify the linear linking scheme [HS91, Sect. 5.1] to achieve accountability. On the other hand, one certificate verification takes as much time as was done by the TSA stamping all the documents. As noted, e.g., in [Jus98b], it is easy to forge stamps when we can assume that the verifier has limited computational power. Hence, the number of stamps per round is limited by the computational power of the weakest potential verifier. This leads us to the question of feasibility.

3.5.2 Graph Density

The time wasted in verification is of order $d_{pt}(G)$. If $d_{pt}(G) \approx n(G)$, the number of stamps issued is very much restricted by the desire to have lightning fast verification procedure. But a time-stamping service is assumed to work for ages, and to issue millions billions of time-stamps. Clearly, in this case verification is feasible only if the authentication graph G is dense, i.e., if $d_{pt}(G) = (\log n(G))^{O(1)}$. The less the value $d_{pt}(G)$, the bigger is the gap between the work done by issuing the stamps and by a single verification. One of the most important results of our work is that we found a tight lower bound for the value of $d_{pt}(G)$.

3.5.3 Connectivity

Tree-like schemes [BdM91, BHS92] were motivated by the desire to compress the data submitted to the TSA during short time intervals referred to as *rounds*. The duration of rounds was assumed to be small enough to think about the time-stamping requests submitted during the same round as simultaneous. Unfortunately, bounding the round lengths reduces the compression effect A simple enlarging of rounds is insecure because of the *reordering attack* where the TSA rearranges the time-stamping requests submitted during the round.
Reordering attack. Let Alice request a stamp for her document X. Bob has bribed the TSA to delay stamping of *any* Alice's documents. Hence, TSA assigns a conveniently large number m to Alice's stamp, marks m as "issued" and continues by assigning the intermediate numbers to the next documents. Alice can neither detect reordering attacks nor prove their existence to third parties.
In case of a simply connected graph, a demonstrated one-way dependent path between two stamps can clearly be interpreted as a proof. On the other hand, if a graph is not simply connected, a possible future verifier of a temporal order is forced to rely on linking conventions such as "previously issued stamps locate left

in the tree, compared to the later issued stamps". Although the use of such conventions will probably almost always suffice in practice, it still introduces a certain degree of looseness into the system. We can certainly imagine a situation, when it is advantageous to all of the principals to follow the protocol for a certain time, but later to change the practice and pretend that at this time in the past, left was right and right was left. Even if such attacks may seem to be exaggerated, we should still worry about them.

Our quest is to elaborate a practical "real world" time-stamping service relying only on some simple, easily understood security preconditions, so that *no matter how* the clients use the time-stamps in higher level protocols, the security objectives of these protocols will not be refuted if an "ideal world" trusted TSS is replaced with the "real world" untrusted TSS. Such approach of simulating an ideal world object with a real world object is common in modern cryptography. The current thesis is more concerned with efficiency than with security, but we stress that reordering attack is one of the possible forgeries the TSA can accomplish and thus, simply connected authentication graph is a necessary precondition for *accountability.*

3.5.4 Accumulated Time-Stamping

In order to make relative temporal authentication feasible in the case when stamps belong to different rounds, it is reasonable to define an additional layer of links between the time-stamps for rounds. We do it by defining *graph composition* $G_1 \circ G_2$. Informally, G_2 is the authentication graph used inside the rounds. G_1 is used as this second layer. Both G_1 and G_2 should be sufficiently large finite graphs (say, of $2^{32} \ldots 2^{80}$ vertices).

Definition 3 Let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two authentication graphs. *Graph composition* $G_1 \circ G_2 = (V, E)$ (Figure 7) is a graph produced from G_1 by replacing every source r of G_1 with a copy G_2^r of G_2 (more precisely, by identifying the sink of G_2^r with r), and by identifying the sink of G_2^r with the first vertex (which is, by definition, a source) of G_2^{r+1}. Trivially, $G_1 \circ G_2$ is an acyclic digraph with sink, and thus, correctly labeled, also an authentication graph. With G_1 and G_2 fixed, we define $\xi(r)$ to be the number of the rth source of G_1 in $G_1 \circ G_2$ (we assume that if $r_1 < r_2$ then $\xi(r_1) < \xi(r_2)$). Let $\mathcal{S}_\circ(G_1 \circ G_2)$ be the set of vertices $m \in V(G_1 \circ G_2)$ such that $(m, \xi(r)) \in E^*$ for some r (i.e., the set of vertices $n \in G_2^r$, for some r). The rth copy G_2^r of G_2 is also called *rth round* (of $G_1 \circ G_2$). The values $\mathcal{L}_r = L_{\xi(r)}$ are also referred to as the *stamps for rounds.*

If (the subgraphs induced by) $G_1 \setminus \mathcal{S}(G_1)$ and $G_2 \setminus \mathcal{S}(G_2)$ are simply connected, then so is (the subgraph induced by) $(G_1 \circ G_2) \setminus \mathcal{S}(G_1 \circ G_2)$. Thus $G_1 \circ G_2$ could be used in a system achieving RTA. Note that the stamps requested from the TSA

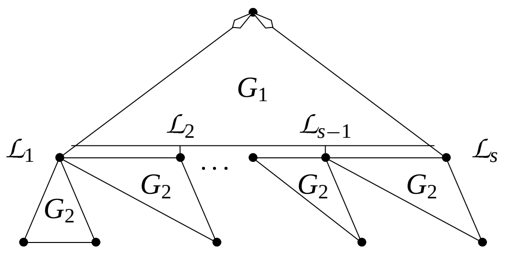


Figure 7: Graph composition $G_1 \circ G_2$. Here, $s = |\mathcal{S}(G_1)|$.

during the verification algorithm should belong to the set of stamps for rounds because only these stamps are available in the TSA (see Sect. 4.2 for more).

An obvious generalization of the construct $G_1 \circ G_2$ is to allow the copies of G_2 differ. Namely, let the *generalized graph composition* $G \circ [G_1 \dots G_s]$ be the graph constructed from G by replacing the rth source with a copy of G_r. Everything else is defined as in Def. 3.

Definition 4 Let G be an authentication graph and let $\xi : V(G) \to V(G)$. (G,ξ) is an *accumulated graph* with *rank m*, if

1. If $\xi(r) < n \leq \xi(r+1)$ then $E^{-1}(n) \subset [\xi(r), \xi(r+1)] \cup \xi(\mathbb{N})$;
2. $\xi(r+1) - \xi(r) \geq m$.

We say that G is *an accumulated graph* if for arbitrary positive m there exists $\xi : V(G) \to V(G)$, such that the (G,ξ)-scheme is an accumulated graph with rank m.

If the underlying authentication graph is accumulated, the duration of the rounds can be flexibly enlarged in order to guarantee that only a small fraction of the time-stamps are kept in the memory of the time-stamping server. A way to achieve accumulated time-stamping was proposed in [BLLV98]. A much simpler way to achieve the flexibility in choosing the round lengths is to use the generalized graph composition $G \circ [G_1 \dots G_s]$, where rth round G_r is itself a member of recursively constructed graph family, so that G_r can be extended or "cut" if necessary. Some examples of accumulated schemes will be given later.

3.6 Necessity of Rounds

There are several, partially controversial, reasons why *rounds* are introduced into time-stamping. Most of the reasons are explained elsewhere in this thesis. Here we just give a concentrated list of *some* of the reasons why it is necessary to have 1) rounds at all and 2) flexibility in choosing the round lengths.

Storage A time-stamping authority will not and should not be able to store all the time-stamps issued during its functioning (cf Sect. 3.3).

Havoc limitation In the systems described in the next sections, time-stamping authority's secret key is required to stay secure only in duration of one-two rounds. Key compromise will influence only the trustworthiness of stamps issued during a single round (cf Sect. 3.4). The same is true in the case if the underlying cryptographic primitives are suddenly broken.

Accountability Accountability is impossible if the verifier (say, the judge) does not possess an authenticated common successor of the two disputed stamps (this is due to the simplicity of building "parallel worlds" of one-way dependencies).

Connection with absolute time As it was stressed in Sect. 3.1, relative temporal authentication is what we can expect from the time-stamping. There still should be a connection with real time, at some point. For example, in court in some cases one would like to refer to a particular date. Then it helps if the absolute starting time of each round is publicly verifiable.

4 ACCUMULATED LINEAR LINKING SCHEME

For pedagogical reasons, we outline the protocols and the basic organizational principles of our system using the linear linking scheme of Sect. 2.2, upon which an accumulated linking scheme is built, corresponding to the ideology of Sect. 3.5.4. We will describe the operation of TSA, and also the protocols. Described scheme fulfills all the trust requirements but is impractical. In the next sections, efficiency of the described scheme is significantly improved by replacing the linear scheme with a binary linking scheme. The final scheme, presented in Sect. 6.2 has tightly (i.e., not only asymptotically) optimal stamp lengths.

4.1 Description of the Authentication Graph

Let the number M of time-stamps per round be a constant known to the participants (clients). The round graph G_2 is equal to the linear linking scheme Λ_M with M sources. The second layer graph G_1 is equal to the linear linking scheme with, say $N = 2^{80}$, sources (Figure 8). Thus, this scheme bases on graph $\Lambda_N \circ \Lambda_M$. The time-stamp for the r-th round has number $\xi(r) = M \cdot r$. Let all the data items H_n be of fixed size of k bits (i.e., the documents are hashed).

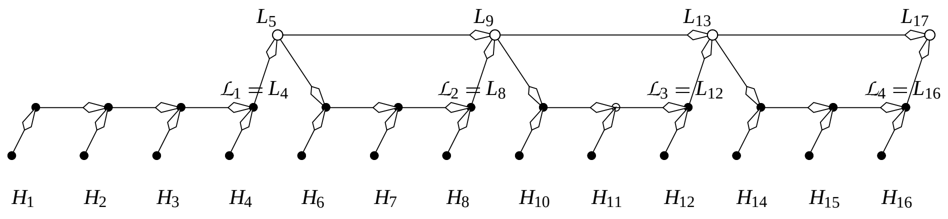


Figure 8: A two-layered linear linking "toy" scheme $\Lambda_4 \circ \Lambda_4$.

Remark 1 *In real life, the underlying scheme would be*

$$\Lambda_N \circ [\Lambda_{M_1} \ldots \Lambda_{M_s}] \ .$$

Due to the construction of linear schemes, the TSA can choose the round length at any time. Thus, the TSA has very flexible choice of round lengths. While this is very important in practice, we will not stress this point for the next schemes.

All our subsequent constructions will use the next definition (resp. to the corresponding underlying graph).

Definition 5 (Head and tail) Let $G = G_1 \circ G_2$ be the underlying graph of an accumulated time-stamping system. For any $m \in [\xi(r-1)+1, \xi(r)]$, let $\mathcal{H}_m := (\xi(r-1), m_1, \dots, m)$ (resp $\mathcal{T}_m := (m, m_2, \dots, \xi(r))$) be the unique shortest path from $\xi(r-1)$ to m (resp from m to $\xi(r)$). Let $\mathcal{H}'_m = (m_1, \dots, m)$ and $\mathcal{T}'_m := (m_2, \dots, \xi(r))$. Let $\text{head}(m) := \text{AP}_{G_2}(\mathcal{H}'_m)$, $\text{tail}(m) := \text{AP}_{G_2}(\mathcal{T}'_m)$. Let

$$\text{Cert}(m) := (m, \vec{L}_{\text{head}(m)}, \vec{L}_{\text{tail}(m)})$$

be the *time certificate* of H_m. For any m and n, $m < n$, let $\mathcal{B}_{mn}$ be the unique shortest path between m and n. Let $\text{body}(m,n) := \text{AP}_G(\mathcal{B}_{mn})$. The functions $\text{head}_\circ(\cdot)$, $\text{tail}_\circ(\cdot)$ and $\text{body}_\circ(\cdot,\cdot)$ are natural extensions of those functions to the whole graph $G_1 \circ G_2$.

If $G = \Lambda_N \circ \Lambda_M$ and m and n belong to the same round, then

$$\vec{L}_{\text{head}(n)} = (L_{\xi(r-1)}, H_{\xi(r-1)+1}, \cdots, H_{n-1}, H_n) \ ,$$
$$\vec{L}_{\text{tail}(m)} = (H_{m+1}, H_{m+1}, \dots, H_{\xi(r)-1}, L_{\xi(r)})$$

and

$$\vec{L}_{\text{body}(m,n)} = (L_m, H_{m+1}, \dots, H_{n-1}, H_n) \ .$$

Clearly, $\vec{L}_{\text{body}(m,n)}$ is computable from $\vec{L}_{\text{tail}(m)} \cup \vec{L}_{\text{head}(n)}$ and

$$\vec{L}_{\text{body}_\circ(m,n)} = \begin{cases} (\vec{L}_{\text{tail}(m)}, \mathcal{L}_{r(m)+1}, \dots, \mathcal{L}_{r(n)-1}, \vec{L}_{\text{head}(n)}), & r(m) < r(n), \\ \vec{L}_{\text{body}(m,n)}, & r(m) = r(n). \end{cases} \qquad (1)$$

4.2 Role of the TSA

The time-stamping authority (TSA) maintains the following databases (Figure 9):

1. the database $\mathcal{D}_c$ of the stamps of the current round.
2. the database $\mathcal{D}_p$ of the stamps of the previous round.
3. the database $\mathcal{D}_r$ of the stamps for rounds.
4. the complete database $\mathcal{D}_{\text{all}}$ of stamps.

The three first databases are considered to be on-line in the sense that any client can make requests into them at any moment. The fourth database is also stored but not on-line (it may be stored into an archive of CD-s). Requests to this database are possible, but costly (e.g., requiring human interaction). After the end of each

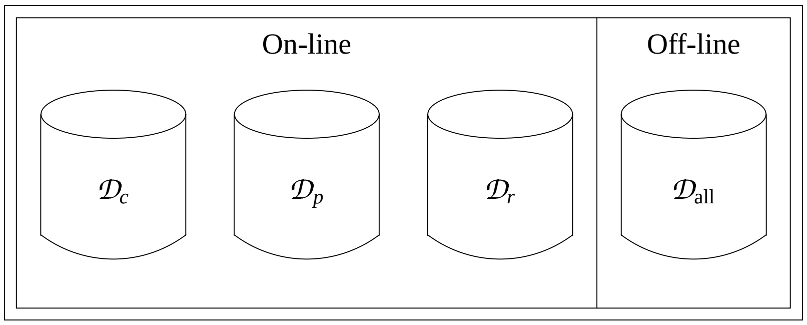


Figure 9: Databases kept by a TSA.

round, the stamps in $\mathcal{D}_p$ are stored to a separate CD (this process *may* be audited). Thereafter, $\mathcal{D}_p$ is emptied. The stamp R_r for the current round is computed, added to $\mathcal{D}_r$ and published in a newspaper. The database $\mathcal{D}_c$ is copied into $\mathcal{D}_p$ and a new database $\mathcal{D}_c$ is created.

Most of the time-stamping schemes proposed to date are vulnerable in sense that if the database $\mathcal{D}_{\text{all}}$ ceases to exist, one is no more able to perform relative temporal authentication. Even if the stamps of rounds are regularly (say weekly, as in the Digital Notary [Sur99] system) published in a newspaper, destruction of the database significantly reduces the accuracy — from (say) one second to one week. What we really expect from the time certificates is that:

- if m and n are "close" enough in time (lie in the same round), their one-way relationship can be established using $\text{Cert}(m)$ and $\text{Cert}(n)$;
- if m and n are not "close" enough in time (lie in different rounds), their one-way relationship can be established using $\text{Cert}(m)$, $\text{Cert}(n)$ and data published in the newspaper.

These requirements are satisfied if $\Lambda_M \circ \Lambda_N$ is used as the underlying authentication graph (cf Eq. (1)). Security of accumulated schemes does not depend on the integrity of the database $\mathcal{D}_{\text{all}}$.

4.3 Stamping Protocol

The *stamping protocol S* consists of two parts, the *stamp issuing protocol* which is executed when a client asks for a stamp, and the *stamp completion protocol* which is executed later, after the end of the round. The Protocol 4.1 is a slight modification of the protocol [BLLV98, Sect. 4.2] due to [BLS99], that minimizes the communication complexity of the stamp issuing.

Protocol 4.1 Stamp issuing protocol with linear linking scheme. Here, r is the current round number.

1. Client sends H_n to the TSA.
2. The TSA finds $L_n = H(H_n, L_{n-1})$, and adds the pair (H_n, L_n) to $\mathcal{D}_c$.
3. The TSA sends the client the message $(n, L_n, \mathrm{sig}_{\mathrm{TSA}}(n, L_n))$.
4. The client verifies the signature of TSA.

After the M requests have been answered, the TSA finishes the round by calculating the round stamp $\mathcal{L}_r$ and publishing $\mathcal{L}_r$ and his public verification key $\mathrm{VK}_{\mathrm{TSA}}$ in the newspaper. The client may now continue, during a limited period, with the *stamp completion protocol* (represented by Figure 4.2) in order to get the time certificate for H_n.

Protocol 4.2 Stamp completion protocol with linear linking scheme.

1. The client sends a request to the TSA.
2. The TSA answers by sending $(\mathrm{Cert}(n), \mathrm{sig}_{\mathrm{TSA}}(\mathrm{Cert}(n)))$ to the client.
3. The client checks, whether $\mathrm{head}(n)$ is consistent with L_n and whether $\mathrm{tail}(n)$ is consistent with $\mathcal{L}_{r(n)}$. Authenticated value $\mathcal{L}_{r(n)}$ can be found either from the newspaper or by requesting for their values from the on-line database $\mathcal{D}_r$ of the TSA.

We stress that every client who is interested in the legal use of some time certificate, should validate it during these two protocols. *In a relatively short period between the issuing protocol and between the completion protocol, the signature key of TSA is trusted to authenticate him* (cf Sect. 3.4) and therefore, his signature on an invalid $\mathrm{Cert}(n)$ can be used as an evidence in the court. But the client is responsible for doing it when the signature key of TSA can still be trusted. Later, the signature of TSA may become unreliable and therefore only the one-way properties can be used.

4.4 Verification Algorithm

Let $r(n)$ denote the round where nth stamp was issued. Assume that the verifier possesses two time-stamped documents $(H_{m_1}, \mathrm{Cert}(m_1))$ and $(H_{m_2}, \mathrm{Cert}(m_2))$ where $m_1 < m_2$. Let

$$v_{m_1,m_2} = (\mathcal{L}_{r(m_1)}, \mathcal{L}_{r(m_1)+1}, \cdots, \mathcal{L}_{r(m_2)-1})$$

be a tuple defined by $\vec{L}_{\mathrm{body}_\circ(m_1,m_2)} = (\vec{L}_{\mathrm{tail}(m_1)}, v_{m_1,m_2}, \vec{L}_{\mathrm{head}(m_2)})$. The algorithm V is represented by Protocol 4.3.

Protocol 4.3 Verification whether H_m was stamped before than H_n was.

INPUT: $(H_{m_1}, \mathrm{Cert}(m_1))$, $(H_{m_2}, \mathrm{Cert}(m_2))$.

1. If $r(m_1) < r(m_2)$, the verifier obtains from the TSA (or from the newspaper) the values v_{m_1,m_2} and $\mathcal{L}_{r(m_2)}$.
2. The verifier checks whether $\vec{L}_{\mathrm{head}(m_i)} \cup L_{m_i} \cup \vec{L}_{\mathrm{tail}(m_i)} \cup \mathcal{L}_{r(m_i)}$ is internally consistent, for $i \in 1, 2$.
3. The verifier verifies that the value of $\mathcal{L}_{r(m)}$ in v_{mn} is equal to the value of $\mathcal{L}_{r(m)}$ in $\mathrm{Cert}(m)$ and that v_{mn} is consistent with $\mathcal{L}_{r(n)-1}$.

4.5 Audit Protocol

Because of the possible legal importance of the stamps issued by the TSA, there should be some mechanism to audit TSA. One easy way to do it is to periodically ask stamps from the TSA and verify them. If these stamps are linked inconsistently, the TSA can be proven to be guilty. Also, there has to be a mechanism for the TSA to prove that he has not issued a certain stamp S in a certain round r. This can be done if the TSA presents all the stamps issued during the r-th round, shows that S is not among them and that the stamp for the r-th round, found by using these stamps and the linking rules, coincides with the published stamp.

In all the time-stamping systems described in this thesis, such disavowal protocol requires the TSA to present all stamps. Existence of a time-stamping system with succinct “negative proofs” is an important open problem.

5 ANTI-MONOTONE SCHEMES

In the current section we give a construction of a practical linking scheme with logarithmic upper bound to the number of issued time-stamps. Described scheme bases directly on the "binary linking schemes" paradigm of Buldas, Laud, Lipmaa and Villemson [BLLV98].

Definition 6 A simply connected graph $G = (V, E)$ is called *anti-monotone binary* (*AM graph*, for short) if (1) $\forall m$, $|E^{-1}(m)| \leq 2$; and (2) if $k < m \leq n$, $(k,n) \in E$ and $(\ell, m) \in E$ then $k \leq \ell$.

As it was shown in [BL98], the family of AM graphs is equal to the family of graphs, constructed recursively from the singleton graph $G = (1, \emptyset)$ by using the anti-monotone composition operator $\otimes$ (Figure 10). Clearly, $G_1 \otimes G_2 = F \circ [G_1 G_2]$, where $F = (\{1,2,3\}, \{(1,3),(2,3)\})$.

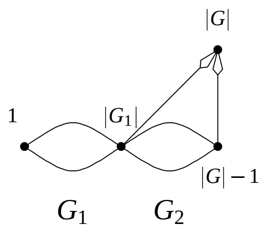


Figure 10: Anti-monotone composition $G := G_1 \otimes G_2$.

Definition 7 The *anti-monotone scheme* (*AM scheme*) $[\![G]\!]$ is constructed from AM graph G by introducing a vertex v' and an edge (v', v) for every vertex $v \in V(G)$ (Figure 11).

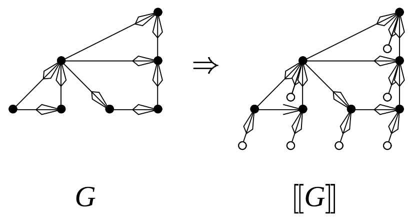


Figure 11: Construction of $[\![G]\!]$ from G.

Lemma 5.1 *The next claims are true.*

1. *Let G be an AM graph. For any m and n, $m < n$, there exists an unique shortest G-path from m to n.*

2. *If G_1 and G_2 are AM graphs, then $[\![G_1]\!] \circ G_2$ is an AM graph. Moreover, $[\![[\![G_1]\!] \circ G_2]\!] = [\![G_1]\!] \circ [\![G_2]\!]$.*

3. *Let G be an AM graph, and let $m_1, m_2 \in V(G)$, $m_1 < m_2$. Let $\mathcal{P}_1$ be the unique shortest path from m_1 to $n(G)$ and let $\mathcal{P}_2$ be the unique shortest path from* 1 *to m_2. Then $\mathcal{P}_1 \cap \mathcal{P}_2 \neq \emptyset$.*

4. *Let $[\![G_1]\!] \circ [\![G_2]\!]$ be the underlying AM graph. The number of stamped documents per round is equal to $|G_2| - 1 = |\mathcal{S}([\![G_2]\!])| - 1$.*

Proof. We shall prove the first claim by induction on the structure of graph G. The base ($G = (1, \emptyset)$) is trivial. Let $G = G_1 \otimes G_2$. If $m, n \in G_1$ or $m, n \in G_2$, then the claim holds by the induction hypothesis. If $m \in G_1$ and $n \in G_2$, then by induction hypothesis there is an unique shortest path from m to $|G_1|$ and and unique shortest path from $|G_1|$ to n. Concatenation of those paths is the unique shortest path from m to n. $\square$

The second claim says that accumulated time-stamping preserves the anti-monotonicity property.

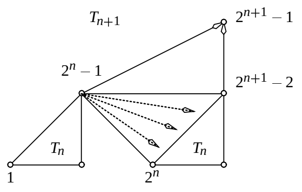
5.1 Concrete Scheme

The next infinite family T_n of AM graphs is a slight modification of the family defined in [BLLV98]:

1. T_1 consists of a single vertex which is labeled with the number 1. This vertex is both the *source* and the *sink* of the graph T_1.

2. Let T_n be already constructed. Its sink is labeled by $2^n - 1$. The graph T_{n+1} consists of two copies of T_n, where the sink of the first copy is linked to the source of the first copy, and an additional vertex labeled by $2^{n+1} - 1$ which is linked from the sink of the second copy. Labels of the second copy are increased by $2^n - 1$. The *source* of T_{n+1} is equal to the source of the first copy, the *sink* of T_{n+1} is equal to the vertex labeled by $2^{n+1} - 1$.

   Thereafter, add a link from the sink of the first copy to all the vertices of the second copy that have less than two in-going links. Note that there is now a double link from the sink of the first copy to the source of the second copy.



The sequence (T_n) defines an countable AM graph with the vertices labeled by natural numbers which contains each graph T_n as its initial segment. The graph T_5 is represented by Figure 12.

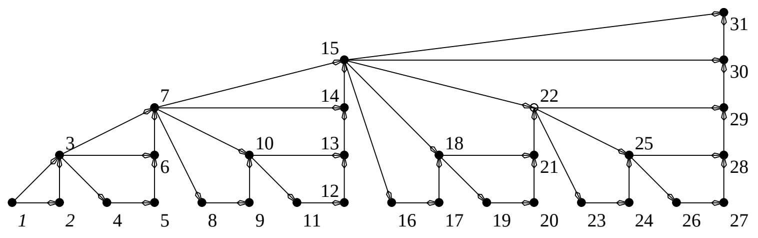


Figure 12: The AM graph T_5.

Theorem 5.2 *If $n > 2$ and $0 < a \leq b < 2^n$ then $d(a,b) \leq 3n-5$. (Proof is presented in Sect. 5.3)*

Denote by $\operatorname{ord} n$ the greatest power of 2 dividing n. In the AM graph (T_n) presented above, it is reasonable to enumerate stamps in the lexicographical order with pairs (m, p), where $0 \leq p \leq \operatorname{ord} m$ and $m > 0$. Then every vertex (m, p) has as predecessors the vertices enumerated with

$$f(m,p) := \begin{cases} (m-2^{p-1}, \operatorname{ord}(m-2^{p-1})) \ , & m = 2^p, \\ (m-2^p, \operatorname{ord}(m-2^p)) \ , & \text{otherwise} \end{cases}$$

and

$$g(m,p) := \begin{cases} (m, p-1), & p > 0, \\ (m-1, \operatorname{ord}(m-1)), & p = 0. \end{cases}$$

5.2 Accumulated Anti-Monotone Graphs

In Sect. 4 we presented an outline of a time-stamping system that fulfills our trust requirements. In the next we show how to make this system feasible by using an

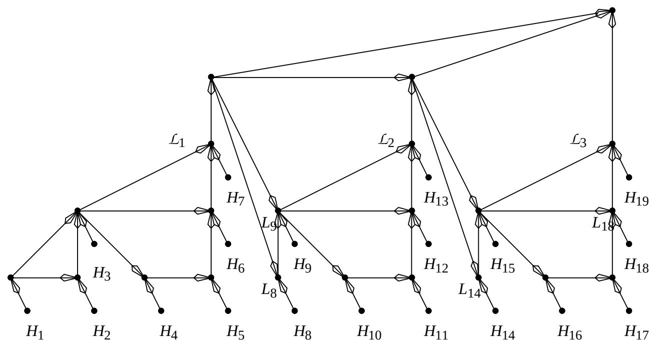


Figure 13: The "toy" accumulated scheme $[\![T_2]\!] \circ [\![T_3]\!]$.

AM scheme. Namely, we take $G_{N,M} = [\![T_N]\!] \circ [\![T_M]\!]$ as an underlying graph, where (say) $N = M = 40$ (a toy example is represented by Figure 13).

Thus, we can directly apply Def. 5 to get the definitions of head, tail, body and $\mathrm{Cert}(m)$ for the graph $[\![T_N]\!] \circ [\![T_M]\!]$.

Example 1 *Let the underlying graph be $[\![T_2]\!] \circ [\![T_3]\!]$ (Figure 13). Then,*

$$\begin{aligned}
\vec{L}_{\mathrm{head}(10)} &:= (L_8, H_9, H_{10}) \ , \\
\vec{L}_{\mathrm{tail}(10)} &:= (H_{11}, L_9, H_{12}, L_9, H_{13}) \ , \\
\vec{L}_{\mathrm{head}(19)} &:= (L_{14}, H_{15}, L_{18}, H_{19}) \ , \\
\vec{L}_{\mathrm{tail}(19)} &:= () \qquad\qquad \text{(empty string)} \ .
\end{aligned}$$

By the construction of T_M, length of the n-th time certificate

$$\mathrm{Cert}(n) = (n, \vec{L}_{\mathrm{head}(n)}, \vec{L}_{\mathrm{tail}(n)})$$

does not exceed $2d_{pt}(m) \cdot k$ bits, where k is the output size of the hash function H. It is easy to show that $\max_n d_{pt}(n) \approx 2 \cdot \log m$ [BL98] and thus

Lemma 5.3 *For any n, $|\mathrm{Cert}(n)| \leq 4 \cdot kM$.*

For example, if $M = 40$ and $k = 160$ bits then $|\mathrm{Cert}(n)| \leq 3200$ bytes.

By Lem .5.1, if $m < n$ then $\mathrm{tail}(m)$ and $\mathrm{head}(n)$ have a common element c which implies that $\mathrm{body}(m,n) \subseteq \mathrm{tail}(m) \cup \mathrm{head}(n)$ and thus, by Theorem 5.2, that $\mathrm{body}(m,n)$ is of logarithmic length. When $r(m) < r(n)$,

$$\mathrm{body}_\circ(m,n) = (m, \ldots, m', \xi(r(m)), \ldots, \xi(r(n)-1), n', \ldots, n) \ ,$$

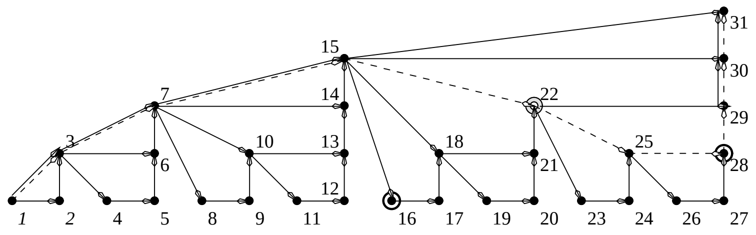


Figure 14: Time certificates in the case $G = T_5$ explained.

where the number of $\xi(j)$-s is logarithmic due to the fact that the time-stamps for rounds are linked together in a way similar to the linking of all time-stamps (Figure 13).

Example 2 *Let us look at the Figure 14. Here, the common element in* tail(16) *and* head(28) *is* 22*.*

Corollary 5.4 *Due to the similarity between the verification and the stamping procedure, for an arbitrary pair of stamped documents the number of steps executed (and therefore, also the number of stamps examined) during a single run of the verification protocol is $O(\log n)$.*

5.3 Proof of Theorem 5.2

In this section we will prove an upper bound for the length of the time certificates for the linking scheme described in Sect. 5. Let $e_k = 2^k - 1$, i.e. e_k is the number of the last vertex of T_k. To simplify the proof we add the vertex 0 to the scheme and link it with all the vertices e_i, $i \leq k$. As previously, let $d(a,b)$ denote the length of the shortest path between a and b. The equations $d(0,e_k) = 1$, $d(e_{k-1},e_k) = 2$ and $e_k - e_{k-1} = e_{k-1} + 1$ follow immediately from the definition.

Lemma 5.5 *If $0 < a \leq e_k \leq b$ then $d(a,b) = d(a,e_k) + d(e_k,b)$. If $e_{k-1} \leq a < e_k$ then $d(a,e_k) = d(a,e_k - 1) + d(e_k - 1,e_k)$.*

The claims above follow immediately from the structural properties of the linking scheme.

Lemma 5.6 *If $e_{k-1} \leq a \leq b < e_k$ then $d(a,b) = d(a - e_{k-1}, b - e_{k-1})$.*

Proof. This follows from the construction of T_k from the two copies of T_{k-1}. Here a and b are vertices in the second copy of T_{k-1} (or the last vertex of the first copy), and $a - e_{k-1}$ and $b - e_{k-1}$ are the same vertices in the first copy of T_{k-1} (or the vertex 0). □

Lemma 5.7 *If* $0 \leq a < e_k$ *then* $d(0,a) \leq k$.

Proof. Induction on k.
Base: $k = 1$. Then $a = 0$ and $d(0,a) = 0 < k$.
Step: $k > 1$. Observe the following cases:

- If $0 \leq a < e_{k-1}$ then the induction assumption gives $d(0,a) \leq k-1 < k$.
- If $e_{k-1} \leq a < e_k$ then $d(0,a) = d(0,e_{k-1}) + d(e_{k-1},a) = 1 + d(0,a-e_{k-1})$ by Lemma 5.6. Observe the following cases:
  - $a = e_k - 1$. Then $d(0,a) = 1 + d(0,a-e_{k-1}) = 1 + d(0,e_{k-1}) = 2 \leq k$.
  - $a < e_k - 1$. Then $d(0,a) = 1 + d(0,a-e_{k-1}) \leq 1 + (k-1) = k$ by induction assumption.

□

Lemma 5.8 *If* $0 < a \leq e_k$ *then* $d(a,e_k) \leq 2(k-1)$.

Proof. Induction on k.
Base: $k = 1$. Then $a = 1$ and $d(a,e_k) = d(1,1) = 0 = 2(k-1)$.
Step: $k > 1$. Observe the following cases:

- If $0 < a \leq e_{k-1}$ then $d(a,e_k) = d(a,e_{k-1}) + d(e_{k-1},e_k) \leq 2(k-2) + 2 = 2(k-1)$ by induction assumption.
- If $e_{k-1} < a \leq e_k$ then observe the following cases:
  - $a = e_k$. Then $d(a,e_k) = 0 \leq 2(k-1)$.
  - $a < e_k$. Then $d(a,e_k) = d(a,e_k-1) + d(e_k-1,e_k) = d(a-e_{k-1},e_{k-1}) + 1$ by the Lemma 5.6. Induction assumption now gives $d(a,e_k) = d(a-e_{k-1},e_{k-1}) + 1 \leq 2(k-2) + 1 < 2(k-1)$.

□

Proof. [Theorem 5.2] Induction on k.
Base: $k = 3$. In this case one can directly verify that $d(a,b) \leq 4$.
Step: $k > 3$. Observe the following cases:

- If $0 < a \leq b \leq e_{k-1}$ then the induction assumption gives us $d(a,b) \leq 3(k-1) - 5 < 3k - 5$.

- If $0 < a \leq e_{k-1} < b \leq e_k$ then $d(a,b) = d(a,e_{k-1}) + d(e_{k-1},b) \leq 2(k-2) + d(e_{k-1},b)$ by the Lemma 5.8. The following cases are possible:
  - $b = e_k$. Then $d(e_{k-1},b) = 2 < k-1$.
  - $b = e_k - 1$. Then $d(e_{k-1},b) = 1 < k-1$.
  - $b < e_k - 1$. Then the lemmas 5.6 and 5.7 give
    $d(e_{k-1},b) = d(0,b-e_{k-1}) \leq k-1$.

  Thus $d(a,b) \leq 2(k-2) + k - 1 = 3k-5$.
- If $e_{k-1} < a \leq b \leq e_k$ then observe the following cases:
  - $b = e_k$. Then $d(a,b) = d(a,e_k) \leq 2(k-1) < 3k-5$ by Lemma 5.8.
  - $b < e_k$. Then $d(a,b) = d(a-e_{k-1}, b-e_{k-1}) \leq 3(k-1)+5 < 3k-5$ by Lemma 5.6 and induction assumption.

□

As $\lceil \log b \rceil = k$ iff $e_{k-1} + 1 < b \leq e_k + 1$ we get $k < \lceil \log b \rceil + 1$ and thus

$$d(a,b) \leq 3\lceil \log b \rceil - 2 \ .$$

6 OPTIMAL GRAPHS

6.1 Optimal Anti-Monotone Schemes

By Lemma 5.1, if the underlying authentication graph G is anti-monotone, then $\mathrm{tail}(m)$ and $\mathrm{head}(n)$ have an intersection point for every m and n, $m < n$, that belong to the same round. Therefore, anti-monotone linking schemes guarantee that any two time certificates $\mathrm{Cert}(m)$ and $\mathrm{Cert}(n)$ together contain sufficient information for establishing a one-way relationship between L_m and L_n.

Though, in the case of the AM schemes $[\![T_n]\!]$ (Sect. 5.1), the certificate length $\mathrm{Cert}(m)$ is logarithmic in $|\mathcal{S}([\![T_n]\!])|$, the size may still become significant if the rounds are large. Thus, it is also important to find tightly, not only asymptoticly, optimal graphs.

As we know (cf page 36), the certificate length is intimately connected to the value $d_{pt}(G)$. Buldas and Laud [BL98] defined a new family T_k of AM graphs, that has minimal value of $d_{pt}(G)$ over all AM graphs (and hence, minimal certificate lengths over all AM graphs), as follows.

Definition 8 Let $U_1 = (1, \emptyset)$ be the singleton graph. For $n > 1$, let

$$U_n := ((U_1 \otimes U_{n-1}) \otimes U_{n-1}) \otimes U_{n-1} \quad .$$

The AM graph U_4 is represented by Figure 15.

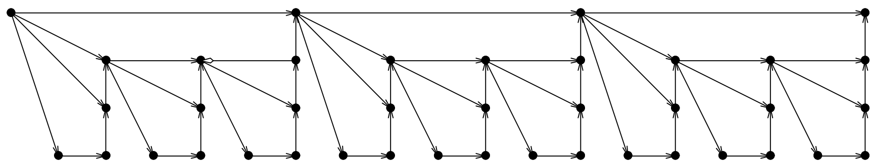


Figure 15: The optimal anti-monotone scheme U_4.

As it was shown in [BL98],

$$|U_n| = \frac{1}{2}3^n - \frac{1}{2}$$

and $d_{pt}(U_n) = 3(n-1)$. Hence,

$$\frac{d_{pt}(U_n)}{\log |U_n|} = \frac{3}{\log_2 3} \cdot \frac{1}{1 + o(1)} \quad .$$

On the other hand, it was also proven by Buldas and Laud, that for any AM graph G, if $d_{pt}(\llbracket G \rrbracket) = m$, then $|G| = 3^{m/3} \cdot O(1)$, and thus the family U_n is tightly optimal family of AM graphs.

6.2 Threaded Authentication Trees

Next we present a new construction of Buldas, Lipmaa and Schoenmakers [BLS99] that uses *threaded authentication trees*. When using Merkle's authentication tree of depth d, the length of a time-certificate is $k \cdot (d+1)$. We will proceed by adding extra edges to the authentication tree such that the resulting graph will be simply connected, but without enlarging the certificates.

Let V_d be the complete binary tree of depth d. We use the standard lexicographic enumeration of the vertices, representing the root by the empty string ϕ, and the left and right predecessors of a vertex v by $s0$ and $s1$, where s is the string representing v. We define the *threaded authentication tree* W_d as the authentication graph built from $V_d = (V, E)$ by

1. adding a vertex $*$ and, for each $s \in \{0,1\}^d$, adding an edge $(*, s)$,
2. for each $s \in \{0,1\}^d$, adding a vertex $s1$ and a corresponding edge $(s1, s)$,
3. for each s', s'' such that $s'1s'' \in \{0,1\}^d$, adding an edge $(s'0, s'1s'')$.

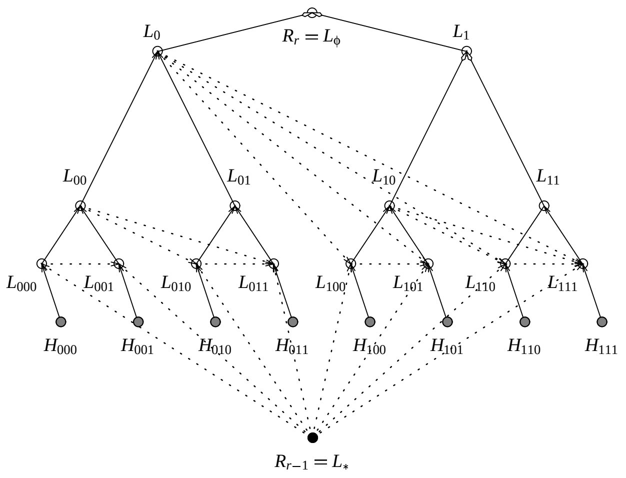


Figure 16: Threaded authentication tree of depth 3.

Example 3 (Figure 16) *Let H_i, $i \in \{0,1\}^3$, be the documents stamped during the rth round. Then,*

$$\begin{aligned}
\vec{L}_{\mathrm{head}(2)} &= (R_{r-1}, L_{00}, H_{010}) \ , \\
\vec{L}_{\mathrm{tail}(2)} &= (L_{011}, L_1) \ , \\
\vec{L}_{\mathrm{head}(6)} &= (L_{-1}, L_0, L_{10}, H_{110}) \ , \\
\vec{L}_{\mathrm{tail}(6)} &= (L_{111}) \ , \\
L_{010} &= H(2, \vec{L}_{\mathrm{head}(2)}) \ , \\
L_{110} &= H(6, \vec{L}_{\mathrm{head}(6)}) \ , \\
L_{n(G)} &= H(L_0, H(L_{10}, H(L_{110}, L_{111}))) \ .
\end{aligned}$$

The union of $\mathrm{Cert}(2)$ *and* $\mathrm{Cert}(6)$ *contains sufficient information to compute the one-way path*

$$\begin{aligned}
&(R_{r-1}, L_{010}, L_{01} = H(L_{010}, L_{011}), L_0, L_{110}, L_{11} = H(L_{110}, L_{111}), \\
&\qquad L_1 = H(L_{10}, L_{11}), R_r = H(L_0, L_1))
\end{aligned}$$

and verify its internal consistency. □

Hence, while in the case of AM schemes, the union of two time certificates contained the whole "proof" of one-way dependency, in the current case the proof itself is not contained in, but can still be computed from the union. That is, intuitively, the main source of the redundancy in the AM schemes compared to the new scheme.

Threaded authentication trees are simply connected. Moreover, for any $m, n \in W_d$, $m \le n \Rightarrow d(m,n) \le 2d-1$. Thus, the verification can be done very efficiently. Moreover, $|\mathrm{Cert}(m)| = (k+1)\log|\mathcal{S}(W_d)| + k$, $\forall m$. It will be proven in Sect. 6.3 that this is also the lower bound.

6.3 Optimality of Threaded Authentication Trees

Let

$$\mathrm{ap}(G) := \max_{v \in \mathcal{S}(G)} \min_{\mathcal{P}} \mathrm{ap}_{\mathcal{P}}(m) \ ,$$

where $\mathcal{P}$ ranges over the set of the root paths $(m, \ldots, k = n(G))$ starting from v. Let $h: \{0,1\}^* \to \{0,1\}^k$. Then, $|\mathrm{Cert}(m)| = \log|\mathcal{S}(G)| + k \cdot \min_{\mathcal{P}} \mathrm{ap}_{\mathcal{P}}(m)$, and thus $\max_m |\mathrm{Cert}(m)| = \log|\mathcal{S}(G)| + k \cdot \mathrm{ap}(G)$.

Therefore, a tight lower bound for $\mathrm{ap}(G)$ gives as a result tight lower bound for $\max_m |\mathrm{Cert}(m)|$. Below we prove that for a acyclic digraph G with sink, $\mathrm{ap}(G) \ge \log|\mathcal{S}(G)| + 1$. Concrete graph G achieving this bound was presented in Sect. 6.2.

Theorem 6.1 *Let G be a acyclic digraph with sink. Then* $\mathrm{ap}(G) \geq \log|\mathcal{S}(G)|+1$.

Proof. Let G be a acyclic digraph with sink. We show in several steps how to transform G by local modifications to a complete binary tree V_n, so that $|\mathcal{S}(V_n)| \geq |\mathcal{S}(G)|$ and $\mathrm{ap}(V_n) \leq \mathrm{ap}(G)$.

First step (eliminating fan-in $f > 2$). Replace any vertex with more than two in-coming edges with an (almost) balanced binary tree (Figure 17, 1). Let G_1 be the resulting graph. Trivially, $|\mathcal{S}(G_1)| = |\mathcal{S}(G)|$ and $\mathrm{ap}(G_1) \leq \mathrm{ap}(G)$.

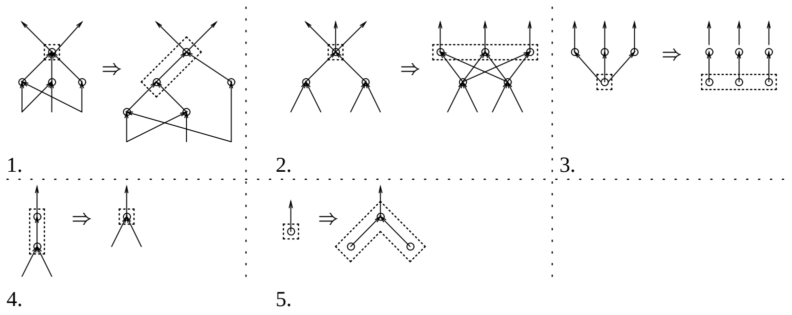


Figure 17: Transforming G into the complete binary tree.

Second step (eliminating fan-outs $f > 1$). Every vertex with fan-out > 1 can be replicated (Figure 17, 2). We can repeat the procedure until we get a graph G_2 with no internal vertex having fan-out > 1. Trivially, $|\mathcal{S}(G_2)| = |\mathcal{S}(G_1)|$ and $\mathrm{ap}(G_2) = \mathrm{ap}(G_1)$.

Third step (replicating the sources). Every source with fan-out > 1 can just be replicated (Figure 17, 3). The resulting binary tree G_3 has $|\mathcal{S}(G_3)| \geq |\mathcal{S}(G_2)|$ and $\mathrm{ap}(G_2) = \mathrm{ap}(G_2)$.

Fourth step (making G_3 complete). Any internal vertex $v \in V(G_3)$ with only one proper ancestor can be deleted (Figure 17, 4). Let the resulting graph be G_4. Trivially, $|\mathcal{S}(G_4)| = |\mathcal{S}(G_3)|$ and $\mathrm{ap}(G_4) = \mathrm{ap}(G_3)$.

Fifth step (balancing the tree). If G_4 is not yet a complete binary tree, there exists a source $v \in V(G_4)$ so that $d(v, n(G_4))$ is less than the height of G_3. We proceed by adding two proper ancestors to the tree (Figure 17). Let the resulting graph be G_5. Trivially, $|\mathcal{S}(G_5)| \geq |\mathcal{S}(G_4)|$ and $\mathrm{ap}(G_5) = \mathrm{ap}(G_4)$.

Thus for any graph G there exists a complete binary tree V_n such that $|\mathcal{S}(V_n)| \geq |\mathcal{S}(G)|$ and $\mathrm{ap}(V_n) = \mathrm{ap}(G)$. But $|\mathcal{S}(V_n)| = 2^n$ and $\mathrm{ap}(V_n) = n+1$, thus for any graph G, $\mathrm{ap}(G) \geq \mathrm{ap}(V_n) = \log|\mathcal{S}(V_n)|+1 \geq \log|\mathcal{S}(G)|+1$. $\square$

REFERENCES

[BdM91] Josh Benaloh and Michael de Mare. Efficient broadcast time-stamping. Technical Report 1, Clarkson University Department of Mathematics and Computer Science, August 1991.

[BHS92] Dave Bayer, Stuart A. Haber, and Wakefield Scott Stornetta. Improving the efficiency and reliability of digital time-stamping. In *Sequences'91: Methods in Communication, Security, and Computer Science*, pages 329–334. Springer-Verlag, 1992.

[BL98] Ahto Buldas and Peeter Laud. New linking schemes for digital time-stamping. In *The 1st International Conference on Information Security and Cryptology*, pages 3–14, 18–19 December 1998.

[BLLV98] Ahto Buldas, Peeter Laud, Helger Lipmaa, and Jan Villemson. Time-stamping with binary linking schemes. In Hugo Krawczyk, editor, *Advances on Cryptology — CRYPTO '98*, volume 1462 of *Lecture Notes in Computer Science*, pages 486–501, Santa Barbara, USA, August 1998. Springer-Verlag.

[BLS99] Ahto Buldas, Helger Lipmaa, and Berry Schoenmakers. Optimally efficient accountable time-stamping. Submitted, May 1999.

[BP97] Niko Barić and Birgit Pfitzmann. Collision-free accumulators and fail-stop signature schemes without trees. In Walter Fumy, editor, *Advances on Cryptology — EUROCRYPT '97*, volume 1233 of *Lecture Notes in Computer Science*, pages 480–494, Konstanz, Germany, May 1997. Springer-Verlag.

[BR93] Mihir Bellare and Philip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *Advances on Cryptology — CRYPTO '93*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249, Santa Barbara, USA, August 1993. Springer-Verlag.

[CS98] Ronald Cramer and Victor Shoup. Signature schemes based on the strong RSA assumption. Unpublished. Available from URL `http://www.inf.ethz.ch/personal/cramer/`, December 1998.

[DBP96] Hans Dobbertin, Antoon Bosselaers, and Bart Preneel. RIPEMD-160: A strengthened version of RIPEMD. In Dieter Grollman, editor, *Fast Software Encryption: Third International Workshop*, volume 1039 of *Lecture Notes in Computer Science*, pages 71–82, Cambridge, UK, 21–23 February 1996. Springer-Verlag.

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Inform. Theory*, IT-22:644–654, November 1976.

[GHR99] Rosario Gennaro, Shai Halevi, and Tal Rabin. Secure hash-and-sign signatures without the random oracle. In Jacques Stern, editor, *Advances on Cryptology — EUROCRYPT '99*, volume 1592 of *Lecture Notes in Computer Science*, pages 123–139, Prague, Czech Republic, 2–6 May 1999. Springer-Verlag.

[GMR88] Shafi Goldwasser, Silvio Micali, and Ronald L. Rivest. A digital signature scheme secure against adaptive chosen-message attacks. *SIAM Journal of Computing*, 17:281–308, 1988.

[Haw88] Stephen W. Hawking. *A Brief History of Time: From the Big Bang to Black Holes*. Bantam Books, April 1988.

[HS90] Stuart Haber and W. Scott Stornetta. How to time-stamp a digital document. In A. J. Menezes and S. A. Vanstone, editors, *Advances in Cryptology—CRYPTO '90*, volume 537 of *Lecture Notes in Computer Science*, pages 437–455. Springer-Verlag, 1991, 11–15 August 1990.

[HS91] Stuart A. Haber and Wakefield Scott Stornetta. How to time-stamp a digital document. *Journal of Cryptology*, 3(2):99–111, 1991.

[HS97] Stuart A. Haber and Wakefield Scott Stornetta. Secure names for bit-strings. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, pages 28–35, April 1997.

[Jus98a] Michael K. Just. *On the Temporal Authentication of Digital Data*. PhD thesis, Carleton University, December 1998.

[Jus98b] Michael K. Just. Some timestamping protocol failures. In *Symposium on Network and Distributed Systems Security*. Internet Society, March 1998.

[Lip98] Helger Lipmaa. IDEA: A cipher for multimedia architectures? In Stafford Tavares and Henk Meijer, editors, *Selected Areas in Cryptography '98*, volume 1556 of *Lecture Notes in Computer Science*, pages 248–263, Kingston, Canada, 17–18 August 1998. Springer-Verlag.

[Lip99] Helger Lipmaa. Accumulated time-stamping made simple. Manuscript, April 1999.

[Mer80] R. C. Merkle. Protocols for public key cryptosystems. In IEEE, editor, *Proceedings of the 1980 Symposium on Security and Privacy, April 14–16, 1980 Oakland, California*, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1980. IEEE Computer Society Press.

[MOV96] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[MQ97] Henry Massias and Jean Jacques Quisquater. Time and cryptography. Technical report, Université catholique de Louvain, March 1997. TIMESEC Technical Report WP1.

[NIS94] NIST. Announcement of weakness in the Secure Hash Standard (SHS). Technical report, May 1994.

[Pfi96] Birgit Pfitzmann. *Digital Signature Schemes*. Springer-Verlag, Berlin, Heidelberg, 1996.

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[Sim98] Daniel R. Simon. Finding collusions on a one-way street: Can secure hash functions be based on general assumptions. In Kaisa Nyberg, editor, *Advances on Cryptology — EUROCRYPT '98*, volume 1403 of *Lecture Notes in Computer Science*, pages 334–345, Helsinki, Finland, June 1998. Springer-Verlag.

[Sur99] Surety Technologies, Inc. Digital notary service. technical overview. Technical report, Surety Technologies, Inc., 1999.

INDEX

TURVALISED JA EFEKTIIVSED AJATEMPLISÜSTEEMID

Kokkuvõte

Viimastel aastatel on arvutisidevõrgud plahvatuslikult kasvanud. Tänu sellele edastatakse elektroonilisel teel üha enam ja enam ka dokumente, millel on otsene või kaudne juriidiline väärtus. Erinevalt paberdokumendist ei ole elektrooniline dokument üksüheselt seotud andmekandjaga, ning seega saab dokumenti vabalt kopeerida, muuta või kustutada. Seadusloome seisukohast mängib suurt rolli dokumentide autentsus: dokumentide omadus, mis seob dokumenti selle loojaga. Elektrondokumentide autentsust tagavaks krüptograafiliseks primitiiviks on *digitaalsignatuur*: protseduur, mis seab dokumendile ja signeerimisvõtmele turvaliselt vastavusse selle dokumendi signatuuri. Kahjuks ei ole digitaalsignatuur üksinda kasutatav, sest puudub meetod signeerimisvõtme ja signeerija isiku seostamiseks. Muu hulgas võib näiteks signeerija hiljem väita, et tema võti oli signeerimishetkel kompromiteerunud. Ehkki leiduvad standardmeetodid võtmekompromiteerumisest teatamiseks (võtmetühistuseks), on vaja ka meetodeid, mis võimaldaksid kindlalt väita, et võtmetühistus toimus enne signeerimist. Üldkrüptograafilised eelteadmised on toodud antud dissertatsiooni 1. peatükis.

Ajatembeldus (*time-stamping*) on organisatoorsete ja matemaatiliste meetodite kogu, mis võimaldab *suhtelist ajalist autentimist* ehk kindlaks teha, milline kahest esitatud dokumendist oli ajatembeldatud varem. Turvalise ja efektiivse ajatemplisüsteemi olemasolu korral saab seega tõestada, et võtmetühistus toimus enne kui signeerimine. Üldistades: on võimalik tõestada, et elektrooniline dokument oli olemas enne ajahetke.

Umbes kümme aastat tagasi nõudsid kõik teadaolevad ajatembeldussüsteemid täielikult usaldatava kolmanda osapoole (ajatemplikeskuse) olemasolu. Kõike, mida too osapool väitis, usuti. 1990. aastal näitasid Haber ja Stornetta, et täielikult usaldatav kolmas osapool ei ole tingimata vajalik. Nende leiutatud ajatemplisüsteem baseerus nn linkimisskeemidele, kus hiljem välja antud ajatempel on "ühesuunaliselt" sõltuv kõigist eelnevalt väljastatud ajatemplitest. Haberi ja Stornetta skeemi on hiljem muudetud nii turvalisemaks kui ka efektiivsemaks. Ülevaade vanematest linkimiskeemidest on toodud dissertatsiooni 2. peatükis.

Uus läbimurre tuli 1998. aastal, kui Buldas, Laud, Lipmaa ja Villemson pöörasid artiklis "Time-Stamping with Binary Linking Schemes" esmakordselt tähelepanu seadusloomes kasutatava ajatemplisüsteemi turvanõuetele. Sellises süsteemis peab olema võimalik avastada ning kolmandatele osapoolele tõestada ajatempliserveri tehtud vigu. Näidati, et eelnevad ajatemplisüsteemid ei võimalda tõestusi (ehk sertifikaate) efektiivselt esitada. Pakuti välja uus, neid tingimusi rahuldav, nn binaarsetel linkimisskeemidel põhinev ajatemplisüsteem. Dissertatsiooni 3. pea-

tükk käsitleb ajatemplisüsteemidele esitatavaid nõudeid. 4. peatükk kirjeldab neid nõudeid rahuldavat ajatemplisüsteemi. Edasi, 5. peatükis on näidatud, kuidas toodud ajatemplisüsteemi saab muuta efektiivseks.

Pakutud süsteemi on hiljem täiendatud. 1998. aasta lõpus formaliseerisid Buldas ja Laud binaarsetelt linkimisskeemidelt nõutud antimonotoonsuse omaduse ning esitasid minimaalse sertifikaadipikkusega antimonotoonse skeemi. 1999. aastal näitasid Buldas, Lipmaa ja Schoenmakers, et antimonotoonsuse tingimus on ebavajalik ning esitas uue süsteemi, mis on optimaalne üldisel juhul. 6. peatükk käsitleb esmalt lühidalt Buldas-Laud'i skeemi ning seejärel keskendub Buldas-Lipmaa-Schoenmakers'i skeemile.

ACKNOWLEDGEMENTS

Your presence is a present in my life.

I would like to thank my coauthor and friend Ahto Buldas for being thoughtful and for a very big number (which is hopefully prime) of discussions. His influence has been profound, I hope this influence has been mutual. Additionally, I will always remember our chess matches in various pubs around the world.

I am very thankful to most of my colleagues in Küberneetika AS. Here I will only name Monika Oit. A “thank you” goes to my coauthors, Peeter Laud and Jan Villemson, for being creative, but also for being good friends. The same words go for Berry Schoenmakers. I hope he will have more chances to visit our beautiful country. I would like to thank my advisor Mati Tombak for giving me freedom to choose my own area of research. It seems that my choice was correct. He has always been very supportive and patient. Many thanks goes to Stuart Haber. Although he is almost always busy, discussions with him have helped us to understand time-stamping.

At last, I would like to thank the people whom I love or whom I have loved for their presence in my life. You know who you are.

CURRICULUM VITAE

Helger Lipmaa

Citizenship: Estonian Republic.
Born: April 8, 1972, Pärnu, Estonia.
Marital status: single.
Address: Sauga alevik 27-5, Pärnu maakond, 80043 Estonia,
phone: (37 2) 6654241, e-mail: helger@cyber.ee

Education

1979–1987 Pärnu Ülejõe Gümnaasium.
1987–1990 Pärnu Koidula Gümnaasium.
1990–1999 Faculty of Mathematics, University of Tartu

Special Courses

1994 — DAIMI, Århus University
1996 — DAIMI, Århus University
1997 — New Trends in Computer Science and Information Technology, Palmse
1997 — School on Natural Computation, Turku
1998 — Parallel and Quantum Computation, Palmse, Estonia
1998 — Summer School in Cryptography and Data Security, Århus, Denmark
1999 — Fourth Estonian Winter School in Computer Science, Palmse, Estonia.

Professional employment

1995–1995 Junior researcher at Institute of Computer Scence, University of Tartu.
1995–1996 Senior assistant at Institute of Computer Science, University of Tartu.
1996–1997 Junior researcher at Institute of Cybernetics, Tallinn.
1997–1999 Senior research engineer at Küberneetika AS, Tallinn.

Scientific work

Coauthor of books “Infosüsteemide turve I. Turvarisk” (1997) and “Infosüsteemide turve II. Turbetehnoloogia” (1998). Main papers: “IDEA: An Architecture for Multimedia Architectures?” (1998) “Time-Stamping with Binary Linking Schemes” (1998). Invited presentations: Information Security Training Seminar of Institute of Cybernetics (1996), Information Security Training Seminar of Küberneetika AS (1997), Autumn School of Young Physicists (1998). Has published surveys in quantum computation and cryptography.

CURRICULUM VITAE

Helger Lipmaa

Kodakondsus: Eesti Vabariik.
Sünniaeg ja -koht: 8. aprill, 1972, Pärnu, Eesti.
Perekonnaseis: vallaline.
Aadress: Sauga alevik 27-5, Pärnu maakond, 80043 Eesti,
tel.: (2) 6654241, e-post: helger@cyber.ee

Haridus

1979–1987 Pärnu Ülejõe Gümnaasium.
1987–1990 Pärnu Koidula Gümnaasium.
1990–1999 Tartu Ülikooli matemaatikateaduskond.

Erialane enesetäiendus

1994 — DAIMI, Århus University
1996 — DAIMI, Århus University
1997 — New Trends in Computer Science and Information Technology, Palmse
1997 — School on Natural Computation, Turku
1998 — Parallel and Quantum Computation, Palmse
1998 — Summer School in Cryptography and Data Security, Aarhus, Taani
1999 — Fourth Estonian Winter School in Computer Science, Palmse.

Erialane teenistuskäik

1995–1995 nooremteadur, Arvutiteaduse Instituut, Tartu Ülikool.
1995–1996 vanemassistent, Arvutiteaduse Instituut, Tartu Ülikool.
1996–1997 nooremteadur, Küberneetika Instituut, Tallinn.
1997–1999 vanemteadur, Küberneetika AS, Tallinn.

Raamatute “Infosüsteemide turve I. Turvarisk” (1997) ja “Infosüsteemide turve II. Turbetehnoloogia” (1998) kaasautor. Peamised artiklid: “IDEA: An Architecture for Multimedia Architectures?” (1998), “Time-Stamping with Binary Linking Schemes” (1998). Kutsutud ettekanded: Küberneetika Instituudi andmeturbe teabepäev (1996), Küberneetika AS andmeturbe teabepäev (1997), Noorte Füüsikute Sügiskool (1998). On avaldanud ülevaateartikleid kvantarvutitest ja krüptograafiast.