Digital timestamps for dispute settlement in electronic commerce: generation, verification, and renewal

Kanta Matsuura
Interfaculty Initiative in Information Studies, University of Tokyo
Komaba 4-6-1, Meguro-ku, Tokyo 153-8505, JAPAN.
Email: kanta@iis.u-tokyo.ac.jp

Hideki Imai
Institute of Industrial Science, University of Tokyo
Komaba 4-6-1, Meguro-ku, Tokyo 153-8505, JAPAN.
Email: imai@iis.u-tokyo.ac.jp

Keywords: Digital notary, time-stamp, dispute settlement, long-lived authenticity

Abstract: Digital notary is receiving more and more attention as a social infrastructure in the network society. Digital time-stamping is a simple but very important form of the notary, and can be used to provide long-term authenticity. Its function is essential for dispute settlement in electronic commerce. This importance motivated us to have a clear survey on how to generate/verify digital time-stamps.

In order to make longer use of digital time-stamping functions, we may have to update or renew our time-stamps. The renewal must be completed before the expiry of the current time-stamp. However, the congestion problem of the renewal request has not yet been studied well. So the survey is followed by an investigation of renewal feasibility. The investigation uses a simple model in which the service provider's mind and the users' mind are independently represented by two different parameters. The condition for accepting renewal requests without queueing is discussed by using the two parameters.

Another issue will occur when we submit high-dimensional contents to be time-stamped and the components of the contents are updated independently in later time; memory requirement on the client depends on the submission strategy. A discussion on different strategies is given as well.

1 INTRODUCTION

In the network society, digital notary is receiving more and more attention as a social infrastructure. Digital time-stamping (Benaloh and de Mare, 1992), (Lipmaa, 1999) is a simple but very important form of the notary, essentially involving only an authentic appendage of time to a digital document. One of the most important functions of time-stamping is the support of long-lived authenticity(Bayer et al., 1992). This function is essential for dispute settlement in electronic commerce. Suppose a digitally signed audit log of electronic-commerce transactions, and suppose that the public verification key for it has been unfortunately revoked. Then we want to be sure about the actual authenticity of the log; up to which entry of the log, the signer's secret key had been kept alive without being revoked? A digital timestamping system is a powerful tool for solving this problem.

If we really need digital timestamping as our social infrastructure, the availability of it would be a critical problem; for example, queueing delay might result in expiry in the worst case. However, congestion of the requests for renewing time-stamps has not been studied well. This is an *availability issue on the server's side*.

We may have an *availability/efficiency issue on the client's side* as well. When we submit high-dimensional contents to be time-stamped and the components of the contents are updated independently in later time, memory requirement on the client depends on the submission strategy.

The purpose of this paper is to survey digital time-stamping and to give the first attempt at time-stamp evaluation regarding (1) its renewal feasibility and (2) submission strategy of high-dimensional contents. Specifically, Section 2 gives the survey, Section 3 investigates (1) the renewal feasibility, and then (2) the submission strategy is discussed in Section 4. Section 5 concludes the paper.

2 GENERATION AND VERIFICATION OF TIME-STAMPS

2.1 Advantage/Disadvantage of Digital Technology

The most remarkable advantage of digital technology is its noise-free property: digital data can maintain their original bit-strings as long as needed. This, in turn, suggests a disadvantage of digital technology; digital society could not make use of natural degradation process which often supports legal proof and police/detective activities. Authentic produce of time or proof of age in the digital world is not a trivial task. Thus we can say that digital time-stamping is a quite important service to support our network life, especially for electronic commerce in which we may face financial damages due to time-related problems.

2.2 Formal Definitions

Time-stamping can be defined in several ways.

Definition 1 (from Reference (Benaloh and de Mare, 1992))

A *time-stamping system* with security parameter N consists of a set of participants $P = \{p_1, p_2, \cdots, p_m\}$ which are synchronously communicating processes together with a triple of protocols $(\mathcal{S}, \mathcal{C}, \mathcal{V})$. The *stamping protocol* $\mathcal{S}$ proceeds in rounds and allows each participant to *post* one or more messages during that round. The *claimant* protocol $\mathcal{C}$ can at any subsequent time be run by a participant who posted a message to convince any "honest" participant running the *verification* protocol $\mathcal{V}$ of the round in which the message was actually posted. Finally, there is no polynomial P such that one or more "dishonest" participants running substitute protocols $\mathcal{S}'$ and $\mathcal{C}'$ can, with probability $1/P(N)$, falsely convince an honest verifier running protocol $\mathcal{V}$ that a message was submitted during a given round.

Definition 2 (from Reference (Lipmaa, 1999)) A *time-stamping system* consists of a set of principals with the *time-stamping authority (TSA)* together with a triple $(\mathcal{S}, \mathcal{V}, \mathcal{A})$ of protocols. The *stamping protocol* $\mathcal{S}$ allows each participant to post a message. The *verification algorithm* $\mathcal{V}$ is used by a principal having two stamps to verify the temporal order between those time-stamps. The *audit protocol* $\mathcal{A}$ 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.

The former mentions a security parameter and considers the temporal order round by round while the latter considers relative temporal order between any two time-stamps. Both of them include stamping and verification protocols. In the following subsection, we will review those protocols currently available. Examples of their implementations will appear later in 2.4.

2.3 Taxonomy

2.3.1 Simple Protocols

A simple solution for time-related notary issue is a "digital safety-deposit box"(Haber and Stornetta, 1991) [1]. Whenever a client wants to obtain a time-stamp on a digital document, he/she transmits the document to a time-stamping authority (TSA). TSA records the date and time when the document was received and stores a copy of the document for safekeeping. When one wants to check the integrity to verify the time-stamp, he/she simply makes a comparison with the copy stored by TSA. This solution has at least the following problems:

- Since the document itself is transmitted, an eavesdropper can see it.
- Clients must worry not only about the trust of TSA but also about the security of TSA which might be compromised.
- The scalability is insufficient and the TSA's storage capacity might be easily exhausted.

The first privacy issue can partly solved by submitting not a document itself but a collision-resistantly hashed value of the document(Haber and Stornetta, 1991).

Another simple protocol was discussed also in (Haber and Stornetta, 1991); TSA appends the current date and time t to the document X, generate his/her digital signature $s = \mathrm{sig}_{\mathrm{TSA}}(t, X)$ on the whole, and then returns t and s to the client. t is represented with a specific granularity[2] depending on the application. When one wants to verify the time-stamp, he/she verifies the digital signature by using TSA's public key. The weakness of this protocol is in that it relies on the security of the digital signature (and its underlying public-key infrastructure) too much. One can hardly

[1] A more general notion called "electronic safe-deposit box" is proposed in (Matsuura et al., 1997).

[2] For most applications in the paper-based world, the granularity is a day, or several hours. Toward a future network society, IETF (Internet Engineering Task Force) currently considers a finest granularity of a micro second in "Internet X.509 Public Key Infrastructure Time Stamp Protocol"(Adams et al., 2001).

imagine that this protocol is used to extend the lifetime of a digitally-signed document. Therefore, we do not suppose simple protocols in the following discussions on long-lived authenticity.

2.3.2 Linking Protocols

To solve the problem in TSA's trust and security, the clients may want the TSA to link all time-stamps together into a chain or a graph by using a collision-resistant hash function H. In the case of a linear linking chain (Haber and Stornetta, 1991), the time-stamp for the hashed value $H_n = H(X_n)$ of the n-th document X_n is

$$s = \mathrm{sig}_{\mathrm{TSA}}\left(n, t_n, \mathrm{ID}_n, H_n, L_n\right) \quad (1)$$

where t_n is the appended time, ID_n is the client's identifier, and L_n is the linking information defined as

$$L_n = \left(t_{n-1}, \mathrm{ID}_{n-1}, H_{n-1}, H\left(L_{n-1}\right)\right). \quad (2)$$

This linking information is periodically publicized in a media which is widely and publicly recorded. A popular example is to print the linking information in a newspaper. Fortunately, a newspaper

- is physically dated (*i.e.*, the issueing date is tamper-resistantly printed) and examined for signs of after-the-fact tampering,
- contains *temporal data* which could not be predicted in advance (weather information, stock-market information, sport results, *etc.*),
- functions as a global public record whose long-term availability at many locations (*e.g.*, major libraries) makes tamper-proof property far more better, and
- could be finally recorded even on library microfilms.

Each time period between two adjacent publications is called a *round*.

When one wants to verify the time-stamp, he/she recomputes a series of hashed values starting from the preceding publicized value. Intermediate hashed values are obtained from the TSA. Each signature of the TSA is verified. After comparing the claimed linking information with the recomputed value, the recomputation goes on to the next publicized value. The verification finishes after checking the next publicized value is correct. Thus the verification is sandwiched by two publications.

To improve the efficiency, one can change the linking mechanism into a tree-based structure as described in Fig. 1. All the client need to store are the original document and the $\lceil \log_2 N \rceil$ hashed values (Bayer et al., 1992). The granularity of the time-stamp is, however, gets as coarse as the length of a round; we couldn't tell the order of requests processed in the same round. Toward more advanced efficiency and optimality, one can consult (Buldas et al., 1998), (Buldas and Laud, 1998), (Lipmaa, 1999), and (Buldas et al., 2000). Toward ordering in the same round, recent works are found in (Jinnai and Sakurai, 1999) and (Ono et al., 2000).

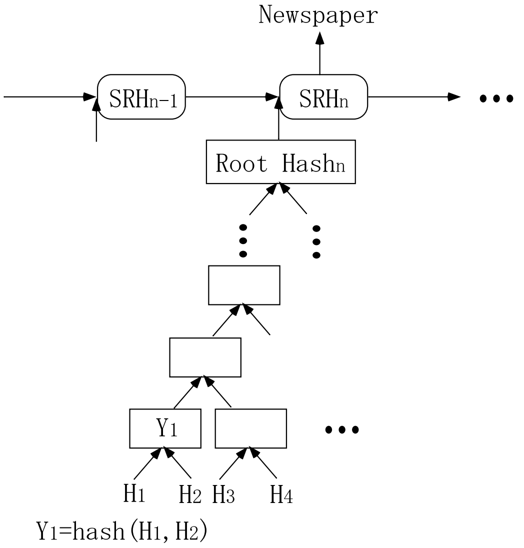

Figure 1: Linking protocol with a binary-tree structure. All the inputs and default padding values are hashed pair by pair (for example, Y_1=hash(H_1, H_2)) to result in a *root hash* value in the round. Finally, the root hash is hashed together with the previous *super-root hash (SRH)* value to produce the SRH of the current round. SRH is publicized periodically.

2.4 Projects and Services

Although after minor modifications, the protocols reviewed in the previous subsections are really used in existing projects or services.

Academic or national projects abroad tend to use linking protocols as follows.

TIMESEC (Belgium: 1996–1998) deploys a linking protocol using tree structure with hash functions such as SHA-1(Menezes et al., 1996) and RIPEMD-160(Dobbertin et al., 1996).

Cuculus (Estonia: 1997) deploys a linear linking protocol based on hash function and digital signature.

PKITS (Spain: 1997–1998) deploys a linking protocol in which not a single but multiple TSAs operate. Digital signature is not used.

By contrast, domestic digital notary projects in Japan (one was by Ministry of Justice in 1998, the other was by New Media Development Association during 1997–1998) uses simple protocols.

Simple protocols are found also in some commercial services:

“Digital Notary Service” by Surety (USA: 1992–) deploys a linking protocol using tree structure.

Firstuse.com (USA: 1998–) deploys a simple protocol which does not use digital signature.

“e-TimeStamp” by DigiStamp (USA: 1998–) deploys a simple protocol which does not use digital signature.

“Stamper” by I.T. Consultancy (UK: 1995–) deploys a simple protocol which uses digital signature. Time-stamps are publicized on the Web page of I.T. Consultancy.

3 RENEWAL OF TIME-STAMPS

3.1 Long-Lived Authenticity

Future dispute-settlement in the court would require authentic reference of digital documents. Since digital data in themselves can be easily copied and modified without detection, digital signature is used to provide the authentic reference. If the digital signature can be completely trusted, almost every reference issue can be solved; what we have to do is just to attach required attribute (who, to whom, where, when, how, *etc.*) to the original document and then sign on the whole. We may commit this procedure to an authority, if necessary.

Currently popular infrastructure for digital signature gives, however, too short lifetime to the signature; most existing public-key certificate services provide a certificate which expires in a few years. This means that we need an additional mechanism to support long-term authenticity (say, fifty years) preferably by extending the lifetime of digital signatures.

Digital time-stamping gives a solution to this problem(Bayer et al., 1992). Let a digital document X and a signature σ on it be time-stamped altogether. Some time later, the signature may become invalid, for any of reasons including

- the compromise of the signer’s private key,
- the expiry of the certificate,
- more and more powerful computation which makes the key length unsafe, and
- the discovery of basic flaws in the signature scheme or key-transfer mechanisms.

Even in these cases, the pair (X, σ) constitutes a valid signature since the time-stamp ensures that it was created at a time when only the legitimate user could have produced it.

If the lifetime of the time-stamp s on (X, σ) is insufficient, the same technique can be used to extend it. It should be noted that we must time-stamp not on s alone but on (s, X, σ) as a whole(Bayer et al., 1992).

3.2 Evaluation Model

Our question is whether a request for time-stamp renewal can be accepted without queueing which might cause a sad expiry. For simplicity, this paper considers the following model:

- Let us consider the maximum number of requests which can be processed without queueing in the round when they are received. We refer to this number or capacity as *round capacity*. The round capacity of the k-th round is represented by N_k.
- The time-stamp service provider makes good estimation of future demand and enough effort to provide sufficient capacity to cover *new* requests *excluding renewal requests*. The ratio of “the number of the new requests” to “the round capacity” is a constant α. We refer to $1-\alpha$ as *capacity-margin parameter*. We assume $0 < \alpha < 1$ since the provider makes enough effort (investment, *etc.*) due to his/her responsibility in a social infrastructure.
- The ratio of
  - the number of renewal requests which come during a single round

  to

  - the number of the existing time-stamps

  is a constant β. We refer to this ratio as *renewal-demand parameter*. We assume $0 < \beta < 1$.

Let us suppose that an initial round has produced αN_0 time-stamps. Then at the next round, in order to accept all the renewal requests without queueing,

$$\alpha\beta N_0 \leq (1-\alpha)N_1 \tag{3}$$

must hold. If this holds and we proceed to the next round, again to accept all the renewal requests,

$$\beta\left(\alpha\beta N_0 + \alpha N_1\right) \leq (1-\alpha)N_2 \tag{4}$$

must hold. This condition can be described as

$$\sum_{j=0}^{k-1} \alpha\beta^{k-j} N_j \leq (1-\alpha)N_k \tag{5}$$

in the k-th round.

3.3 Constant Demand

First we examine a stable society where the demand of time-stamp αN_k is constant and therefore N_k is constant. In this situation, the condition (5) becomes

$$\frac{\beta\left(1-\beta^k\right)}{1-\beta} \leq \frac{1-\alpha}{\alpha}. \tag{6}$$

When k gets larger enough to assume $\beta^k = 0$, we can see that (6) approaches

$$\alpha + \beta \leq 1. \tag{7}$$

If (7) holds, we can satisfy (6) for any k. This is because the condition (7) and $0 < 1 - b^k < 1$ leads to

$$\frac{1-\alpha}{\alpha} \geq \frac{\beta}{1-\beta} > \frac{\beta\left(1-\beta^k\right)}{1-\beta}. \tag{8}$$

3.4 Linearly-Expanding Demand

Next we consider a situation where the demand of time-stamp αN_k increases linearly and thus $N_k = N_0 + \gamma k$ (γ: positive constant). In this case, the condition (5) becomes

$$\alpha\beta^{k+1}N_0+\alpha\beta\gamma+\frac{\beta^2-\beta^{k+1}}{1-\beta} \geq (\alpha+\beta-1)(N_0+\gamma k). \tag{9}$$

Again, if $\alpha + \beta \leq 1$, the condition (9) is satisfied for any k. This is because

$$\alpha\beta^{k+1}N_0 + \alpha\beta\gamma + \frac{\beta^2-\beta^{k+1}}{1-\beta} > 0 \tag{10}$$

and

$$N_0 + \gamma k > 0. \tag{11}$$

4 SUBMISSION OF HIGH-DIMENSIONAL CONTENTS

Suppose that a client submits a high-dimensional content $X_n = (x_{1,n}, x_{2,n}, \cdots, x_{m,n})$ whose components $x_{i,n}(i = 1, 2, \cdots, m)$ may be updated and resubmitted in later time. The client may submit it in a way that

(Content Submission) Submit the content to be time-stamped as a whole, *i.e.*, submit a hashed value $H_n = H(X_n)$.

(Updated-Bulk Submission) Submit the updated components to be time-stamped as a whole, *i.e.*, submit a hashed value $H_n = H(x_{i_1,n}, x_{i_2,n}, \cdots, x_{i_k,n})$ when the i_1-th, the i_2-th, $\cdots$, and the i_k-th components are updated.

(Component Submission) Submit the updated components to be time-stamped individually, *i.e.*, submit hashed values $h_{i_j,n} = H(x_{i_j,n})$ ($j = 1, 2, \cdots, k$) when the i_1-th, the i_2-th, $\cdots$, and the i_k-th components are updated.

Our concern is which is the most efficient way of submission with respect to the memory cost on the client.

If the client must keep older components and their time-stamps, the least efficient way is obviously the component submission. Let M be a memory cost caused by a single time-stamp and $|x|$ be a size of each component which is assumed to be common among all the components for simplicity. Suppose a client submits an m-dimensional content and updates the first and the second components later. Then the content submission costs $m|x| + M + 2|x| + M = (m+2)|x|+2M$. The updated-bulk submission costs the same while the component submission costs more: $m|x| + mM + 2|x| + 2M = (m + 2)(|x| + M)$.

If the client does not have to keep older components and their time-stamps, the situation is more complicated. The client can discard a timestamp and the time-stamped object once all the components included in the object have been updated. Therefore, in this case, the most efficient way is not trivial. An obvious fact is that the updated-bulk submission could not be the most efficient because it costs more than the content submission. Thus our future work shall include a stochastic model which can be used for the analysis of this problem.

5 CONCLUDING REMARKS

We firstly gave a survey on how to generate and verify digital time-stamps. As an infrastructure, availability is quite important for time-staming service. So we examined availability issues on the server's side, as well as on the client's side.

As a server-side issue, renewal feasibility of time-stamps was studied. In a simple model, we investigated two situations: one keeps the demand of time-stamp constant and the other considers a linear increase of the demand. The result recommends that service providers estimate not only new-request demand but also renewal-request demand. If the providers set capacity-margin parameter larger than renewal-demand parameter, clients can renew their time-stamps promptly. The renewal-demand parameter is defined as the ratio of "the number of renewal requests which come during a single round" to "the number of *all the existing* time-stamps".

As a client-side issue, three submission strategies of high-dimensional contents were compared: *content submission*, *updated-bulk submission*, and *component submission*. The result depends on whether the client must keep older components and their time-stamps. Future works may include a stochastic analysis of this issue.

REFERENCES

Adams, C., Cain, P., Pinkas, D., and Zuccherato, R. (2001). "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)". RFC3161.

Bayer, D., Haber, S. A., and Stornetta, W. S. (1992). “improving the efficiency and reliability of digital time-stamping”. In Capocelli, R., DeSantis, A., and Vaccaro, U., editors, *Sequences’91: Methods in Communication, Security, and Computer Science*, pages 329–334, Berlin, New York, Tokyo. Springer-Verlag.

Benaloh, J. and de Mare, M. (1992). “efficient broadcast time-stamping”. Technical Report 1, Clarkson University Department of Mathematics and Computer Science. (Extended Abstract).

Buldas, A. and Laud, P. (1998). “new linking schemes for digital time-stamping”. In *Proc. of the 1st International Conference on Information Security and Cryptology*, pages 3–14, Seoul.

Buldas, A., Laud, P., Lipmaa, H., and Villemson, J. (1998). “time-stamping with binary linking schemes”. In *Advances in Cryptology — CRYPTO’98*, pages 486–501, Berlin, New York, Tokyo. Springer-Verlag. Lecture Notes in Computer Science 1462.

Buldas, A., Lipmaa, H., and Schoenmarkers, B. (2000). “optimally efficient accountable time-stamping”. In Imai, H. and Zheng, Y., editors, *Proc. of Third International Workshop on Practice and Theory in Public Key Cryptosystems (PKC 2000)*, Lecture Notes in Computer Science 1751, pages 293–305, Berlin, New York, Tokyo. Springer-Verlag.

Dobbertin, H., Bosselaers, A., and Preneel, B. (1996). “RIPEMD-160: A Strengthened Version of RIPEMD”. In *Proc. of Fast Software Encryption Cambridge Workshop*, pages 71–82, Berlin, New York, Tokyo. Springer-Verlag. Lecture Notes in Computer Science 1039.

Haber, S. A. and Stornetta, W. S. (1991). “how to time-stamp a digital document”. *Journal of Cryptology*, 3(2):99–111.

Jinnai, M. and Sakurai, K. (1999). “an absolute time-stamping protocol using relative time-stamping by distributed authorities”. In *Proc. of Computer Security Symposium’99 (CSS’99)*, pages 1–6. (in Japanese).

Lipmaa, H. (1999). *“Secure and Efficient Time-Stamping Systems”*. PhD thesis, University of Tartu.

Matsuura, K., Zheng, Y., and Imai, H. (1997). “electronic safe-deposit box”. *Abstracts of the 20th Symposium on Information Theory and Its Applications (SITA97)*, pages 385–388.

Menezes, A., van Oorschot, P., and Vanstone, S. (1996). *Handbook of Applied Cryptography*. CRC Press, Inc., Boca Raton, Florida.

Ono, Y., Kikuchi, H., and Nakanishi, S. (2000). “the fair distributed time-stamping protocol based on packets transmission delay”. In *The 2000 Symposium on Cryptography and Information Security(SCIS2000)*, Okinawa. (in Japanese).