Institut Eurécom
Department of Corporate Communications
2229, route des Crètes
B.P. 193
06904 Sophia-Antipolis
FRANCE

Research Report RR-05-140

A Distributed Time Stamping Scheme

May 9 2005

Bonnecaze Alexis

A. Bonnecaze is with CNRS-EURECOM FRE2660, Department of Corporate Communications, and LIF, Université de Provence, France.

Tel : (+33) 4 93 00 26 26
Fax : (+33) 4 93 00 26 27
Email : alexis.bonnecaze@eurecom.fr

[1]Institut Eurécom's research is partially supported by its industrial members: Bouygues Télécom, Fondation d'entreprise Groupe Cegetel, Fondation Hasler, France Télécom, Hitachi, ST Microelectronics, Swisscom, Texas Instruments, Thales

Abstract

In this paper, we define a trusted reliable distributed time stamping scheme. This scheme is based on a network of servers managed by administratively independent entities.

keywords: Time Stamping System, Distributed System, One Way Accumulator.

1 Introduction

The Internet Economy is compared today to the Industrial Revolution in potential scope and size. This new economy relies on highspeed networks based on the Internet Protocol, Internet applications, new marketing and business tools, and electronic intermediaries to increase the efficiency of Internet-driven markets. Successful e-commerce depends upon proving the identity of persons on-line and linking them to a transaction without repudiation. Digital signatures help to provide ongoing assurance of authenticity, data integrity, confidentiality and non-repudiation. A lot of applications in various domains like patent submissions, intellectual property or electronic commerce also require to make use of a time-stamping service. Indeed, it is very important to be able to certify that an electronic document has been created at a certain date.

In this paper, we propose a new time-stamping scheme based on the notion of threshold signature. Our scheme involves n independent servers and relies on two main protocols $\mathcal{S}$ and $\mathcal{V}$. The first one is the time-stamping protocol the aim of which is to create a time-stamp certificate (the token). The second one is executed by a verifier in order to convince herself that the dating of the document is correct.

The first time-stamping schemes were presented in the early 90's by Haber and Stornetta [HabSto91] and Benaloh and de Mare [BenMar91]. In the years that follow, a lot of new schemes were proposed and their security analysed [MasQui97], [BulLauLipVill98], [BulLip], [MasSerQui99], [Jus]. They make use of a trusted Time-Stamping Authority (TSA) which is expected to securely time-stamp an electronic document. The Network Working Group specifies, in the Internet X.509 PKI Time-Stamp Protocol (rfc3161), an Internet standards track protocol for time-stamping. This RFC describes the format of a request sent to a TSA and of the response that is returned. It also establishes several security-relevant requirements for TSA operation, with regards to processing requests to generate responses. According to this rfc, the TSA is required:

1. to use a trustworthy source of time,
2. to include a trustworthy time value for each time-stamp token,
3. to include a unique integer for each newly generated time-stamp token,
4. to produce a time-stamp token upon receiving a valid request from the requester, when it is possible,

5. to only time-stamp a hash representation of the datum.

As the first message of this mechanism, the requesting entity (called here the client) requests a time-stamp token by sending a request to the Time Stamping Authority. As the second message, the Time Stamping Authority responds by sending a response to the requesting entity. The TSA must sign each time-stamp message with a key reserved specifically for that purpose. A TSA may have distinct private keys, e.g., to accommodate different policies, different algorithms, different private key sizes or to increase the performance.

Most of the protocol use the concept of trusted third party even though it may be difficult to build a third party server that can be trusted. Indeed a server may be corrupted or victim of denial of service attacks (DoS). Moreover, the problem may not have a malicious origin but a hardware or software origin. As we will see in the next section, one of the aim of existing protocols is to prevent the server from failing.

In fact, we claim that time-stamping schemes relying on a unique third party server, cannot be trusted. Therefore our objective in this paper is to propose a time-stamping scheme using a multiserver architecture which can be shortly described as follows:
The protocol uses n third party servers. For each time-stamping request, k servers among the n servers are randomly chosen to process the request. The client is able to create a time-stamp token after having received at least λ responses (called shares) from the servers. The value λ is a security parameter that has to be optimized. A share σ_i contains the dating of the document signed by the server i while the time-stamp token represents a multisignature constructed from λ shares.

The paper is organized as follows. Section 2 briefly analyses the weaknesses of existing protocols. In Section 3, we give the required properties of our model and we introduce the notion of threshold signature in Section 4. In Section 5, we present our time-stamping scheme and discuss its security.

2 Existing protocols and their weaknesses

Most of the existing systems rely on a centralized server model that has to be trusted. The idea behind existing time-stamping schemes is to prevent the server from forging fake time-stamp tokens by linking the tokens in a chronological chain (see for example [HabSto91]). Periodically, a token is published on an unalterable and widely witnessed media like a newspaper. This scheme offers the following advantages:

- The publication provides us with an absolute time.
- After a token has been published at time t, the server cannot forge a fake time-stamp token former to time t.

- Since tokens are linked in a chronological chain, we can obtain a relative dating of the requests submitted between two publications.

However, this scheme has the following drawbacks:

- The publication step is costly and not convenient.
- Before the next publication, the server can tamper the tokens which have been issued since the last publication.
- The entire chronological chain must be stored for verification.
- Finally, centralized systems are very vulnerable to Denial of Service attacks.

Obviously, the chain structure is not the best one to obtain an efficient scheme. In order to reduce the amount of information to be stored, most of the protocols use a binary tree structure also called Merkle Tree (recall that a Merkle Tree is a construction introduced by Ralph Merkle in 1979 [Mer79] to build secure authentication and signature schemes from hash functions). This method allows us to reduce by a logarithmic factor the amount of information to be stored and the verification consists in rebuilding a half of the tree. However, protocols using linking informations are not always accurate and efficient. This is trivially the case when the number of time-stamped documents is very small while the frequency of publication is very low (typically a week). In that case, the accuracy of the time-stamp may not satisfy the client. Notice also that a scheme using a binary tree is not efficient when the number of documents is not close to a power of 2. The worst case being reached when this number is $2^n + 1$.

Recently, Blibech et al [BliGab05] proposed to use a new structure called skip-list (developped by Bill Pugh [Pug90]). A skip-list is a probabilistic data structure that can be used in place of a balanced tree. Algorithms for insertion and deletion in skip lists are simpler and faster than equivalent algorithms for balanced trees.

There exists in the literature an other kind of scheme, using accumulator functions, which is also very interesting. Accumulator functions [BendeMar93] represent an (algebraic) alternative to the aforementioned data structures. Using these functions, the verification process can be done in just one operation. Moreover, the amount of information that has to be stored does not depend on the number of time-stamped documents. Accumulator functions which are generally used are modular exponentiation.

The first scheme which takes into account the problem of a denial of service attack is presented in [BonLiaGabBli05]. This scheme does not eliminate the process of publication which is done electronically with the help of a replicated data base. The verification process is efficient but servers have to interact in order to time-stamp documents.

In the next section, we set the requirement properties that our protocol must offer.

3 Design requirements

Our aim is to design a secure, reliable and efficient time-stamping system.

1. security requirements:
   (a) independence from any administrative entity (like a country, a multinational company,...);
   (b) resistance against a Denial of Service (DoS);
   (c) robustness against an attack involving less than $n/3$ servers. It is known that any protocol can be made provably secure (without any cryptographic assumptions) if and only if less than one third of the involved parties are corrupted;
2. reliability
   (a) resistance against material failures;
3. efficiency
   (a) deliver an absolute time with an *a priori* fixed error of Δt;
   (b) a self-contained time-stamp token, which allows the client to prove the datation without any additional information;
   (c) a robust, simple and efficient verification protocol;
   (d) no publication process.

4 Signature scheme

Our protocol needs a special signature scheme which shares some properties of a multisignature and a threshold signature. A multisignature allows a group of signers to convince a verifier that every member of that group participated in signing. The goal of a threshold signature is different in the sense that it must convince the verifier that at least λ signers belonging to a given group participated in signing. Here, anonymity of the signers is required and it is just the number of signers which is important. Note that there exist many threshold signature schemes (see for example [DesFra89] [DesFra91] [Rab98][Sho00] [BreSteSzy02] [Bol03]) but most of the existent protocols do not take into account the possible presence of malicious signers. Generally, shares of a secret are distributed to n parties with the help of a third party or by running an interactive protocol among all parties. Signing a message requires the knowledge of more than λ shares. Parties must group together to reconstruct the secret using, for example, Lagrange interpolation.

Recently, spontaneous ad-hoc groups have been introduced. These schemes do not require any group secret. In [Wei04], V. Wei proposes such a scheme which is well adapted to ad-hoc networks with minimal infrastructure. There is no setup

stage but signers must know the set of signers and the set of public keys of all n members in order to calculate their shares. Signers have to solve a system of λ equations with λ unknowns to obtain their shares and the verification process consists in verifying λ equalities. Therefore, the length of the signature, the complexity of the calculation of the shares and of the verification all linearly depend on the value λ of the threshold.
In most schemes, signers know the subgroup of signers before calculating their shares and they can interact all together when needed. In our scheme, signers don't know in advance the set of active servers and anonymity of signers is not required. Also, for efficiency and security reasons, we prefer to avoid interaction between servers. We just require the signature to be short, robust and easily verifiable. BLS signature scheme [BonLynSha02] can easily be adapted in order to obtain a multisignature (see [Bol03]). This last scheme seems to be the most appropriate to fit our needs. It uses groups where the Computational Diffie-Hellman (CDH) problem is hard while the Decisional Diffie-Hellman (DDH) problem is easy. These groups are called Gap Diffie-Hellman (GDH) groups. Boldyreva proved in [Bol03] that her multisignature is secure against existential forgery under chosen message attack in the random oracle model. Moreover, this signature scheme leads to very short signatures of length approximately 160 bits. This algorithm uses intrinsic properties of elliptic curves and have no equivalent in more conventional groups like $\mathbf{Z}^p$. Note that BLS algorithm is much faster, in practice, than other kinds of signature algorithms that produce smaller signatures [PatCouGou00], [CouFinSen01]. Moreover it does not seem to be covered by patents

5 A time-stamping scheme

In this section, we propose a distributed time stamping scheme which takes into account the problem of a denial of service attack. The idea of using a distributed system is not new. A distributed solution to archive documents has been studied in [ManGiuBak01] and distributed time-stamping systems have also been studied, for example in [BenMar91], [HabSto91] or [Tak99]. However, these studies were neither able to design a secure and efficient system, nor able to give an answer to the problem of denial of service. Recently, Bonnecaze et al proposed in [BonLiaGabBli05] a new distributive scheme which is secure under the different kind of mentioned attacks. This scheme uses accumulator functions and has an efficient verification process. The main drawbacks of this system are the existence of a costly replicated (electronic) data base and the great interaction between the servers when the number of requests is high.

Our system lies on a distributed network of n servers the logical topology of which can be represented as a star with the client at its center. There is no interaction between the servers and there is no data base. The time-stamp is self contained in the sense that it contains all the required informations to prove the datation.
Among the n servers, we suppose that at most d are not working properly. They can

either be malicious or out of order. They can also operate in malicious collusion.

We call active a server which is involved in the calculation of a particular time stamp. To minimize the amount of calculations, only k ($k < n$) servers are active for a given document. These servers are randomly chosen by the client in order to maintain the server load balancing.

Time is discretized in rounds of length Δt and servers and clients are synchronized regularly. Each round is identified by an absolute date. For example, a round can be identified by: August 4^{th} 2005 at 12.04am.

In order to describe our protocol, we adopt the following notations:
Let G be an additive group of a prime order p and G' be a multiplicative group of the same order p. Let P be an arbitrary generator of G, h a hash function and let $H\ : \{0,1\}^* \rightarrow G^*$ be a hash function mapping arbitrary binary strings to the elements of $G \setminus \{0\}$, where 0 denotes the identity element of G. We assume that both G and G' are GDH groups. It means that it is hard, given the three random group element (P, aP, bP) to compute abP but it is easy, given (P, aP, bP, cP) to decide whether equality $c = ab$ holds (aP denotes P added to itself a times). A map $e : G \times G \rightarrow G'$ is called a cryptographic bilinear map if it satisfies the following properties:

1. bilinearity: for all $P,\ Q \in G$ and $a,\ b \in \mathbf{Z},\ e(aP,\ bQ) = e(P,\ Q)^{ab}$
2. non-degeneracy: $e(P, P)$ is a generator of G' and therefore $e(P, P) \neq 1$
3. computable: there exists an efficient algorithm to compute $e(P, Q)$ for all $P, Q \in G$.

Each server S_i chooses randomly its secret key x_i and obtain its public key $v_i := x_i P$. The value λ is the threshold of the scheme: for a given document, a time-stamping token can be calculated as soon as λ servers correctly time stamped the document and delivered their shares. Since we assume that at most d servers do not work properly, λ must be greater than d. Indeed, if $\lambda < d$, malicious servers could collude to produce a fake token. When the client receives λ correct shares, she calculates the multisignature as a proof of the time-stamp.
More precisely, the protocol can be described in the following way. Each client c who wish to submit a request has to do the following operations :

1. calculate the hashed value of the document D, denoted $h(D)$;
2. determine (pseudo) randomly k active servers;
3. sign the hash to form the request $r := (h(D))_c$
4. send the request r to the k servers.

Note that the choice of the k servers is done randomly, not for a security reason, but to balance the load of the servers.

Each server S_i which receives a request r during round t constructs $M := (h(D)||t)$ and calculates its share $\sigma_i = x_i H(M)$ of the signature. It sends to the client the values $i,\ \sigma_i,\ M$. When the client has received λ correct shares (signing the same M), it computes the multisignature $\sigma = \sum_i \sigma_i$ and the token is $T := (h(D),\ t,\ \lambda,\ L,\ \sigma)$, where L is the list of the λ signers.

5.1 Verification scheme

The verification process is very simple since there is no need to check any data base. The token is really self contained. Given T, the verifier computes $v := \sum_i v_i$ and verifies that $e(P,\ \sigma) = e(v,\ H(M))$.

6 Discussion

This scheme is secure if $\lambda > d$ since there are, by hypothesis, at most d malicious servers. In this case, if the client obtains λ correct signatures on the same value $(h(D),\ t,\ \lambda)$, she knows that t is the right date for document D. If $k > \lceil 2n/3 \rceil$, the client is assured to obtain at least λ correct shares.

The main disadvantage of using a distributed structure instead of a one server model is that the value of Δt (the accuracy of the time-stamping system) cannot be less than a few seconds, say 10 seconds. This accuracy depends on the network properties and the synchronization. Note, however that one server models, having an accuracy of one second, published their token every week (or two weeks). Their certified Δt is therefore equal to one or two weeks.

Our protocol requires a simple threshold signature and does not need anonymity of the signers. Moreover, servers don't interact and therefore are not able to know the list of signers. Hence, a protocol like the one of V. Wei [Wei04], Bresson et al [BreSteSzy02], or using a secret sharing is not appropriate.

The drawback of this protocol is the use of an expensive special hash function Map-to-point that encodes finite strings to elements of G. ZSS [ZhaSafSus03] signature is more efficient than BLS [BonLynSha02] but it cannot be modified in order to obtain a multisignature scheme. Barreto et al propose in [BarKim01] a fast hash algorithm that maps arbitrary messages onto points of an elliptic curve defined over a finite field of characteristic 3. Their algorithm called Map3Group runs in time $O(m^2)$ for curves over F_{3^m} (instead of $O(m^3)$). The computational complexity derives from the squaring of a field element and from solving a system of linear equations over $\mathbf{F}_{3^m}$ with coefficients in $\mathbf{F}_3$. When $m = 163$, Map3Group takes 0.164 seconds in a 550 MHz Pentium III processor.

6.1 Efficiency

We denote **Pa** the pairing operation, **Pm** the point scalar multiplication on G, **Ad** the point addition on G, and **MTP** the MapToPoint hash operation. After a

given document has been sent, the following operations are to be done:
Each server: 1 **MTP**, 1 **Pm**
The client: λ **Ad** to construct σ; 2λ **Pa**, 1 **MTP** to check the λ shares
The verifier: λ **Ad**, 2 **Pa**, 1 **MTP**.
We can see that the protocol is not heavy for the servers and the verifier. The client has to operate more operations (that can be done in parallel) if she wants to verify that the shares she receives are correct.

7 Conclusion

Our scheme represents a solution to the problem of denial of service attacks and publication process. The probability of success of a denial of service attack is reduced thanks to the distributed structure of the system. Furthemore, the use of a threshold signature allows the token to be as powerful as λ tokens, each of which being calculated by one server, while the verification is done in one step. Thanks to the signature protocol based on groups constructed from elliptic curves, the time-stamp token is short. Moreover, its length is constant and does not depend on the parameters n, k and λ.

8 Acknowledgement

The author would like to thank P. Michiardi and R. Molva for helpful discussions.

References

[BarKim01] P. S. L. M. Barreto and H. Y. Kim, *Fast hashing onto elliptic curves over fields of characteristic 3*, Cryptology ePrint Archive, Report 2001/098

[BenMar91] J. Benaloh and M. de Mare *Efficient Broadcast time-stamping* Technical Report 1, Clarkson University Department of Mathematics and Computer Science, August 1991. url = "citeseer.ist.psu.edu/benaloh91efficient.html

[BendeMar93] J. Benaloh and M. de Mare *One-Way Accumulators: A Decentralized Alternative to Digital Signatures* Advances in Cryptology–EUROCRYPT'93. LNCS, vol.765, pp.274–285, Springer–Verlag, 1994.

[BliGab05] K. Blibech, A. Gabillon, *Authenticated dictionary based on Skip Lists for time stamping systems*, submitted to vldb 2005, PdD workshop.

[Bol03] A. Boldyreva, *Efficient threshold signature, multisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme*, Practice and Theory in Public Key Cryptography – PKC'2003, Lecture

Notes on Computer Science 2567, Springer-Verlag (2003), pp. 31–46. See also Cryptology ePrint Archive, Report 2002/118.

[BonLiaGabBli05] A. Bonnecaze, P. Liardet, A. Gabillon, K. Blibech, *A Distributed time stamping scheme*, SAR 2005.

[BonLynSha02] D. Boneh, B. Lynn, H. Shacham, *Short signatures from the Weil pairing*, Advances in Cryptology – Asiacrypt'2001, Lecture Notes on Computer Science 2248, Springer-Verlag (2002), pp. 514–532.

[BreSteSzy02] E. Bresson, J. Stern, and M. Szydlo. *Threshold ring signatures and applications to ad-hoc groups*, In: CRYPTO 2002, LNCS 2442, pp. 465–480. Springer-Verlag, 2002.

[BulLauLipVill98] A. Buldas, P. Laud, H. Lipmaa and J. Villemson *Time-stamping with Binary Linking Schemes*, Advances on Cryptology — CRYPTO '98, Lecture Notes in Computer Science, Springer-Verlag, (1998), 486–501.

[BulLip] A. Buldas and H. Lipmaa *Digital Signatures, Timestamping and the Corresponding Infrastructure* url = citeseer.ist.psu.edu/article/buldas98digital.html

[CouFinSen01] N. Courtois, M. Finiasz and N. Sendrier, *How to achieve a McEliece-based digital signature scheme*, Proceedings of Asiacrypt'2001, to appear. Available online at http://eprint.iacr.org/2001/010/.

[DesFra89] Y. Desmedt and Y. Frankel, *Threshold cryptosystems*, In Proc. CRYPTO '89 (LNCS 435), pp. 307–315, 1989

[DesFra91] Y. Desmedt and Y. Frankel, *Shared generation of authenticators and signatures*, In Advances in Cryptology - Crypto '91, Proceedings (Lecture Notes in Computer Science 576), pp. 457-469. Springer-Verlag, 1992.

[HabSto91] Stuart Haber and W. Scott Stornetta *How to Time-Stamp a Digital Document*, Journal of Cryptology: the Journal of the International Association for Cryptologic Research 3(2), pp. 99–112, 1991.

[Mer79] R. Merkle, *Secrecy, authentication, and public key systems*, Ph.D. dissertation, Dept. of Electrical Engineering, Stanford Univ., 1979.

[ManGiuBak01] P. Maniatis T.J. Giuli and M. Baker *Enabling the Long-Term Archival of Signed Documents through Time Stamping* Computer Science Department, Stanford University, Technical Report, 2001, url = citeseer.ist.psu.edu/maniatis01enabling.html

[MasSerQui99] H. Massias and X. Serret and J. Quisquater *Timestamps: Main issues on their use and implementation* In Proceedings of IEEE 8th International Workshops on enabling Technologies: Infrastructure for Collaborative Enterprises - Fourth International Workshop on Enterprise Security, pages 178-183, June 1999. ISBN 0-7695-0365-9.

[MasQui97] H. Massias and J. Quisquater, *Time and cryptography* Technical report, Université catholique de Louvain, March 1997. TIMESEC Technical Report WP1.

[Jus] M. Just *Some Timestamping Protocol Failures*, url = citeseer.ist.psu.edu/just98some.html

[PatCouGou00] J. Patarin, N. Courtois and L. Goubin, *Quartz, 128-bit long digital signatures*, NESSIE submission, 2000. Available online at http://www.cryptonessie.org/.

[Pug90] W. Pugh, *Skip Lists: Skip lists: A probabilistic alternative to balanced trees*, Communications of the ACM, 33(6):668–676, June 1990.

[Rab98] T. Rabin, *A Simplified Approach to Threshold and Proactive RSA*, CRYPTO 1998: 89-104

[Sho00] V. Shoup, *Practical threshold signatures*, Eurocrypt 00, 2000.

[Tak99] A. Takura, S. Ono, S. Naito *A Secure and trusted Time Stamping Authority* Proceedings of IWS 99, 1999, pp.123-128.

[Wei04] V. K. Wei, *A Bilinear Spontaneous Anonymous Threshold Signature for Ad Hoc Groups*, Cryptology ePrint Archive, Report 2004/039

[ZhaSafSus03] F. Zhang, R. Safavi-Naini, W. Susilo, *Attack on Han et al.'s ID-based Confirmer (Undeniable) Signature at ACM-EC'03*, Cryptology ePrint Archive, Report 2003/129.