



Jupiter House, Station Road, Cambridge, UK, CB1 2JD
Tel: +44 (0)1223 723600 Fax: +44 (0)1223 723601

In response to the Public Consultation on

ELECTRONIC TRANSACTIONS ACT

STAGE III: REMAINING ISSUES

Version 1.0
Tuesday, August 16, 2005

Table of Contents

• About nCipher Corporation Limited	3-4
• Executive Summary	5
• Response to the Questions to the ETA Public Consultation III	6-9
• Digital Versus Electronic Signatures	10-12
• The Electronic Audit Trail	13
• List of Users Using nCipher Time Stamping and Digital Signing Solution [CONFIDENTIAL]	[Extracted]
• nCipher and National Time Authority in Singapore and Japan	14-15
• Summary	16

Appendix

RFC 3161	17
----------	----

About nCipher Corp Ltd

nCipher is a leading provider of cryptographic security, enabling our customers to meet the challenges of verifying identity, protecting data and complying with security regulations. nCipher's solutions provide a unified approach to cryptographic management providing strict access controls and high assurance trusted processing, overcoming traditional issues of scalability, performance and weak platform security.

The world's leading organizations work with nCipher to protect security critical systems such as web site infrastructure, online banking and payment processing networks, PKI, web services, databases and digital rights management schemes.

Founded in 1996 by brothers Alex van Someren and Dr. Nicko van Someren, nCipher has developed Internet security products that optimize the use of cryptography to protect critical points of risk— Digital Certificate infrastructures, authentication systems, databases, and payment processing networks—across the enterprise.

nCipher's specific areas of expertise include secure, high performance cryptography, scaleable and secure key management infrastructures, application interoperability and hardware design. From its global headquarters and R & D centre in Cambridge, UK, nCipher has pioneered a number of industry 'firsts' including netHSM - the world's first FIPS-validated network HSM, Secure Execution Engine (SEE) Technology, Impath network protocol and nToken client authentication.

nCipher's products are particularly well suited to organizations with high volumes of security-sensitive transactions and those that act as guardians of valuable information, such as financial institutions, government agencies, technology companies, pharmaceutical companies, utilities and online service providers.

nCipher currently has a 59% global market share in FIPS-certified hardware security modules [source: Infonetics, 2003] and is highly regarded by Gartner, KPMG and other analysts.

nCipher is the winner of the Microsoft Security Solution of the Year award 2004. The company is listed on the London Stock Exchange (LSE:NCH) and is a member of the FTSE TechMARK and FTSE4Good indices with offices in Cambridge, UK; Boston, Washington, Hamburg and Tokyo. For more information on nCipher, visit www.ncipher.com

Contact nCipher:

Mr. Scott Mustard – Sales Director (Time Stamping Solutions)
Scott@ncipher.com
DID: +1 781 994 4086 (4086)

Ms. Brigitte Tay – Country Manager (Singapore and Malaysia)
Btay@ncipher.com
DID: +65 6726 9885
HP: +65 92717356

Brief corporate history

- 1996 Founded in Cambridge, UK, by brothers Alex and Dr Nicko van Someren
- 1997 Delivers industry's first secure cryptographic accelerator – nFast
 Open US headquarters in Boston, MA
- 1998 nFast named 'Hot product of the year' by Data communications
 Develops nForce secure SSL accelerator
- 1999 nForce wins Network Computing Editor's Choice Award
 Launches nShield HSM, first crypto-accelerated HSM to receive FIPS 140-1 Level 3
 validation
 Total private equity reached \$25 million in mezzanine round
- 2000 Launches KeySafe – key management console for enterprise key management
 Completes £99 million IPO and is listed on London Stock Exchange (LSE:NCH)
- 2001 First HSM vendor to support new AES encryption standard
 Launches CodeSafe secure application toolkit, featuring unique
 SEE technology
 Strategic partnership with Broadcom announced
- 2002 OEM relationship with F5 Networks
 Launches payShield, enabling entry into the payment processing market
- 2003 First hardware vendor to achieve Microsoft Gold Certified Partner status
 Launches netHSM, the world's first FIPS-validated network HSM
 Launches DSE 200 document sealing and time-stamping appliance
- 2004 Launches SecureDB database encryption solution
 Launches pdfProof PDF document signing and time-stamping solution
 Winner of Microsoft Security Solution of the Year award 2004
 nCipher joins Trusted Computing Group - alongside AMD, HP, IBM, Intel, Microsoft,
 Sony & Sun

Executive Summary

We live in an era of significant change in daily business practice, one in which the “paper trail”, the physical evidence that links past actions to current events, has largely disappeared. Like it or not, the vast bulk of our daily commercial communications are conducted wholly in cyberspace.

The audit trail, the primary means by which we document corporate and governmental events, now resides in the email server or the “work flow” document repository. The shift from paper to bits has been accomplished without much thought for preservation of electronic audit trails.

Records management best practices require that we deploy safeguards to protect the electronic audit trail. It is relatively straightforward to automate one’s interoffice memo system; simple email will suffice. That does not allow one to assume that the archival copies behind each message are preserved in a form suitable for use as evidence should the need arise.

Traditionally, this has been accomplished by preserving paper copies, often stamped as “official” or “archival” copies and set aside in corporate “document archives”. Our challenge as we further automate our business processes is to preserve sufficiently compelling electronic data as to replace paper proofs.

Digital signature technology based on public key cryptography offers significant promise as an effective solution. Early implementations focused on authentication at the individual level. While the technology works, the logistical difficulty of deployment often outweighs the benefits. Cost of ownership and maintenance of the complex network of registration and validation systems required for large scale deployment are prohibitive.

Applications of digital signature technology are evolving that embed the technology in applications that digitally sign on behalf of organizational units, and companies. Just as businesses deploy automated signing systems for paper based check processing, special purpose devices that sign digitally can be deployed in support of paper elimination with secure audit trail preservation without the cost and logistical headaches of a broad deployment of public key technology.

This paper examines the application of secure public key based digital signature technology to the process of “signing by proxy”, that is, using public key cryptographic technology to seal documents, data objects, and system logs.

List of question from Annex E (pg 137 - 139) of the Public Consultation paper – Stage III Remaining Issues.

Question	Comments	reference
<p>Part IV – Electronic Government (Pg 51-72 of the Consultation paper) A soft copy of the Consultation paper may be downloaded from : http://www.ida.gov.sg/idaweb/pnr/index.jsp (under Consultation Papers) or http://www.agc.gov.sg (under Publications).</p> <p>NOTE: For more info on the ETA 1998 of Section 9 and 47, please visit : http://agcvldb4.agc.gov.sg/</p>		
Q10	<p>Do you have any comments on the proposed amendments to section 9 of the ETA in Annex B?</p>	<p>Yes. Electronic records require provable integrity for 9(1)(b) and 9(1)(c). Digital signatures from accredited certificate authorities with time-stamps for non-repudiation are critical towards this end.</p> <p>A digital signature and a time-stamp allows for both parties of a contract to attest to the records integrity and temporal properties. It does not guarantee endorsement of the content of the record.</p>
Q11	<p>Do you have any comments on the proposed amendments to section 47 of the ETA in Annex B?</p>	<p>Yes. nCipher agrees with all of the additions to section 47 but would like to highlight that a time-stamp could be added as a possible measure to 47(2)(b) to ensure the validity certificate associated to the signature and date and time of the electronic record.</p>

Q12	Should Singapore adopt a single provision on electronic originals or provide specifically for different situations in which electronic communications may be used as a functional equivalent of paper or other non-electronic forms? ³⁵⁸ (See paragraphs 4.12.1 to 4.12.9, especially paragraphs 4.12.8 and 4.12.9).	Yes. A single provision makes the definition and requirements clearer for business and government lower costs and complexity while increasing usability. Naturally this encourages adoption.	<ul style="list-style-type: none"> • Go to g 51-72 of the consultation paper- (Part IV –Electronic Gov) • (See paragraphs 4.12.1 to 4.12.9).- pg 66-69 of consultation paper - (Part IV – Electronic Gov)
Q14	Proposed sections 9 and 9A of the ETA ³⁵⁹ require compliance with any additional technical requirements as to form and procedure that Government agencies may have in relation to the acceptance of electronic originals. Should there be express requirements to comply with such additional technical requirements in the case where the intended recipient of electronic originals is not a Government agency? Would your views differ if, instead of a single provision on electronic originals, there are specific provisions on the use of electronic communications in different situations? (See paragraphs 4.12.13 to 4.12.16)	No. A consistent framework for secure electronic record management will make it easier and less confusing for International and local entities to implement. Yes.	<ul style="list-style-type: none"> • Go to g 51-72 of the consultation paper- (Part IV –Electronic Gov) • (See paragraphs 4.12.13 to 4.12.16) - - pg 66-69 of consultation paper - (Part IV –Electronic Gov)

Question	Comments	Reference
Part V – UNCITRAL Electronic Contracts Convention and related Issues (pg 73-90 of the Consultation paper)		
Q15	Do you agree that the definition of an electronic signature should not require such a signature to fulfill both an identification as well as an approval function?	Yes. nCipher agrees with Singapore's response to UN. http://daccessdds.un.org/doc/UNDOC/GEN/V05/844/58/PDF/V0584458.pdf?OpenElement
Q16	Do you agree that a general provision providing for the functional equivalence of electronic signatures to handwritten signatures (e.g. section 8) should not contain any reliability requirement?	<ul style="list-style-type: none"> • Go to pg 81 - 85 Consultation paper - (Part V – UNCITRAL Electronic Contracts Convention and related Issues) • For more info on the ETA 1998 of Section 8, please visit : http://agcvldb4.agc.gov.sg/

Q18	What difficulties or benefits do you foresee if the provisions of article 9(4) and (5) of the draft convention (relating to originals) are adopted in the ETA?	Presenting an original electronic record is difficult to define without providing a greater definition of the technology of process. As with the US comments, http://daccessdds.un.org/doc/UNDOC/GEN/V05/850/78/PDF/V0585078.pdf?OpenElement , I believe “made available” would be a better phrasing with clear comments as to what “made available” entails.	<ul style="list-style-type: none"> • Go to pg 85 - 90 Consultation paper - (Part V – UNCITRAL Electronic Contracts Convention and related Issues) • For more info on the ETA 1998 of Section 9 please visit : http://agcvldb4.agc.gov.sg/
Q19	Do you have any comments on proposed section 9A in Annex B? Do you agree with the criteria for acceptance of electronic originals in proposed section 9A(1) and (2) in Annex B?	<p>nCipher agrees with the criteria of acceptance for electronic originals in the proposed section. It allows for government or business to store electronic records and produce the record in its original form.</p> <p>The easiest way to ensure the form is preserved would be the use of digital signatures. Validation of the provided electronic record against the time-stamped digital signature created at time of storage/process would ensure the integrity to the original electronic record.</p>	<ul style="list-style-type: none"> • Go to pg 85 - 90 Consultation paper - (Part V – UNCITRAL Electronic Contracts Convention and related Issues) • Go to Annex B –pg 121 to 125 of the Consultation paper - (Proposed Legislation amendments relating to Electronic Government)

Digital Versus Electronic Signatures

Digital Signatures

The most full-featured and, arguably, the most secure type of e-signature is the digital signature, which relies on public key cryptography. The PKI, based on encryption technology, was created to secure transactions on the Internet. The foundation of PKI, public key cryptography, is an encryption method that uses a two-part key that consists of a public and private component.

Data are encrypted with the public key and is then decrypted by the recipient with his or her own private key. This technology is quickly becoming the best way to ensure safe business-to-business communication using public key signing models to seal data, protect conversations, and secure the electronic audit trail..

Digital certificates, (electronic credentials) are a legally binding electronic confirmation for business transactions. The Certificate contains data (meta-data) about the quality and the ownership of the key pair used in the encryption (signature) process. It provides strong proof (depending on the quality and cost of the certificate issuance process) that the document sealed by the signature process has not been changed since it was signed. Non-repudiation is a key element of long term document archival; simply put, it supports the ability to use electronic documents to prove who did what with whom and when.

Perhaps the single greatest drawback to traditional PKI technologies is the complexity and expense of the infrastructure deployment required for its use as individual signing technology. Business needs a means of issuing certificates to individuals, aging them, revoking them, and tracking all of this activity. This has impeded the uptake of the technology in spite of almost universal technical agreement on its elegance and fitness for the tasks at hand. In addition, batch signing technologies are arising that greatly simplify the use of digital signatures based on public key technology in batch processes where return on investment is more obvious than individual authentication and signing applications..

Electronic Signatures

Signatures in the broad electronic signature category do not rely on cryptographic methods. Instead, these are often image based or biometrics-based solutions in which the characteristics of a fingerprint, the iris of the eye, or the pressure of pen on a surface is encoded digitally and then stored and transmitted by computer to verify identity. While such signatures may be “digitized”—translated from physical characteristics or actions to computer-readable code—they are not “digital.” Often, these methods are combined with other technologies:

Imaging technology permits an image such as a pen-and-ink signature or document to be scanned and the results stored. The resulting signatures are verified when the stored digitized image is compared for verification. It is often combined with other technologies. Applications of image based electronic signatures are common in American retail environments; everyone has “signed” with an electronic pen, thus encumbering a credit card transaction at one time or another.

While these technologies are useful in authenticating individuals, they do nothing to assure long-term viability of electronic data. There is no encryption, no “sealing” of the transaction data. The image of the signature gathered at the check out counter can be associated with any data set

without setting off any electronic “alarms”. The ability to scan a retina and authenticate an individual does nothing to prevent use of that individual’s identity data for fraudulent purposes. Unlike public key based digital signatures, there is nothing in this technology that is of use to the records management problem set.

Not all signatures are equal. While all imply assent and assume identity some carry more weight, and require more ceremony, than others. Determining the importance, the “weight” of a signature is a business decision. At times, individuals must show identification and sign in the presence of a witness. Other business processes permit signature by proxy. For example, closing a mortgage in the United States requires the presence and clear identification of the parties with signing performed in the presence of often highly paid witnesses. Attempts to automate this process have been largely unsuccessful in spite of the potentially huge benefit in time and cost. This is largely due to the complexity of the process, the degree to which the process is grounded in culture and ceremony, the “weight” of the process.

Producing a run of accounts payable checks, on the other hand, is often completely automated. The amounts are calculated by computer on behalf of the organization. The checks are produced on preprinted stock and the signatures applied mechanically. Fraud and error are controlled by processes – audit, both pre and post check run, is common, as is separation of the signature plates and the check stock. The end result is a commitment by the organization to pay the face amount of the check to the payee. Assent to action and content has been achieved, on behalf of the business entity, in the name of the business entity, by mechanical proxy through a fully automated process.

Business processes that convey assent or that protect and acknowledge content are ripe for automation. One automates a process to take time, risk and cost out of the business. Document archival, for example, is ripe for the next step in automation. Companies spend millions on document retention, from development of document retention policies to real estate devoted to climate-controlled storage of long-term archives. Often, documents consigned to the archives are fixed with a mark, a time stamp or organizational seal that marks them as “official” archive components. The mark is used to distinguish the official, original file copy from a non-binding version. The mark, often applied by a mechanical clock or time stamper, is, in effect, an organizational signature.

There is no analogue to this process in cyberspace. Electronic records exist as “disk dust”, arrays of bits on media. There is no currently accepted means of marking one file as part of the archive, thus excluding all other copies of the file as acceptable evidence. Versioning and revision systems (document management engines) use metadata to isolate and describe data files, but the data files themselves are indistinguishable from their numerous copies. They all, as well, are subject to undetectable change.

This document suggests an application of public key cryptography as a solution to this dilemma. It presents a network appliance, a “black box” that shelters a digital certificate(s) on behalf of an organization or an individual within the organization. It applies the most stringent security procedures to the acquisition and management of the identity proxied in the certificate. It then accepts data from the enterprise, from, for example, a payroll application or an automated trading system, and signs and timestamps the audit trail data on behalf of the organizational identity bound up in the certificate.

The effect of the process is a set of data that have been legally signed as defined in the Signature law. They have been “sealed”, signed by proxy, just as is done in a check run. The output of the process can be stored as audit data with high evidentiary value or used to authorize other enterprise processes. The potential benefits include final elimination of the paper

audit trail, replacing it with an electronic audit trail that carries the evidentiary weight of its paper predecessor.

The Electronic Audit Trail

Managing digital “paper trails” presents some major challenges for the evolving electronic corporation. Use of electronic records in lieu of paper records is exploding. The tools and methods of records management are woefully behind the daily reality of the increasingly digital world of corporate archives. Even the electronic records management policies of the National Archives and Records Administration, the US national record keeper, “have not yet evolved to reflect the modern record keeping environment”.

Real business, with real business consequences, is conducted every day with technology that seems casual. Email, instant messaging, and “personal” computer applications have enabled us to leap large buildings in a single bound in terms of productivity improvements. It is, however, the content of our communication that constitutes the audit trail, not the form by which we create and communicate business information. For example, the National Association of Securities Dealers (NASD) “requires that correspondence with public customers, both written and electronic, be maintained in compliance with NASD rules and the SEC Rules 17a-3 and 17a-4. This means that a RR’s (registered representative) email correspondence with the public relating to the firm’s business, generated both at the office and at home, is subject to these provisions”.

If the audit trail is the sum of the data points that enables us to prove how we got here from there, then electronic content has assumed a significant proportion of audit trail’s constituent parts. In addition, electronic communications technologies may inadvertently memorialize and create an audit trail for conversations that would have been temporal without electronic capture and transmission models. An email, for example, that communicates information better left to a phone call or discussed over lunch may be convenient, but it creates audit trail data that can be used to substantiate claims where the lunch conversation would have been fleeting and inadmissible.

The point is obvious – electronic records have become an integral component of the audit trail, the corporate equivalent of the “permanent record”. Careful management and preservation of these records, in accordance with a company records retention policy, helps to assure that e records do not “come back to bite” at some time in the future. Assuring that those records targeted for long term storage are made “tamper evident” by use of appropriate technology is as essential to their long term storage and validity as use of acid free paper and silver halide film were to physical records management. The application of digital signature technology to long-term electronic storage is a logical extension of current records management best practices into the realm of electronic document management.

nCipher and National Time Authority in APAC provides Integrity protection of Electronics Records with Trusted Time Source

nCipher Digital Signing and Time Stamping solutions gets Trusted Time Source from National Time Authority.

nCipher plc (LSE:NCH), a leading provider of IT cryptographic security and SPRING have entered into a collaboration with National Time Authority in APAC to provide a trusted time source into nCipher TimeSource Master Clock (TMC200) and Document Sealing Engine (DSE200) for integrity and timeliness protection of electronic records.

One critical element in ensuring the integrity and security of electronic transactions and records is the trusted time source providing the time reference for digital signing and time-stamping.

In Singapore, nCipher and SPRING Singapore is in the process of providing a trusted and secured time source for nCipher TimeSource Master Clock (TSMC200) and Digital Sealing Engine (DSE200), which will in turn enable enterprises to protect the integrity and security of electronic transactions and records.

Time source provided by SPRING comes from the national time standard for Singapore, which is the authoritative reference for governmental and commercial services that need accurate, precise and reliable time. As the national measurement authority, SPRING maintains the time standard using caesium atomic clocks. SPRING is also a member of the global network of time authorities that maintain the international time scale known as the Coordinated Universal Time, from which Singapore's time standard is derived.

In Japan, Amano, the leading Time Authority in Japan, has developed a system that uses nCipher's TMC200 together with nCipher's Document Sealing Engine (DSE200) to provide a new Time Stamp Authority Service. The service will enable Japanese businesses to comply with Japan's imminent e-document legislation. Most transactions in Japan are in paper format and the e-document law will enable organizations to scan paper documents and store these in electronic form. This e-document law is scheduled to come into force on 1 April 2005 and is expected to save organizations millions of Yen in storage costs.

In order to ensure each document maintains integrity and identity during the storage period and that the date of filing can not be tampered with, the documents are required to have a secure time-stamp. Amano has developed a system that uses nCipher's TMC200 together with nCipher's Document Sealing Engine (DSE200) to provide customers with a calibrated, undisputable time source with which to time-stamp their documents. Secom TrustNet is delivering a Time Stamp Authority Service using this system and is offering a third-party certification service to Japanese businesses.

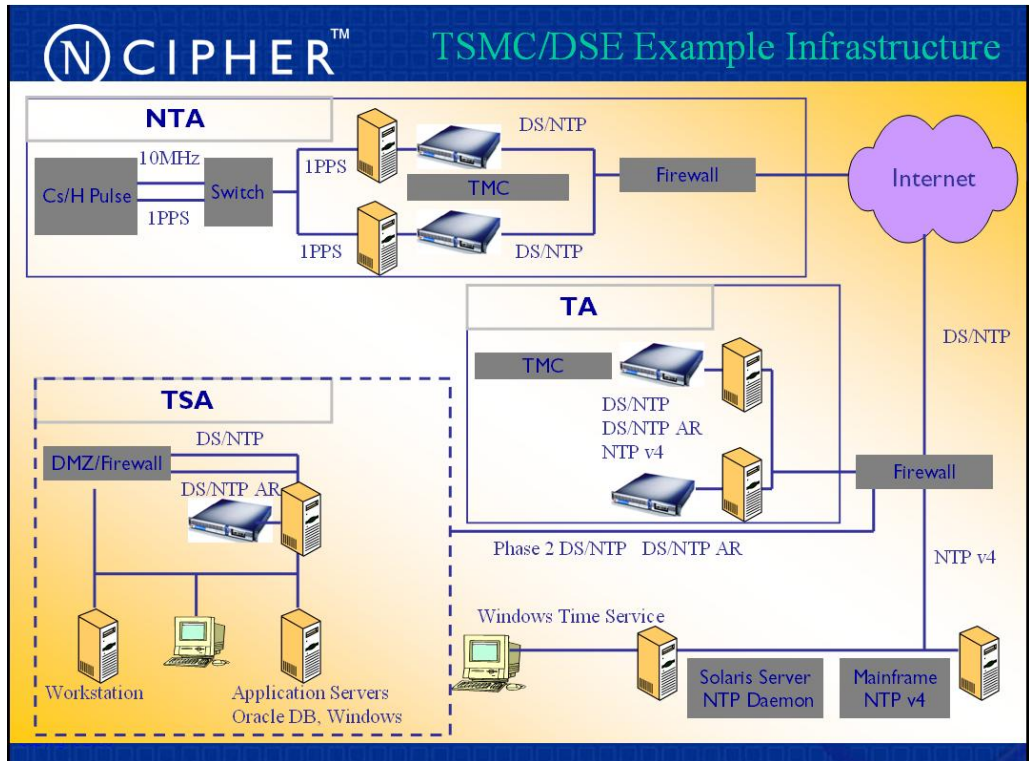


Fig 1. TSMC/DSE Infrastructure with National Time Authority

Summary

In today's business world almost every document or transaction is digital in its origination. Documents that are "paper" are usually created first on a computer, not on a typewriter. Orders are put into the "system," which is more often than not an IT system. Voice mails have gone from analog to digital. Security cameras are increasingly digital, as is the bulk of evidence submitted in both civil and criminal cases.

In order for today's business world to operate in an efficient manner, there needs to be a degree of trust between parties. Millions of stock transactions take place daily. Businesses develop relationships with each other and make assumptions based on those relationships. Integrity plays a crucial role in day-to-day business dealings. When trust or integrity is questioned, businesses can be brought to a crashing halt. Billions of dollars can be lost in minutes.

In the business world trust is measured by reputation, word, by signed and dated documents, by witnesses. It is customary that parties to a transaction rely on each other to provide proof, if asked, that transactions indeed took place and when. In fact no business could operate for very long if it could not provide such assurances. Yet, that is exactly how business currently operates in the digital realm of the Internet. The probability is high that no one can demonstrate with certainty when an online transaction took place, nor can the content of the digital audit trail be guaranteed accurate. This "clouds" the integrity and the possible validity of digital records.

Public key cryptography is the core technology behind the strongest, most reliable form of digital signatures on the market. The expense, difficulty of implementation, and lack of interoperability between vendor products have impeded adoption of the technology. The focus on use of the PKI for individual authentication has exacerbated the implementation dilemma; simple password and PIN systems are often "good enough" for individual authentication and are certainly easier and cheaper to deploy than is the PKI.

A new kind of device, known as a "Document Sealing Engine", has arisen to fill the void. This device takes the form of a network appliance, a "black box", which securely manages the signing process on behalf of departments and businesses.

It operates much like the old-fashioned check signing machines ubiquitous in accounts payable and payroll applications. The digital certificate takes the place of the signature plates, and the signing engine the place of the check signing machinery. It can be deployed deep within enterprise applications, tasked to operate on behalf of the company, producing the strong evidence necessary to realize the next wave of automation efficiencies, eliminating the last vestiges of the "paper trail".

Appendix

- **RFC 3161**

Network Working Group
Request for Comments: 3161
Category: Standards Track

C. Adams
Entrust
P. Cain
BBN
D. Pinkas
Integris
R. Zuccherato
Entrust
August 2001

Internet X.509 Public Key Infrastructure
Time-Stamp Protocol (TSP)

Status of this Memo

This document specifies an Internet standards track protocol for the Internet community, and requests discussion and suggestions for improvements. Please refer to the current edition of the "Internet Official Protocol Standards" (STD 1) for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Copyright (C) The Internet Society (2001). All Rights Reserved.

Abstract

This document describes the format of a request sent to a Time Stamping Authority (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.

1. Introduction

A time-stamping service supports assertions of proof that a datum existed before a particular time. A TSA may be operated as a Trusted Third Party (TTP) service, though other operational models may be appropriate, e.g., an organization might require a TSA for internal time-stamping purposes.

Non-repudiation services [ISONR] require the ability to establish the existence of data before specified times. This protocol may be used as a building block to support such services. An example of how to prove that a digital signature was generated during the validity period of a public key certificate is given in an annex.

The key words "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", "SHALL", "RECOMMENDED", "MAY", and "OPTIONAL" in this document (in uppercase, as shown) are to be interpreted as described in [RFC2119].

In order to associate a datum with a particular point in time, a Time Stamp Authority (TSA) may need to be used. This Trusted Third Party provides a "proof-of-existence" for this particular datum at an instant in time.

The TSA's role is to time-stamp a datum to establish evidence indicating that a datum existed before a particular time. This can then be used, for example, to verify that a digital signature was applied to a message before the corresponding certificate was revoked thus allowing a revoked public key certificate to be used for verifying signatures created prior to the time of revocation. This is an important public key infrastructure operation. The TSA can also be used to indicate the time of submission when a deadline is critical, or to indicate the time of transaction for entries in a log. An exhaustive list of possible uses of a TSA is beyond the scope of this document.

This standard does not establish overall security requirements for TSA operation, just like other PKIX standards do not establish such requirements for CA operation. Rather, it is anticipated that a TSA will make known to prospective clients the policies it implements to ensure accurate time-stamp generation, and clients will make use of the services of a TSA only if they are satisfied that these policies meet their needs.

2. The TSA

The TSA is a TTP that creates time-stamp tokens in order to indicate that a datum existed at a particular point in time.

For the remainder of this document a "valid request" shall mean one that can be decoded correctly, is of the form specified in Section 2.4, and is from a supported TSA subscriber.

2.1. Requirements of the TSA

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 include within each time-stamp token an identifier to uniquely indicate the security policy under which the token was created.
6. to only time-stamp a hash representation of the datum, i.e., a data imprint associated with a one-way collision resistant hash-function uniquely identified by an OID.
7. to examine the OID of the one-way collision resistant hash-function and to verify that the hash value length is consistent with the hash algorithm.
8. not to examine the imprint being time-stamped in any way (other than to check its length, as specified in the previous bullet).
9. not to include any identification of the requesting entity in the time-stamp tokens.
10. to sign each time-stamp token using a key generated exclusively for this purpose and have this property of the key indicated on the corresponding certificate.
11. to include additional information in the time-stamp token, if asked by the requester using the extensions field, only for the extensions that are supported by the TSA. If this is not possible, the TSA SHALL respond with an error message.

2.2. TSA Transactions

As the first message of this mechanism, the requesting entity requests a time-stamp token by sending a request (which is or includes a `TimeStampReq`, as defined below) to the Time Stamping Authority. As the second message, the Time Stamping Authority responds by sending a response (which is or includes a `TimeStampResp`, as defined below) to the requesting entity.

Upon receiving the response (which is or includes a `TimeStampResp` that normally contains a `TimeStampToken` (TST), as defined below), the requesting entity SHALL verify the status error returned in the response and if no error is present it SHALL verify the various fields contained in the `TimeStampToken` and the validity of the digital signature of the `TimeStampToken`. In particular, it SHALL verify that what was time-stamped corresponds to what was requested to be time-stamped. The requester SHALL verify that the `TimeStampToken` contains the correct certificate identifier of the

TSA, the correct data imprint and the correct hash algorithm OID. It SHALL then verify the timeliness of the response by verifying either the time included in the response against a local trusted time reference, if one is available, or the value of the nonce (large random number with a high probability that it is generated by the client only once) included in the response against the value included in the request. For more details about replay attack detection, see the security considerations section (item 6). If any of the verifications above fails, the TimeStampToken SHALL be rejected.

Then, since the TSA's certificate may have been revoked, the status of the certificate SHOULD be checked (e.g., by checking the appropriate CRL) to verify that the certificate is still valid.

Then, the client application SHOULD check the policy field to determine whether or not the policy under which the token was issued is acceptable for the application.

2.3. Identification of the TSA

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. The corresponding certificate MUST contain only one instance of the extended key usage field extension as defined in [RFC2459] Section 4.2.1.13 with KeyPurposeID having value:

id-kp-timeStamping. This extension MUST be critical.

The following object identifier identifies the KeyPurposeID having value id-kp-timeStamping.

```
id-kp-timeStamping OBJECT IDENTIFIER ::= {iso(1)
    identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7)
    kp (3) timestamping (8)}
```

2.4. Request and Response Formats

2.4.1. Request Format

A time-stamping request is as follows:

```
TimeStampReq ::= SEQUENCE {
    version                INTEGER { v1(1) },
    messageImprint         MessageImprint,
    --a hash algorithm OID and the hash value of the data to be
```

```

--time-stamped
reqPolicy      TSAPolicyId      OPTIONAL,
nonce          INTEGER          OPTIONAL,
certReq        BOOLEAN          DEFAULT FALSE,
extensions     [0] IMPLICIT Extensions OPTIONAL }

```

The version field (currently v1) describes the version of the Time-Stamp request.

The messageImprint field SHOULD contain the hash of the datum to be time-stamped. The hash is represented as an OCTET STRING. Its length MUST match the length of the hash value for that algorithm (e.g., 20 bytes for SHA-1 or 16 bytes for MD5).

```

MessageImprint ::= SEQUENCE {
    hashAlgorithm      AlgorithmIdentifier,
    hashedMessage      OCTET STRING }

```

The hash algorithm indicated in the hashAlgorithm field SHOULD be a known hash algorithm (one-way and collision resistant). That means that it SHOULD be one-way and collision resistant. The Time Stamp Authority SHOULD check whether or not the given hash algorithm is known to be "sufficient" (based on the current state of knowledge in cryptanalysis and the current state of the art in computational resources, for example). If the TSA does not recognize the hash algorithm or knows that the hash algorithm is weak (a decision left to the discretion of each individual TSA), then the TSA SHOULD refuse to provide the time-stamp token by returning a pkiStatusInfo of 'bad_alg'.

The reqPolicy field, if included, indicates the TSA policy under which the TimeStampToken SHOULD be provided. TSAPolicyId is defined as follows:

```

TSAPolicyId ::= OBJECT IDENTIFIER

```

The nonce, if included, allows the client to verify the timeliness of the response when no local clock is available. The nonce is a large random number with a high probability that the client generates it only once (e.g., a 64 bit integer). In such a case the same nonce value MUST be included in the response, otherwise the response shall be rejected.

If the certReq field is present and set to true, the TSA's public key certificate that is referenced by the ESSCertID identifier inside a SigningCertificate attribute in the response MUST be provided by the TSA in the certificates field from the SignedData structure in that response. That field may also contain other certificates.

If the certReq field is missing or if the certReq field is present and set to false then the certificates field from the SignedData structure MUST not be present in the response.

The extensions field is a generic way to add additional information to the request in the future. Extensions is defined in [RFC 2459]. If an extension, whether it is marked critical or not critical, is used by a requester but is not recognized by a time-stamping server, the server SHALL not issue a token and SHALL return a failure (unacceptedExtension).

The time-stamp request does not identify the requester, as this information is not validated by the TSA (See Section 2.1). In situations where the TSA requires the identity of the requesting entity, alternate identification /authentication means have to be used (e.g., CMS encapsulation [CMS] or TLS authentication [RFC2246]).

2.4.2. Response Format

A time-stamping response is as follows:

```
TimeStampResp ::= SEQUENCE {
    status                PKIStatusInfo,
    timeStampToken       TimeStampToken    OPTIONAL }
```

The status is based on the definition of status in section 3.2.3 of [RFC2510] as follows:

```
PKIStatusInfo ::= SEQUENCE {
    status            PKIStatus,
    statusString     PKIFreeText    OPTIONAL,
    failInfo         PKIFailureInfo OPTIONAL }
```

When the status contains the value zero or one, a TimeStampToken MUST be present. When status contains a value other than zero or one, a TimeStampToken MUST NOT be present. One of the following values MUST be contained in status:

```
PKIStatus ::= INTEGER {
    granted                (0),
    -- when the PKIStatus contains the value zero a TimeStampToken, as
    -- requested, is present.
    grantedWithMods       (1),
    -- when the PKIStatus contains the value one a TimeStampToken,
    -- with modifications, is present.
    rejection              (2),
    waiting                 (3),
    revocationWarning      (4),
```

```
-- this message contains a warning that a revocation is
-- imminent
revocationNotification (5)
-- notification that a revocation has occurred }
```

Compliant servers SHOULD NOT produce any other values. Compliant clients MUST generate an error if values it does not understand are present.

When the TimeStampToken is not present, the failInfo indicates the reason why the time-stamp request was rejected and may be one of the following values.

```
PKIFailureInfo ::= BIT STRING {
  badAlg          (0),
    -- unrecognized or unsupported Algorithm Identifier
  badRequest      (2),
    -- transaction not permitted or supported
  badDataFormat   (5),
    -- the data submitted has the wrong format
  timeNotAvailable (14),
    -- the TSA's time source is not available
  unacceptedPolicy (15),
    -- the requested TSA policy is not supported by the TSA
  unacceptedExtension (16),
    -- the requested extension is not supported by the TSA
  addInfoNotAvailable (17)
    -- the additional information requested could not be understood
    -- or is not available
  systemFailure   (25)
    -- the request cannot be handled due to system failure }
```

These are the only values of PKIFailureInfo that SHALL be supported.

Compliant servers SHOULD NOT produce any other values. Compliant clients MUST generate an error if values it does not understand are present.

The statusString field of PKIStatusInfo MAY be used to include reason text such as "messageImprint field is not correctly formatted".

A TimeStampToken is as follows. It is defined as a ContentInfo ([CMS]) and SHALL encapsulate a signed data content type.

```
TimeStampToken ::= ContentInfo
  -- contentType is id-signedData ([CMS])
  -- content is SignedData ([CMS])
```


The fields of type EncapsulatedContentInfo of the SignedData construct have the following meanings:

eContentType is an object identifier that uniquely specifies the content type. For a time-stamp token it is defined as:

```
id-ct-TSTInfo OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) ct(1) 4 }
```

eContent is the content itself, carried as an octet string. The eContent SHALL be the DER-encoded value of TSTInfo.

The time-stamp token MUST NOT contain any signatures other than the signature of the TSA. The certificate identifier (ESSCertID) of the TSA certificate MUST be included as a signerInfo attribute inside a SigningCertificate attribute.

```
TSTInfo ::= SEQUENCE {
  version                INTEGER { v1(1) },
  policy                 TSAPolicyId,
  messageImprint         MessageImprint,
  -- MUST have the same value as the similar field in
  -- TimeStampReq
  serialNumber           INTEGER,
  -- Time-Stamping users MUST be ready to accommodate integers
  -- up to 160 bits.
  genTime                GeneralizedTime,
  accuracy               Accuracy                OPTIONAL,
  ordering               BOOLEAN                DEFAULT FALSE,
  nonce                 INTEGER                OPTIONAL,
  -- MUST be present if the similar field was present
  -- in TimeStampReq. In that case it MUST have the same value.
  tsa                   [0] GeneralName        OPTIONAL,
  extensions             [1] IMPLICIT Extensions OPTIONAL }
```

The version field (currently v1) describes the version of the time-stamp token.

Conforming time-stamping servers MUST be able to provide version 1 time-stamp tokens.

Among the optional fields, only the nonce field MUST be supported.

Conforming time-stamping requesters MUST be able to recognize version 1 time-stamp tokens with all the optional fields present, but are not mandated to understand the semantics of any extension, if present.

The policy field MUST indicate the TSA's policy under which the response was produced. If a similar field was present in the TimeStampReq, then it MUST have the same value, otherwise an error (unacceptedPolicy) MUST be returned. This policy MAY include the following types of information (although this list is certainly not exhaustive):

- * The conditions under which the time-stamp token may be used.
- * The availability of a time-stamp token log, to allow later verification that a time-stamp token is authentic.

The messageImprint MUST have the same value as the similar field in TimeStampReq, provided that the size of the hash value matches the expected size of the hash algorithm identified in hashAlgorithm.

The serialNumber field is an integer assigned by the TSA to each TimeStampToken. It MUST be unique for each TimeStampToken issued by a given TSA (i.e., the TSA name and serial number identify a unique TimeStampToken). It should be noticed that the property MUST be preserved even after a possible interruption (e.g., crash) of the service.

genTime is the time at which the time-stamp token has been created by the TSA. It is expressed as UTC time (Coordinated Universal Time) to reduce confusion with the local time zone use. UTC is a time scale, based on the second (SI), as defined and recommended by the CCIR, and maintained by the Bureau International des Poids et Mesures (BIPM). A synonym is "Zulu" time which is used by the civil aviation and represented by the letter "Z" (phonetically "Zulu").

The ASN.1 GeneralizedTime syntax can include fraction-of-second details. Such syntax, without the restrictions from [RFC 2459] Section 4.1.2.5.2, where GeneralizedTime is limited to represent the time with a granularity of one second, may be used here.

GeneralizedTime values MUST include seconds. However, when there is no need to have a precision better than the second, then GeneralizedTime with a precision limited to one second SHOULD be used (as in [RFC 2459]).

The syntax is: YYYYMMDDhhmmss[.s...]Z
Example: 19990609001326.34352Z

X.690 | ISO/IEC 8825-1 provides the following restrictions for a DER-encoding.

The encoding MUST terminate with a "Z" (which means "Zulu" time). The decimal point element, if present, MUST be the point option ".". The fractional-seconds elements, if present, MUST omit all trailing 0's; if the elements correspond to 0, they MUST be wholly omitted, and the decimal point element also MUST be omitted.

Midnight (GMT) shall be represented in the form: "YYYYMMDD000000Z" where "YYYYMMDD" represents the day following the midnight in question.

Here are a few examples of valid representations:

```
"19920521000000Z"
"19920622123421Z"
"19920722132100.3Z"
```

accuracy represents the time deviation around the UTC time contained in GeneralizedTime.

```
Accuracy ::= SEQUENCE {
    seconds      INTEGER          OPTIONAL,
    millis       [0] INTEGER (1..999)  OPTIONAL,
    micros       [1] INTEGER (1..999)  OPTIONAL }
```

If either seconds, millis or micros is missing, then a value of zero MUST be taken for the missing field.

By adding the accuracy value to the GeneralizedTime, an upper limit of the time at which the time-stamp token has been created by the TSA can be obtained. In the same way, by subtracting the accuracy to the GeneralizedTime, a lower limit of the time at which the time-stamp token has been created by the TSA can be obtained.

accuracy can be decomposed in seconds, milliseconds (between 1-999) and microseconds (1-999), all expressed as integer.

When the accuracy optional field is not present, then the accuracy may be available through other means, e.g., the TSAPolicyId.

If the ordering field is missing, or if the ordering field is present and set to false, then the genTime field only indicates the time at which the time-stamp token has been created by the TSA. In such a case, the ordering of time-stamp tokens issued by the same TSA or different TSAs is only possible when the difference between the genTime of the first time-stamp token and the genTime of the second time-stamp token is greater than the sum of the accuracies of the genTime for each time-stamp token.

If the ordering field is present and set to true, every time-stamp token from the same TSA can always be ordered based on the genTime field, regardless of the genTime accuracy.

The nonce field MUST be present if it was present in the TimeStampReq. In such a case it MUST equal the value provided in the TimeStampReq structure.

The purpose of the tsa field is to give a hint in identifying the name of the TSA. If present, it MUST correspond to one of the subject names included in the certificate that is to be used to verify the token. However, the actual identification of the entity that signed the response will always occur through the use of the certificate identifier (ESSCertID Attribute) inside a SigningCertificate attribute which is part of the signerInfo (See Section 5 of [ESS]).

extensions is a generic way to add additional information in the future. Extensions is defined in [RFC 2459].

Particular extension field types may be specified in standards or may be defined and registered by any organization or community.

3. Transports

There is no mandatory transport mechanism for TSA messages in this document. The mechanisms described below are optional; additional optional mechanisms may be defined in the future.

3.1. Time-Stamp Protocol Using E-mail

This section specifies a means for conveying ASN.1-encoded messages for the protocol exchanges described in Section 2 and Appendix D via Internet mail.

Two MIME objects are specified as follows:

```
Content-Type: application/timestamp-query
Content-Transfer-Encoding: base64
<<the ASN.1 DER-encoded Time-Stamp message, base64-encoded>>
```

```
Content-Type: application/timestamp-reply
Content-Transfer-Encoding: base64
<<the ASN.1 DER-encoded Time-Stamp message, base64-encoded>>
```

These MIME objects can be respectively sent and received using common MIME processing engines and provides a simple Internet mail transport for Time-Stamp messages.

For the application/timestamp-query and application/timestamp-reply MIME types, implementations SHOULD include the optional "name" and "filename" parameters. Including a file name helps preserve type information when time-stamp queries and replies are saved as files. When these parameters are included, a file name with the appropriate extension SHOULD be selected:

MIME Type	File Extension
application/timestamp-query	.TSQ
application/timestamp-reply	.TSR

In addition, the file name SHOULD be limited to eight characters followed by a three letter extension. The eight character filename base can be any distinct name.

3.2. File Based Protocol

A file containing a time-stamp message MUST contain only the DER encoding of one TSA message, i.e., there MUST be no extraneous header or trailer information in the file. Such files can be used to transport time stamp messages using for example, FTP.

A Time-Stamp Request SHOULD be contained in a file with file extension .tsq (like Time-Stamp Query). A Time-Stamp Response SHOULD be contained in a file with file extension .tsr (like Time-Stamp Reply).

3.3. Socket Based Protocol

The following simple TCP-based protocol is to be used for transport of TSA messages. This protocol is suitable for cases where an entity initiates a transaction and can poll to pick up the results.

The protocol basically assumes a listener process on a TSA that can accept TSA messages on a well-defined port (IP port number 318).

Typically an initiator binds to this port and submits the initial TSA message. The responder replies with a TSA message and/or with a reference number to be used later when polling for the actual TSA message response.

If a number of TSA response messages are to be produced for a given request (say if a receipt must be sent before the actual token can be produced) then a new polling reference is also returned.

When the final TSA response message has been picked up by the initiator then no new polling reference is supplied.

The initiator of a transaction sends a "direct TCP-based TSA message" to the recipient. The recipient responds with a similar message.

A "direct TCP-based TSA message" consists of:
 length (32-bits), flag (8-bits), value (defined below)

The length field contains the number of octets of the remainder of the message (i.e., number of octets of "value" plus one). All 32-bit values in this protocol are specified to be in network byte order.

Message name	flag	value
tsaMsg	'00'H	DER-encoded TSA message
	--	TSA message
pollRep	'01'H	polling reference (32 bits), time-to-check-back (32 bits)
	--	poll response where no TSA message response ready; use polling
	--	reference value (and estimated time value) for later polling
pollReq	'02'H	polling reference (32 bits)
	--	request for a TSA message response to initial message
negPollRep	'03'H	'00'H
	--	no further polling responses (i.e., transaction complete)
partialMsgRep	'04'H	next polling reference (32 bits), time-to-check-back (32 bits), DER-encoded TSA message
	--	partial response (receipt) to initial message plus new polling
	--	reference (and estimated time value) to use to get next part of
	--	response
finalMsgRep	'05'H	DER-encoded TSA message
	--	final (and possibly sole) response to initial message
errorMsgRep	'06'H	human readable error message
	--	produced when an error is detected (e.g., a polling reference
	--	is received which doesn't exist or is finished with)

The sequence of messages that can occur is:

- a) entity sends tsaMsg and receives one of pollRep, negPollRep, partialMsgRep, or finalMsgRep in response.
- b) end entity sends pollReq message and receives one of negPollRep, partialMsgRep, finalMsgRep, or errorMsgRep in response.

The "time-to-check-back" parameter is an unsigned 32-bit integer. It is the time in seconds indicating the minimum interval after which the client SHOULD check the status again.

It provides an estimate of the time that the end entity should send its next pollReq.

3.4. Time-Stamp Protocol via HTTP

This subsection specifies a means for conveying ASN.1-encoded messages for the protocol exchanges described in Section 2 and Appendix D via the HyperText Transfer Protocol.

Two MIME objects are specified as follows.

Content-Type: application/timestamp-query

<<the ASN.1 DER-encoded Time-Stamp Request message>>

Content-Type: application/timestamp-reply

<<the ASN.1 DER-encoded Time-Stamp Response message>>

These MIME objects can be sent and received using common HTTP processing engines over WWW links and provides a simple browser-server transport for Time-Stamp messages.

Upon receiving a valid request, the server MUST respond with either a valid response with content type application/timestamp-response or with an HTTP error.

4. Security Considerations

This entire document concerns security considerations. When designing a TSA service, the following considerations have been identified that have an impact upon the validity or "trust" in the time-stamp token.

1. When a TSA shall not be used anymore, but the TSA private key has not been compromised, the authority's certificate SHALL be revoked. When the reasonCode extension relative to the revoked certificate from the TSA is present in the CRL entry extensions, it SHALL be set either to unspecified (0), affiliationChanged (3), superseded (4) or cessationOfOperation (5). In that case, at any future time, the tokens signed with the corresponding key will be considered as invalid, but tokens generated before the revocation time will remain valid. When the reasonCode extension relative to the revoked certificate from the TSA is not present in the CRL entry extensions, then all the tokens that have been signed with the corresponding key SHALL be considered as invalid. For that reason, it is recommended to use the reasonCode extension.

2. When the TSA private key has been compromised, then the corresponding certificate SHALL be revoked. In that case, the reasonCode extension relative to the revoked certificate from the TSA may or may not be present in the CRL entry extensions. When it is present then it SHALL be set to keyCompromise (1). Any token signed by the TSA using that private key cannot be trusted anymore. For this reason, it is imperative that the TSA's private key be guarded with proper security and controls in order to minimize the possibility of compromise. In case the private key does become compromised, an audit trail of all tokens generated by the TSA MAY provide a means to discriminate between genuine and false backdated tokens. Two time-stamp tokens from two different TSAs is another way to address this issue.
3. The TSA signing key MUST be of a sufficient length to allow for a sufficiently long lifetime. Even if this is done, the key will have a finite lifetime. Thus, any token signed by the TSA SHOULD be time-stamped again (if authentic copies of old CRLs are available) or notarized (if they aren't) at a later date to renew the trust that exists in the TSA's signature. time-stamp tokens could also be kept with an Evidence Recording Authority to maintain this trust.
4. A client application using only a nonce and no local clock SHOULD be concerned about the amount of time it is willing to wait for a response. A 'man-in-the-middle' attack can introduce delays. Thus, any TimeStampResp that takes more than an acceptable period of time SHOULD be considered suspect. Since each transport method specified in this document has different delay characteristics, the period of time that is considered acceptable will depend upon the particular transport method used, as well as other environment factors.
5. If different entities obtain time-stamp tokens on the same data object using the same hash algorithm, or a single entity obtains multiple time-stamp tokens on the same object, the generated time-stamp tokens will include identical message imprints; as a result, an observer with access to those time-stamp tokens could infer that the time-stamps may refer to the same underlying data.

6. Inadvertent or deliberate replays for requests incorporating the same hash algorithm and value may happen. Inadvertent replays occur when more than one copy of the same request message gets sent to the TSA because of problems in the intervening network elements. Deliberate replays occur when a middleman is replaying legitimate TS responses. In order to detect these situations, several techniques may be used. Using a nonce always allows to detect replays, and hence its use is RECOMMENDED. Another possibility is to use both a local clock and a moving time window during which the requester remembers all the hashes sent during that time window. When receiving a response, the requester ensures both that the time of the response is within the time window and that there is only one occurrence of the hash value in that time window. If the same hash value is present more than once within a time window, the requester may either use a nonce, or wait until the time window has moved to come back to the case where the same hash value appears only once during that time window.

5. Intellectual Property Rights

The IETF takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on the IETF's procedures with respect to rights in standards-track and standards-related documentation can be found in BCP-11. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementors or users of this specification can be obtained from the IETF Secretariat.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights which may cover technology that may be required to practice this standard. Please address the information to the IETF Executive Director.

The following eight (8) United States Patents related to time stamping, listed in chronological order, are known by the authors to exist at this time. This may not be an exhaustive list. Other patents MAY exist or be issued at any time. This list is provided for informational purposes; to date, the IETF has not been notified of intellectual property rights claimed in regard to any of the

specification contained in this document. Should this situation change, the current status may be found at the online list of claimed rights (IETF Page of Intellectual Property Rights Notices).

Implementers of this protocol SHOULD perform their own patent search and determine whether or not any encumbrances exist on their implementation.

Users of this protocol SHOULD perform their own patent search and determine whether or not any encumbrances exist on the use of this standard.

5,001,752 Public/Key Date-Time Notary Facility

Filing date: October 13, 1989

Issued: March 19, 1991

Inventor: Addison M. Fischer

5,022,080 Electronic Notary

Filing date: April 16, 1989

Issued: June 4, 1991

Inventors: Robert T. Durst, Kevin D. Hunter

5,136,643 Public/Key Date-Time Notary Facility

Filing date: December 20, 1990

Issued: August 4, 1992

Inventor: Addison M. Fischer

Note: This is a continuation of patent # 5,001,752.)

5,136,646 Digital Document Time-Stamping with Catenate Certificate

Filing date: August 2, 1990

Issued: August 4, 1992

Inventors: Stuart A. Haber, Wakefield S. Stornetta Jr.

(assignee) Bell Communications Research, Inc.,

5,136,647 Method for Secure Time-Stamping of Digital Documents

Filing date: August 2, 1990

Issued: August 4, 1992

Inventors: Stuart A. Haber, Wakefield S. Stornetta Jr.

(assignee) Bell Communications Research, Inc.,

5,373,561 Method of Extending the Validity of a Cryptographic Certificate

Filing date: December 21, 1992

Issued: December 13, 1994

Inventors: Stuart A. Haber, Wakefield S. Stornetta Jr.

(assignee) Bell Communications Research, Inc.,

5,422,953 Personal Date/Time Notary Device
Filing date: May 5, 1993
Issued: June 6, 1995
Inventor: Addison M. Fischer

5,781,629 Digital Document Authentication System
Filing date: February 21, 1997
Issued: July 14, 1998
Inventor: Stuart A. Haber, Wakefield S. Stornetta Jr.
(assignee) Surety Technologies, Inc.,

6. References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2246] Dierks, T. and C. Allen, "The TLS Protocol, Version 1.0", RFC 2246, January 1999.
- [RFC2510] Adams, C. and S. Farrell, "Internet X.509 Public Key Infrastructure, Certificate Management Protocols", RFC 2510, March 1999.
- [RFC2459] Housley, R., Ford, W., Polk, W. and D. Solo, "Internet X.509 Public Key Infrastructure, Certificate and CRL Profile", RFC 2459, January 1999.
- [CMS] Housley, R., "Cryptographic Message Syntax", RFC 2630, June 1999.
- [DSS] Digital Signature Standard. FIPS Pub 186. National Institute of Standards and Technology. 19 May 1994.
- [ESS] Hoffman, P., "Enhanced Security Services for S/MIME", RFC 2634, June 1999.
- [ISONR] ISO/IEC 10181-5: Security Frameworks in Open Systems. Non-Repudiation Framework. April 1997.
- [MD5] Rivest, R., "The MD5 Message-Digest Algorithm", RFC 1321, April 1992.
- [SHA1] Secure Hash Standard. FIPS Pub 180-1. National Institute of Standards and Technology. 17 April 1995.

7. Authors' Addresses

Carlisle Adams
Entrust, Inc.
1000 Innovation Drive
Ottawa, Ontario
K2K 3E7
CANADA

E-Mail: cadams@entrust.com

Pat Cain
BBN
70 Fawcett Street
Cambridge, MA 02138
U.S.A.

E-Mail: pcain@bbn.com

Denis Pinkas
Integris
68 route de Versailles
B.P. 434
78430 Louveciennes
FRANCE

E-Mail: Denis.Pinkas@bull.net

Robert Zuccherato
Entrust, Inc.
1000 Innovation Drive
Ottawa, Ontario
K2K 3E7
CANADA

E-Mail: robert.zuccherato@entrust.com

APPENDIX A - Signature Time-stamp attribute using CMS

One of the major uses of time-stamping is to time-stamp a digital signature to prove that the digital signature was created before a given time. Should the corresponding public key certificate be revoked this allows a verifier to know whether the signature was created before or after the revocation date.

A sensible place to store a time-stamp is in a [CMS] structure as an unsigned attribute.

This appendix defines a Signature Time-stamp attribute that may be used to time-stamp a digital signature.

The following object identifier identifies the Signature Time-stamp attribute:

```
id-aa-timeStampToken OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) aa(2) 14 }
```

The Signature time-stamp attribute value has ASN.1 type
SignatureTimeStampToken:

```
SignatureTimeStampToken ::= TimeStampToken
```

The value of messageImprint field within TimeStampToken shall be a hash of the value of signature field within SignerInfo for the signedData being time-stamped.

APPENDIX B - Placing a Signature At a Particular Point in Time

We present an example of a possible use of this general time-stamping service. It places a signature at a particular point in time, from which the appropriate certificate status information (e.g., CRLs) MUST be checked. This application is intended to be used in conjunction with evidence generated using a digital signature mechanism.

Signatures can only be verified according to a non-repudiation policy. This policy MAY be implicit or explicit (i.e., indicated in the evidence provided by the signer). The non-repudiation policy can specify, among other things, the time period allowed by a signer to declare the compromise of a signature key used for the generation of digital signatures. Thus a signature may not be guaranteed to be valid until the termination of this time period.

To verify a digital signature, the following basic technique may be used:

- A) Time-stamping information needs to be obtained soon after the signature has been produced (e.g., within a few minutes or hours).
- 1) The signature is presented to the Time Stamping Authority (TSA). The TSA then returns a TimeStampToken (TST) upon that signature.
 - 2) The invoker of the service MUST then verify that the TimeStampToken is correct.
- B) The validity of the digital signature may then be verified in the following way:
- 1) The time-stamp token itself MUST be verified and it MUST be verified that it applies to the signature of the signer.
 - 2) The date/time indicated by the TSA in the TimeStampToken MUST be retrieved.
 - 3) The certificate used by the signer MUST be identified and retrieved.
 - 4) The date/time indicated by the TSA MUST be within the validity period of the signer's certificate.
 - 5) The revocation information about that certificate, at the date/time of the Time-Stamping operation, MUST be retrieved.
 - 6) Should the certificate be revoked, then the date/time of revocation shall be later than the date/time indicated by the TSA.

If all these conditions are successful, then the digital signature shall be declared as valid.

APPENDIX C: ASN.1 Module using 1988 Syntax

```
PKIXTSP {iso(1) identified-organization(3) dod(6) internet(1)
  security(5) mechanisms(5) pkix(7) id-mod(0) id-mod-tsp(13)}
```

```
DEFINITIONS IMPLICIT TAGS ::=
```

```
BEGIN
```

```
-- EXPORTS ALL --
```

```
IMPORTS
```

```

Extensions, AlgorithmIdentifier
FROM PKIX1Explicit88 {iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7)
id-mod(0) id-pkix1-explicit-88(1)}

GeneralName FROM PKIX1Implicit88 {iso(1)
identified-organization(3) dod(6) internet(1) security(5)
mechanisms(5) pkix(7) id-mod(0) id-pkix1-implicit-88(2)}

ContentInfo FROM CryptographicMessageSyntax {iso(1)
member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
smime(16) modules(0) cms(1)}

PKIFreeText FROM PKIXCMP {iso(1) identified-organization(3)
dod(6) internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
id-mod-cmp(9)} ;

-- Locally defined OIDs --

-- eContentType for a time-stamp token

id-ct-TSTInfo OBJECT IDENTIFIER ::= { iso(1) member-body(2)
us(840) rsadsi(113549) pkcs(1) pkcs-9(9) smime(16) ct(1) 4}

-- 2.4.1

TimeStampReq ::= SEQUENCE {
    version                INTEGER { v1(1) },
    messageImprint         MessageImprint,
    --a hash algorithm OID and the hash value of the data to be
    --time-stamped
    reqPolicy              TSAPolicyId                OPTIONAL,
    nonce                  INTEGER                    OPTIONAL,
    certReq                BOOLEAN                    DEFAULT FALSE,
    extensions              [0] IMPLICIT Extensions  OPTIONAL }

MessageImprint ::= SEQUENCE {
    hashAlgorithm          AlgorithmIdentifier,
    hashedMessage          OCTET STRING }

TSAPolicyId ::= OBJECT IDENTIFIER

-- 2.4.2

TimeStampResp ::= SEQUENCE {
    status                 PKIStatusInfo,
    timeStampToken         TimeStampToken            OPTIONAL }

Adams, et al.                Standards Track                [Page 22]

```

-- The status is based on the definition of status
-- in section 3.2.3 of [RFC2510]

```
PKIStatusInfo ::= SEQUENCE {
    status          PKIStatus,
    statusString    PKIFreeText    OPTIONAL,
    failInfo        PKIFailureInfo OPTIONAL }
```

```
PKIStatus ::= INTEGER {
    granted          (0),
    -- when the PKIStatus contains the value zero a TimeStampToken, as
    -- requested, is present.
    grantedWithMods (1),
    -- when the PKIStatus contains the value one a TimeStampToken,
    -- with modifications, is present.
    rejection       (2),
    waiting         (3),
    revocationWarning (4),
    -- this message contains a warning that a revocation is
    -- imminent
    revocationNotification (5)
    -- notification that a revocation has occurred }

-- When the TimeStampToken is not present
-- failInfo indicates the reason why the
-- time-stamp request was rejected and
-- may be one of the following values.
```

```
PKIFailureInfo ::= BIT STRING {
    badAlg          (0),
    -- unrecognized or unsupported Algorithm Identifier
    badRequest     (2),
    -- transaction not permitted or supported
    badDataFormat  (5),
    -- the data submitted has the wrong format
    timeNotAvailable (14),
    -- the TSA's time source is not available
    unacceptedPolicy (15),
    -- the requested TSA policy is not supported by the TSA.
    unacceptedExtension (16),
    -- the requested extension is not supported by the TSA.
    addInfoNotAvailable (17)
    -- the additional information requested could not be understood
    -- or is not available
    systemFailure   (25)
    -- the request cannot be handled due to system failure }
```

```
TimeStampToken ::= ContentInfo
```



```

-- contentType is id-signedData as defined in [CMS]
-- content is SignedData as defined in([CMS])
-- eContentType within SignedData is id-ct-TSTInfo
-- eContent within SignedData is TSTInfo

```

```

TSTInfo ::= SEQUENCE {
    version                INTEGER { v1(1) },
    policy                 TSAPolicyId,
    messageImprint        MessageImprint,
    -- MUST have the same value as the similar field in
    -- TimeStampReq
    serialNumber          INTEGER,
    -- Time-Stamping users MUST be ready to accommodate integers
    -- up to 160 bits.
    genTime               GeneralizedTime,
    accuracy              Accuracy                OPTIONAL,
    ordering              BOOLEAN                DEFAULT FALSE,
    nonce                INTEGER                OPTIONAL,
    -- MUST be present if the similar field was present
    -- in TimeStampReq. In that case it MUST have the same value.
    tsa                  [0] GeneralName        OPTIONAL,
    extensions            [1] IMPLICIT Extensions OPTIONAL }

```

```

Accuracy ::= SEQUENCE {
    seconds              INTEGER                OPTIONAL,
    millis              [0] INTEGER (1..999)  OPTIONAL,
    micros              [1] INTEGER (1..999)  OPTIONAL }

```

END

APPENDIX D: Access descriptors for Time-Stamping.

[This annex describes an extension based on the SIA extension that will be defined in the "son-of-RFC2459". Since at the time of publication of this document, "son-of-RFC2459" is not yet available, its description is placed in an informative annex. The contents of this annex will eventually become incorporated into the son-of-RFC2459 document, at which time this annex will no longer be needed. A future version of this document will likely omit this annex and refer to son-of-RFC2459 directly.]

A TSA's certificate MAY contain a Subject Information Access (SIA) extension (son of RFC2459) in order to convey the method of contacting the TSA. The accessMethod field in this extension MUST contain the OID id-ad-timestamping:

The following object identifier identifies the access descriptors for time-Stamping.

```
id-ad-timeStamping OBJECT IDENTIFIER ::= {iso(1)
    identified-organization(3) dod(6)
    internet(1) security(5) mechanisms(5) pkix(7)
    ad (48) timestamping (3)}
```

The value of the accessLocation field defines the transport (e.g., HTTP) used to access the TSA and may contain other transport dependent information (e.g., a URL).

Full Copyright Statement

Copyright (C) The Internet Society (2001). All Rights Reserved.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this paragraph are included on all such copies and derivative works. However, this document itself may not be modified in any way, such as by removing the copyright notice or references to the Internet Society or other Internet organizations, except as needed for the purpose of developing Internet standards in which case the procedures for copyrights defined in the Internet Standards process must be followed, or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by the Internet Society or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Acknowledgement

Funding for the RFC Editor function is currently provided by the Internet Society.