C. Adams(Entrust Technologies)
P. Cain (BBN)
D. Pinkas (Bull)
R. Zuccherato(Entrust Technologies)

Internet Draft
expires in six months
June 4, 1998

Time Stamp Protocols

<draft-adams-time-stamp-02.txt>

Status of this Memo

Abstract

This document describes the format of the data returned by a Time Stamp Authority and the protocols to be used when communicating with it. The time stamping service can be used as a Trusted Third Party (TTP) as one component in building reliable non-repudiation services (see [ISONR]). In order to reduce the amount of trust required of a TSA we introduce (in Appendix C) the optional Temporal Data Authority (TDA) whose function is to provide further corroborating evidence of the time contained in the token. We also give an example of how to place a signature at a particular point in time, from which the appropriate certificate status information (e.g. CRLs) may be checked.

1. Introduction

In order to associate a message 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 message at an instant in time. A TSA may also be used when a trusted time reference is required and when the local clock available cannot be trusted by all parties. The TSA's role is to time stamp a message to establish evidence indicating the time before which the message was generated. This can then be used, for example, to verify that a digital signature was applied before the certificate was revoked thus allowing a revoked public key certificate to be used for verifying signatures created prior to the time of revocation.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.

2. The TSA

The TSA is a TTP that creates time stamp tokens in order to indicate that a message 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, contains the correct TSA name, and is from a supported TSA subscriber.

2.1. Requirements of the TSA

The TSA is required:

1. to provide a trusted source of time.
2. not to examine or verify the requesting entities in any way.
3. not to examine the imprint being time stamped in any way.
4. to include a monotonically incrementing value of the time of day into its time stamp token.
5. to produce a time stamp token upon receiving a valid request from the requester.
6. to include within each time stamp token an identifier to uniquely indicate the trust and validation policy under which the token was created.
7. to only time stamp a hash representation of the message, i.e. a data imprint associated with a one-way collision resistant hash-function OID.
8. to examine the OID of the one-way collision resistant hash-function and to verify that this function is "sufficient" (see Section 2.4).
9. 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.
10. to include supplementary temporal information in the time stamp token (from TDA's) if asked by the requester. If this is not possible, the TSA shall respond with an error message.
11. to provide a signed receipt (i.e. in the form of an appropriately defined time stamp token) to the requester, where appropriate, as defined by policy.

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 TimeStampToken, as defined below) to the requesting entity.

Upon receiving the token, the requesting entity verifies its validity by verifying the digital signature in the TimeStampToken and by verifying that what was time stamped corresponds to what was requested to be time stamped. The requester should verify that the TimeStampToken

contains the correct TSA name, the correct data imprint and the correct hash algorithm OID. It should 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, and/or the value of the nonce included in the response against the value included in the request. 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. If TemporalDataToken's are included in the TimeStampToken, then these should also be verified as was the TimeStampToken (see Appendix C). The token can now be used to establish a trusted time reference.

2.3. Identification of the TSA

The TSA must sign all time stamp messages with a key reserved specifically for that purpose. The corresponding certificate must contain the extended key usage field extension as defined in [CCP] Section 4.2.1.14 with KeyPurposeID having value id-kp-timeStamping. This extension must be critical.

2.4. Request and Token Formats

A time stamping request is as follows.

```
TimeStampReq ::= SEQUENCE  {
     version                      Integer  { v1(0) },
     reqPolicy                    PolicyInformation OPTIONAL,
     tdas                         SEQUENCE OF GeneralName OPTIONAL,
     nonce                        Integer,
     messageImprint               MessageImprint
       --a hash algorithm OID and the hash value of the data to be
       --time stamped
}
```

The reqPolicy field, if included, indicates the policy under which the

TimeStampToken should be provided. PolicyInformation is defined in Section 4.2.1.5 of [CCP].

The tdas field identifies those TDAs which are requested to provide supplementary temporal evidence in the time stamp token. (See Appendix C.)

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

The hash algorithm indicated in the hashAlgorithm field must be a strong hash algorithm. That means that it must be one-way and collision resistant. It is up to the Time Stamp Authority to decide whether or not the given hash algorithm is "sufficient" (based on the current state of knowledge in cryptanalysis and the current state of the art in computational resources, for example).

The hashedMessage field should contain the hash of the message to be time stamped. The hash is represented as an OCTET STRING.

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, it is suggested that alternate identification means be used (e.g. CMS encapsulation or SSL authentication).

A TimeStampToken is as follows. It is encapsulated as a SignedData construct [CMS]. The content is of type TSTInfo, which is indicated by the OID:

```
TSTInfo OBJECT IDENTIFIER  ::= { ??????  }
```

The time stamp token must contain only the signature of the TSA. In some environments, the CA might not perform a proof-of-possession of the private key when issuing certificates. In these instances, either the certificate of the TSA, or the certificate issuer and serial number shall be included as an authenticated attribute.

```
TSTInfo ::= SEQUENCE  {
     version                      Integer  { v1(0) },
     policy                       PolicyInformation,
     status                       PKIStatusInfo,
     tsa                          GeneralName,
     genTime                      GeneralizedTime,
     tdaTokens                    SEQUENCE OF TemporalDataToken
                                    OPTIONAL,
     nonce                        Integer,
       --this field must have the same value as the similar field
       --in TimeStampReq
     messageImprint               MessageImprint,
       --this field must have the same value as the similar field
       --in TimeStampReq
     tsaFreeData                  OCTET STRING OPTIONAL
       --contains supplementary information from the TSA
}
```

PKIStatusInfo is defined in Section 3.2.3 of [CMP]. If the PKIStatus field has value `waiting' (3), then this token is a receipt, as defined in Section 2.1. Otherwise, the status field is present to indicate whether or not the time stamping request was fulfilled and, if not, the reason it was rejected. A valid time stamp token will always have value **0 (granted) in the PKIStatus field of PKIStatusInfo.** Since not all environments will require the use of receipts, support for the value `waiting' is optional.

```
PKIFailureInfo ::= BITSTRING  {
    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 TSAs time source is not available
    tdaNotAvailable  {15},
      -- at least one of the TDAs that were requested isn't available
}
```

The statusString field of PKIStatusInfo can be used to include reason text such as "messageImprint field is missing".

The purpose of the tsa field is to identify the name of the TSA. It must correspond to one of the subject names included in the certificate that is to be used to verify the token.
TemporalDataToken is defined in Appendix C of this document. The tdaTokens field contains the supplementary evidence requested in the TimeStampReq.

There may be situations where the TSA may wish to include supplementary non-time stamp related information in the time stamp token (e.g. billing information, usage statistics, etc.). The format of this information is TSA dependant and the value can be placed in the tsafreedata field as an OCTET STRING. Conformant clients are not required to process this field, if present.

3. Transports

There is no mandatory transport mechanism in this document. All mechanisms are optional.

3.1. File Based Protocol

A file containing a time stamp message must contain only the DER encoding of one PKI 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.

3.2. Socket Based Protocol

The socket based protocol for time stamp messages is identical to that used in [CMP] Section 5.2 except that port 309 must be used.

3.3. 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 C via Internet mail.

A simple MIME object is specified as follows.

```
   Content-Type: application/timestamp
   Content-Transfer-Encoding: base64

   <<the ASN.1 DER-encoded Time Stamp message, base64-encoded>>
```

This MIME object can be sent and received using common MIME processing engines and provides a simple Internet mail transport for Time Stamp messages.

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 C via the HyperText Transfer Protocol.

A simple MIME object is specified as follows.

```
   Content-Type: application/timestamp

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

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

4. Security Considerations

This entire document concerns security considerations.

When designing a TSA/TDA service, the following considerations have been identified that have an impact upon the validity or "trust" in the time stamp token.

1. When there is a reason to believe that the TSA can no longer be trusted, the authority's certificate must be revoked and placed on the appropriate CRL. Thus, at any future time the tokens signed with the corresponding key will not be valid.
2. The TSA private key is compromised and the corresponding certificate is revoked. In this case, any token signed by the TSA using that private key cannot be trusted. 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 tokens.
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. Since the TSA does not verify message data or the identity of the entities, the requester field in TimeStampReq and TimeStampToken should be considered untrusted. If authentication of this field is needed, it is recommended that the Notary Authority be used, as described in [NOTARY].
5. An application using the TSA service 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 TimeStampToken that takes more than an acceptable period of time should be considered suspect.

6. In certain circumstances, a TSA may not be able to produce a valid response to a request (for example, if it is unable to compute signatures for a period of time). In these situations the TSA must wait until it is again able to produce a valid response before responding, if this is possible. If this is not possible, it must ignore the requests and not respond. Under no circumstances shall a TSA produce an unsigned response to a request.

7. This protocol assumes that the CA has conducted a test for proof of possession for each user's signing private key (including the TSA signing private key). If this is not the case, or when additional assurances are required, the certificate or certificate serial number and issuer of the TSA shall be included in the encapsulation of the time stamp token as an authenticated attribute.

5. Patent Information

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

# 4,309,569   Method of Providing Digital Signatures
(issued) January 5, 1982
(inventor) Ralph C. Merkle
(assignee) The Board of Trustees of the Leland Stanford Junior
University

# 5,001,752   Public/Key Date-Time Notary Facility
(issued) March 19, 1991
(inventor) Addison M. Fischer

# 5,022,080   Electronic Notary
(issued) June 4, 1991
(inventors) Robert T. Durst, Kevin D. Hunter

# 5,136,643   Public/Key Date-Time Notary Facility
(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
(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
(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
(issued) December 13, 1994
(inventors) Stuart A. Haber, Wakefield S. Stornetta Jr.
(assignee) Bell Communications Research, Inc.,

# 5,422,95 Personal Date/Time Notary Device
(issued) June 6, 1995
(inventor) Addison M. Fischer

6. References

[CMP] C. Adams, S. Farrell, "Internet Public Key Infrastructure, Certificate Management Protocols," draft-ietf-pkix-ipki3cmp-0X.txt, 1997 (work in progress).

[NOTARY] C. Adams, R. Zuccherato, "Notary Protocols," draft-adams-notary-0X.txt, 1998 (work in progress).

[CCP] R. Housley, W. Ford, W. Polk, D. Solo, "Internet Public Key Infrastructure, X.509 Certificate and CRL Profile," draft-ietf-pkix-ipki-part1-0X.txt, 1997 (work in progress).

[ISONR] ISO/IEC 10181-5: Security Frameworks in Open Systems. Non-Repudiation Framework.

[CMS] R. Housley "Cryptographic Message Syntax", draft-ietf-smime-cms-02.txt, 1998 (work in progress).

7. Authors' Addresses

Carlisle Adams
Entrust Technologies
750 Heron Road
Ottawa, Ontario
K1V 1A7
CANADA
cadams@entrust.com

Pat Cain
BBN
70 Fawcett Street
Cambridge, MA 02138
U.S.A.
pcain@bbn.com

Denis Pinkas
Bull S.A.
Rue Jean Jaures
B.P. 68
78340 Les Clayes sous Bois
FRANCE
Denis.Pinkas@bull.net

Robert Zuccherato
Entrust Technologies
750 Heron Road
Ottawa, Ontario
K1V 1A7
CANADA
robert.zuccherato@entrust.com

APPENDIX A - Storage of Data and Token

A time stamp token is meaningless without its associated data. Thus, a method is required to allow users to store the data and token together securely. They may be stored as a PKCS #7 SignedData object as described in [CMS]. That is, the contentType is signedData and contentInfo is Data, which contains the message associated with the time stamp token. The SignedData object is signed by the person storing the data and token. This signature is to be used only for storage and for verifying the integrity of the token and data. Anyone using the token and data at some future time must verify the data and token at that time. This is just a method for keeping the two pieces of information together, with some integrity.

For this purpose, we define a PKCS #9 [PKCS9] time stamp token attribute type. This attribute type specifies the time stamp token, which must be included as an authenticated attribute of the SignedData object. The time stamp token attribute type has ASN.1 type TimeStampToken (as defined in Section 2.4 of this document). A time stamp token attribute must have a single attribute value.

The object identifier timeStampToken identifies the time stamp token attribute type.

```
timeStampToken ::= { pkcs-9 n <<To be supplied>> }
```

[CMS] R. Housley "Cryptographic Message Syntax", draft-ietf-smime-cms-02.txt, 1998 (work in progress).

[PKCS9] RSA Laboratories, "The Public-Key Cryptography Standards (PKCS)", RSA Data Security Inc., Redwood City, California, November **1993 Release.**

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 signature that incorporates an untrusted time, the following basic technique may be used:

A) Time stamping information needs to be obtained by the signer or a verifier.

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 evidence must be verified :

1) The date/time indicated by the signer in the signature shall be compared with the date/time in the TST. If they are not close enough (e.g., less than a few hours) the evidence is considered to be invalid.
2) The certificate included in the signed message should be verified to be valid at the time of the signature. It must first be verified and then the appropriate CRL must be checked.

The signature has now been placed at a particular point in time. The appropriate CRLs or other certificate status information mechanism may be examined to determine the validity of the signature at that time.

Appendix C - The TDA

The Temporal Data Authority is a TTP that creates a temporal data token. This temporal data token associates a message with a particular event and provides supplementary evidence for the time included in the time stamp token. For example, a TDA could associate the message with the most recent closing value of the Dow Jones Average. The temporal data with which the message is associated should be unpredictable in order to prevent forward dating of tokens. Authentic values of this data should also be available from a large number of trustworthy sources in order to make collusion or corruption of data more difficult. For a list of possible types of temporal data, see Appendix D.

C.1. Requirements of the TDA

The TDA is required:

1. to only provide a trusted source of temporal data.
2. not to examine the imprint being time stamped.
3. to include the current data associated with a specific

unpredictable event in each temporal data token.

4. to produce a temporal data token upon receiving a valid request from the TSA.
5. to only produce a temporal data token on a hash representation of the message.
6. to sign each temporal data token using a key generated exclusively for this purpose and have this property of the key indicated on the corresponding certificate.

C.2. TDA Transactions

As the first message of this mechanism, the TSA requests a temporal data token by sending a request (which is or includes a TemporalDataReq, as defined below) to the TDA. As the second message, the TDA responds by sending a response (which is or includes a TemporalDataToken, as defined below) to the TSA.

C.3. Verifying a TemporalDataToken

The TSA is required to verify the structure of the TemporalDataToken. It must verify the digital signature in the TemporalDataToken and also verify that what was signed corresponds to what was requested to be signed. The requester should verify that the TemporalDataToken contains the correct TDA name, the correct data imprint and the correct hash algorithm OID. It should also verify the timeliness of the response by verifying the value of the nonce included in the response against the value included in the request (exact match needed). Since the TDA'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.

In order to verify the TemporalData inside a TemporalDataToken, it is necessary to know the form of the temporal data that the TDA has included in the token.

The TSA is not required to verify the TemporalData. However, either the entity requesting a Time Stamping Token or an entity verifying a Time Stamping Token containing temporal information may be interested in such a verification.

In the first case, it is unlikely that the temporal information will be available ahead of time and thus the entity requesting a Time Stamping Token may need to enter into an online protocol with the TDA, or some other entity, to obtain it. A secure link with that trusted source will be necessary, i.e. the communication channel or the information itself must be authenticated and integrity protected. Such a protocol is TDA dependent and is outside the scope of this document.

In the second case, if the verification occurs some time after the Time

Stamping Token has been produced, then it is possible to rely on an authentic source (e.g. a newspaper or a CD-ROM) to verify it against. The exact method of verification is TDA dependent and is thus outside the scope of this document.

C.4. Identification of the TDA

The TDA must sign all temporal data tokens with a key reserved specifically for that purpose. The corresponding certificate must contain the extended key usage field extension as defined in [CCP] Section 4.2.1.14 with KeyPurposeID having value id-kp-temporalData. This extension must be critical.

```
id-kp-temporalData    OBJECT IDENTIFIER ::= {id-kp  ??}
  -- Providing temporal data in support of time stamping services.  Key
  -- usage bits that may be consistent:  digitalSignature,
  -- nonRepudiation
```

C.5. Request and Token Formats

A temporal data request from a TSA is as follows.

```
TemporalDataReq ::= SEQUENCE  {
     version                      Integer  { v1(0) },
     nonce                        Integer,
       --must be the same value as the corresponding field in
       --TimeStampReq
     messageImprint               MessageImprint
       --a hash of the data to be time stamped, must be the same
       --value as the corresponding field in TimeStampReq
}
```

A TemporalDataToken is as follows. It is encapsulated as a SignedData construct [CMS]. The content is of type TDTInfo, which is indicated by the OID:

```
TDTInfo OBJECT IDENTIFIER  ::= { ??????  }
```

The temporal data token must contain only the signature of the TDA. In some environments, the CA might not perform a proof-of-possession of the private key when issuing certificates. In these instances, either the certificate of the TDA, or the certificate issuer and serial number shall be included as an authenticated attribute.

```
TDTInfo ::= SEQUENCE  {
     version                      Integer  { v1(0) },
     tda                          GeneralName,
     nonce                        Integer,
       --must have the same value as the corresponding field in
       --TimeStampReq
     temporalData                 TemporalData,
     messageImprint               MessageImprint
       --must have the same value as the corresponding field in
       --TimeStampReq
}
```

The temporalData field contains the actual temporal data that will be used as substantiating evidence in the time stamp token.

```
TemporalData ::= SEQUENCE  {
     format                       TEMPORALDATACLASS.&id,   --objid
     rawdata                      TEMPORALDATACLASS.&Type  --open type
}

TEMPORALDATACLASS ::= CLASS  {
     &id                          OBJECT IDENTIFIER UNIQUE,
     &Type                                                    }
WITH SYNTAX  { &Type IDENTIFIED BY &id }
```

C.6. Security Considerations

When designing a TDA service, the following considerations have been identified that have an impact upon the validity or "trust" in the temporal data token.

1. When there is a reason to believe that the TDA can no longer be trusted, the authority's certificate must be revoked and placed on the appropriate CRL. Thus, at any future time the tokens signed with the corresponding key will not be valid.
2. The TDA private key is compromised and the corresponding certificate is revoked. In this case, any token signed by the TDA using that private key cannot be trusted. For this reason, it is imperative that the TDA'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 TDA may provide a means to discriminate between genuine and false tokens.
3. The TDA 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 time stamp token containing the TDA's signature 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 TDA's signature. Time stamp tokens could also be kept with

an Evidence Recording Authority to maintain this trust.

4. In certain circumstances, a TDA may not be able to produce a valid response to a request (for example, if it is unable to compute signatures for a period of time). In these situations the TDA must wait until it is again able to produce a valid response before responding, if this is possible. If this is not possible, it must ignore the requests and not respond. Under no circumstances shall a TDA produce an unsigned response to a request.
5. This protocol assumes that the CA has conducted a test for proof of possession for each user's signing private key (including the TDA signing private key).. If this is not the case, or when additional assurances are required, the certificate or certificate serial number and issuer of the TDA shall be included in the encapsulation of the temporal data token as an authenticated attribute.

[CMS] R. Housley "Cryptographic Message Syntax", draft-ietf-smime-cms-02.txt, 1998 (work in progress).

APPENDIX D - Possible Types of Temporal Data

1) Stock market information
2) Sports results
3) Official weather data for a specific location
4) Lottery results
5) Birth or death announcements in specific newspapers
6) Headlines in specific newspapers
7) Information linking the request with previous and subsequent requests (e.g. hash values) that can be verified against information that is made public by the TDA.
8) A signed packet from a secure time source.