PKIX Working Group C. Adams(Entrust Technologies)
Internet Draft P. Cain (BBN)
expires in six months D. Pinkas (Bull)
R. Zuccherato(Entrust Technologies)
July 29, 1997

Internet Public Key Infrastructure

Part V: Time Stamp Protocols

<draft-ietf-pkix-ipki5tsp-00.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]). We also give an example of how to place a signature at a particular point in time, from which the appropriate 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 \223proof-of-existence\224 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\222s 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 key was put on a CRL, 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. Requirements of the TSA

The TSA is required to:

1. include a monotonically incrementing value of the time of day into its time stamp token.
2. produce a time stamp token upon receiving a valid request from the requester.
3. include within each time stamp token an identifier to

uniquely determine the trust and validation policy used for this signature.

4. support time stamping of a hash representation of the message from the requester.

3. TSA Transactions

As the first message of this mechanism, the requesting entity requests a time stamping service 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 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 time, the correct TSA name, the correct data imprint and the correct hash algorithm OID. Since the TSA\222s certificate may have been revoked, the appropriate ARL should be checked to verify that the certificate is still valid. The token can now be used to establish a trusted time reference.

4. Request and Token Formats

A time stamping request is as follows.

```
TimeStampReq ::= SEQUENCE  {
     requester               [0] GeneralName OPTIONAL,
     reqPolicy               [1] PolicyInformation OPTIONAL,
     tsa                         GeneralName,
     messageImprint              MessageImprint
       --a hash 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 PKIX Part 1 [PKIX1].

The tsa field identifies the name of the TSA. GeneralName is defined in Section 4.2.1.7 of PKIX Part 1 [PKIX1].

```
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 \223sufficient\224 (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.

A time stamp token is as follows. The signature is computed over tstInfo (encoded using the ASN.1 distinguished encoding rules (DER)).

```
TimeStampToken ::= SEQUENCE  {
     tstInfo                     TSTInfo,
     signature                   BIT STRING,
       --over the ASN.1 DER encoding of tstInfo
}

TSTInfo ::= SEQUENCE  {
     policy                       PolicyInformation,
     status                       PKIStatusInfo,
     requester               [0]  GeneralName OPTIONAL,
       --must be present if the requester field is present in
       --TimeStampReq
     tsa                          GeneralName,
     signatureAlgorithm           AlgorithmIdentifier,
     certId                       CertId,
       --must refer to the TSA\222s public verification certificate
     certs                        SEQUENCE OF Certificate OPTIONAL,
     genTime                      GeneralizedTime,
     messageImprint               MessageImprint
       --this field must have the same value as the similar field
       --in TimeStampReq
}
```

PKIStatusInfo is defined in Section 3.2.3 of PKIX Part 3 [PKIX3]. 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.

CertId is defined in Section 3.2.4 of PKIX Part 3 [PKIX3].

5. Time Stamp Protocol Using E-mail

This section specifies a means for conveying ASN.1-encoded messages for the protocol exchanges described in Section 4 via Internet mail.

A simple MIME object is specified as follows.

```
   Content-Type: application/x-pkix5
   Content-Transfer-Encoding: base64

   <<the ASN.1 DER-encoded PKIX-5 message, base64-encoded>>
```

This MIME object can be sent and received using common MIME processing engines and provides a simple Internet mail transport for PKIX-5 messages.

7. Security Considerations

When designing a TSA service, the following considerations have been identified that have an impact upon the validity or \223trust\224 in the time stamp token.

1. 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\222s 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.

2. 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 at a later date to renew the trust that exists in the TSA\222s signature.

8. References

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

[PKIX1] R. Housley, W. Ford, W. Polk, D. Solo, \223Internet Public Key Infrastructure, Part I: X.509 Certificate and CRL Profile,\224 draft-ietf-pkix-ipki-part1-0X.txt, 1997 (work in progress).

[PKIX3] C. Adams, S. Farrell, \223Internet Public Key Infrastructure, Part III: Certificate Management Protocols,\224 draft-ietf-pkix-ipki3cmp-0X.txt, 1997 (work in progress).

9. Authors\222 Addresses

Carlisle Adams
Entrust Technologies
750 Heron Road, Suite 800
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
D.Pinkas@frcl.bull.fr

Robert Zuccherato
Entrust Technologies
750 Heron Road, Suite 800
Ottawa, Ontario
K1V 1A7
CANADA
robertz@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 [PKCS7]. 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.

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 4 of this document). A time stamp token attribute can have a single attribute value.

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

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

[PKCS7] RSA Laboratories, \223The Public-Key Cryptography Standards (PKCS)\224, RSA Data Security Inc., Redwood City, California, November 1993 Release.

[PKCS9] RSA Laboratories, \223The Public-Key Cryptography Standards (PKCS)\224, 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 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 may be examined to determine the validity of the signature at that time.