

# Trusted Archiving

Santosh Chokhani & Carl Wallace  
Orion Security Solutions

## Abstract

Digital signatures are a powerful tool for demonstrating data integrity and performing source authentication. Timestamps are a powerful tool for confirming data existence by a particular point in time. Today, the value of digital signatures (and timestamps containing digital signatures) is limited due to a lack of tools and techniques that address the problems associated with digital signatures that accrue over time, including: expiration, revocation, cryptanalytic advances and computational advances. In this paper, we describe a system concept and protocol to achieve secure storage of data for long periods with preservation of integrity. The approach uses periodically refreshed time stamps to address these problems. The techniques can be used for a wide variety of applications, including those requiring long-term non-repudiation of digital signatures. The concept and protocol are based on minimizing trust in individual system components in order to reduce the security requirements for those components and to enhance the trust in the overall system. A proof-of-concept implementation based on the ideas and protocol described in this paper has been developed and successfully tested.

## 1. Introduction

One of the challenges of using digital signatures is how to prove the validity of signatures well into the future when the signer's, or a related certification authority's, credentials are no longer valid or available. Trusted archiving is a process that involves the active storage of data where evidence is periodically obtained, or generated, and stored to create an unbroken history demonstrating the integrity of data from storage time to verification time. Trusted archives are a missing piece of the PKI puzzle that are required if digital signatures are to have a durability similar to paper and ink signatures.

We have designed and developed a client-server system that addresses this problem. This paper describes our work. Section 2 contains the system concept. Section 3 provides an overview of the client-server protocol for implementing the system. Section 4 describes some security considerations. Section 5 provides a summary of the implemented system\*. Section 6 describes lessons learned. Section 7 describes future plans.

---

\* The ideas and work described in this paper (including the proof-of-concept) were funded by the United States Marine Corps.

## 2. Trusted Archive System Concept

A trusted archive should meet the following requirements, at a minimum:

- Provide evidence to demonstrate the integrity and, optionally, the source of data after the expiration of the cryptanalysis period for related keys and algorithms.
- Provide evidence to demonstrate the integrity and, optionally, source of data if a related certification authority (CA) is no longer operational.
- Provide active controls to protect the integrity of archived information.†

The central component of the solution is a trusted archive authority (TAA). A TAA accepts data for long-term storage and is responsible for ensuring that an evidence trail is produced and stored to enable demonstration of data integrity at any point in the future. TAAs participate in client-server transactions

---

† Many cryptographic mechanisms (such as digital signature or HMAC) are detection mechanisms with regard to integrity and source authentication. From a practical viewpoint, a trusted archive service needs to ensure that the archived information is protected from tampering. The mechanisms described in this paper extend the detection mechanisms and are not a substitute for secure, redundant storage, tamper protection, etc.

with entities seeking to exercise the TAA's services. TAAs use current credentials to generate signed responses as part of these transactions. Clients verify TAA signatures using a trust anchor known to the client at the time of the transaction.

Upon submission and periodically thereafter, the TAA obtains or generates a new time stamp for archived data in order to account for cryptanalytic advances against hashing or signature algorithms and to account for expiration of TSA keys. This periodic acquisition of new time stamps is referred to as "time stamp refresh" throughout the remainder of this document. The amount of trust invested in a TAA can be minimized by using the services of a trusted time stamp authority (TSA) to obtain time stamps for archived data instead of generating timestamps directly.

The client-server protocol between the client and TAA is a simple set of request/response transactions that enable submission of data to a TAA and retrieval or deletion<sup>‡</sup> of data from a TAA. The transactions are defined in ASN.1 and, generally, are DER encoded. Cryptographic Message Syntax (CMS), defined in [CMS], is used for all digital signatures. CMS was chosen because it is an IETF standard, many products support it, it provides flexibility to apply the cryptographic services appropriate for the application, and it provides the flexibility to include time stamps, certificates, revocation information, etc. as needed. An ASN.1 based protocol was chosen due to the requirement for an ASN.1 encoder/decoder to process most PKI artifacts. An XML submission format could be defined to provide a broad entry to a TAA. Retrieval should be sufficiently rare and in need of special purpose software, e.g. for historic algorithms, to be sustainable by a single format.

The TAA is designed to securely archive information of any type and need not have knowledge of the format of archived data. Where archived data contains digital signatures that must be verifiable in the future, collection and packaging of the items required to support signature verification are the responsibility of the archive submitter. Given the likelihood that the submitter will have performed verification, this requirement is not particularly onerous and can be easily implemented by packaging the artifacts from that verification operation, e.g. trust anchors, certificates, Certificate Revocation Lists (CRLs), Online Certificate Status Protocol (OCSP) responses, Simple Certificate Validation Protocol

(SCVP) responses, etc., with the data to archive prior to submission to the TAA. Alternatively, a TAA may provide server-side verification services to simplify and streamline the process of verifying and archiving data.

Trust anchors will come and go over the course of time but always must be obtained in a trusted manner to support certificate path validation. A TAA may archive a set of trust anchors that can be provided to retrieval clients. This capability allows the retriever to validate a digital signature without having to rely on the good intentions of the original submitter. This capability also permits a functional separation in order to enhance the trustworthiness of the archival service, i.e. one TAA can be store archived data and evidence and another TAA can store trust anchors.

In summary, the system concept consists of the following:

- TAAs use digital signatures to demonstrate the integrity and source of responses from the TAA. This is primarily a concern where responses contain trust anchors.
- TAAs periodically refresh time stamps in order to protect against advances in technology that can break hash and signature algorithms and to maintain verifiability in cases of key (or certificate) expiration.
- Archive submission clients collect and submit all information (e.g., certificates, revocation information, SCVP responses, etc.) required for long-term non-repudiation of digital signatures that cover the data submitted to a TAA.
- Archive retrieval clients verify the signatures on the TAA response and on the associated archive record to confirm the integrity of the data. The retrieval client may use trust anchors from one or more of the following sources to verify signatures contained in the evidence record or in archived data itself, if the archived data was signed:
  - Trust anchors from the signed retrieval response.
  - Trust anchors obtained independently from the same or different TAA.
  - Trust anchors known to the retrieval client or obtained via other out-of-band means.

---

<sup>‡</sup> Where a TAA maintains archived data on write-only media, deletion may simply be cessation of refresh operations rather than actual deletion. Deletion is not addressed in detail in this document.

### 3. Trusted Archive Protocol (TAP)

#### 3.1 Assumptions and Background

The Trusted Archive Protocol (TAP) was developed and submitted to the IETF as an Internet Draft (I-D) of the PKIX working group. The TAP I-D was a contributing factor in the formation of the IETF Long Term Archive and Notary (LTANS) working group (WG) and has served as input to the protocol being produced by that group. Activities of the LTANS WG and its relationship to this work are described in Further Research section.

TAP was designed using the following principles:

- The CMS will be used in all cases where digital signatures are applied.
- A TAA shall provide an archive submitter a response that includes a time stamp token, identifying information and a TAA-generated digital signature. Clients can verify the timestamp token to confirm the correct data was received and (presumably) archived by the TAA.
- The TAA shall verify the time stamp token received from the TSA in accordance with RFC 3161 [TSP].
- The TAA shall periodically refresh the time stamp token.
- The TAA shall provide all timestamps obtained for the archived data in the response.

The rest of this section provides a summary of the protocol defined in [TAP].

#### 3.2 Definitions

During the development of TAP, the need arose for a common vocabulary describing the various processes and artifacts involved in archiving. The following terms were defined to meet this need:

**Archived data:** archived data is the data presented to the TAA by the submitter.

**Archive token:** an archive token is an object generated by the TAA when data is submitted and accepted for archiving. The archive token is returned to the submitter and may be used to request retrieval or deletion of the archived data and associated cryptographic information. For purposes of future retrieval or deletion, applications may treat the archive token as an opaque blob. The archive token

includes: submitter DN, timestamp token, TAA date and time upon submission and, optionally, tracking information.

**Archive record:** an archive record contains the cryptographic refresh history compiled by the TAA. The initial archive record is the timestamp token obtained for the submitted data. The timestamp token format is defined in [TSP] and consists of a ContentInfo object containing a TSTInfo object. Upon each refresh, the most recent archive record becomes the prevArchRecord field of a new TimeStampedData object, a timestamp is obtained for the TimeStampedData object and is placed in the timestamp field of a new ArchiveRecordData and the entire ArchiveRecordData structure placed in a ContentInfo object. The ContentInfo object serves as the new archive record. When verifying an archive record, verification terminates when the original timestamp token is verified against the archived data.

**Archive package:** an archive package is an object containing, minimally, the archive token, archive record and archived data. The archive package may include additional cryptographic information. Archive packages are returned during retrieval.

Figure 1 illustrates the communication protocol among the TAA and the clients.

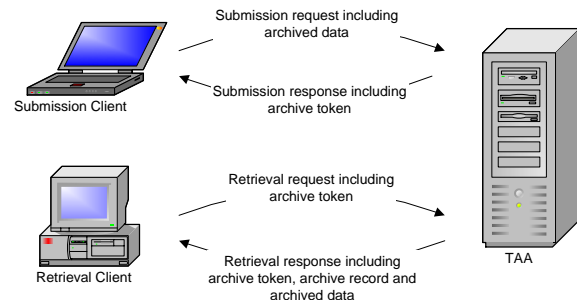


Figure 1 Client interactions with TAA

Figure 2 illustrates the archive record that is maintained by the TAA. The archive record contains nested timestamps with a timestamp covering the archived data at its innermost layer.

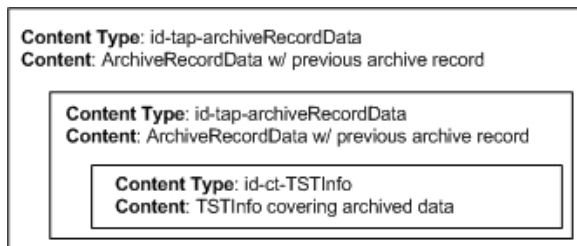


Figure 2 Archive record after two refresh operations

### 3.3 Protocol Summary

The [TAP] protocol defines 3 request types: submission, retrieval and deletion. The steps involved in a submission request are as follows:

- 1) A client prepares a data object for submission to a TAA. If the data object is signed, the client verifies the object and includes the necessary material to verify the object (except the trust anchor) in the object itself, e.g. in a certificate or CRL bag. Optionally, the client signs the request. The client sends the request to the TAA.
- 2) The TAA receives the request and verifies the signature on the request, if present. The TAA unpacks the data object and prepares a TSP request. Optionally, the TAA signs the TSP request. The TAA then sends the TSP request to a TSA.
- 3) The TSA receives the response and verifies the signature on the request, if present. The TSA then generates a signed timestamp token and returns it to the TAA.
- 4) The TAA stores the archived data and the timestamp token and starts the refresh clock for the archived data. The timestamp token is packaged in an archive token along with additional information. The archive token is included in a signed response and returned to the client.
- 5) The client verifies the signature on the response. The client verifies the archive token to ensure the correct data was archived by the TAA. The client may store the archive data along with the original data item, e.g. as an unsigned attribute.

The steps involved in a retrieval request are as follows:

- 1) A client prepares a retrieval request containing the archive token of the data item for which an archive record is required. The

client signs the request and sends it to the TAA.

- 2) The TAA verifies the signature on the request and confirms the requestor has access to the requested data item. The TAA prepares an ArchivePackage containing the refresh history compiled for the requested data item, packages it in a signed response and returns it to the client.
- 3) The client verifies the signature on the response then verifies the archive package. The outermost layer in the archive record is verified using a current trust anchor. Interior layers are verified to a trust anchor provided by the TAA in the archive package.

### 3.4 Protocol Data Formats

The section describes some of the key data formats defined in [TAP].

#### Archive Submission

Archive submission requests are defined as follows:

```
ArchiveSubmissionReq ::= SEQUENCE
{
    version          TAPVersion DEFAULT v1,
    submitterName   GeneralName,
    policy          OBJECT IDENTIFIER
                  OPTIONAL,
    archiveControls [0] ArchiveControls
                  OPTIONAL,
    archivedData    ArchivedData
}
```

#### Archive Data

Archived data, i.e. data submitted to a TAA for preservation, has the following format.

```
ArchivedData ::= SEQUENCE
{
    type    ArchivedDataType OPTIONAL,
    data    OCTET STRING
}
ArchivedDataType ::= CHOICE
{
    oid        OBJECT IDENTIFIER,
    mimeType   UTF8String
}
```

#### Archive Submission Response

Archive submission responses are defined as follows:

```
ArchiveSubOrDelResp ::= SEQUENCE
{
    version          TAPVersion DEFAULT v1,
    status          ArchiveStatus,
    archiveToken    ArchiveToken
}
```

```

        archiveControls  [0] ArchiveControls
        OPTIONAL
    }

```

### **Archive Token**

Archive tokens have the following format.

```

ArchiveToken ::= ContentInfo
-- content type: id-tap-archiveToken
-- content: ArchiveTokenData
ArchiveTokenData ::= SEQUENCE
{
    submitterName      GeneralName,
    timestamp          TimeStampToken,
    curTime            GeneralizedTime,
    trackingInfo       TrackingInfos
                    OPTIONAL
}

```

The archiveControls field can be used to return information associated with a control included in the request, for example, the outcome of server-side validation or a nonce from the request. TAAs must not include controls in a response that are not associated with controls in a request. Submission clients should be able to process controls in accordance with the control definition.

### **Archive Record**

The archive record contains a nested structure with the complete refresh history for the archived data. TAAs should store all cryptographic information necessary to verify each layer of the archive record in the certificates, CRLs and unsignedAttrs fields of the timestamp token, i.e. each timestamp token in the history should be self-contained for validation purposes under protection of the next layer in the archive record. A CryptoInfos unsignedAttrs field may be used to convey OCSP responses and/or trust anchor information. Archive record has the following format:

```

ArchiveRecord ::= ContentInfo
-- content type: id-tap-archiveRecordData
-- content: ArchiveRecordData

ArchiveRecordData ::= SEQUENCE
{
    timestampedData    TimeStampedData,
    timestamp          TimeStampToken
}
TimeStampedData ::= SEQUENCE
{
    prevArchRecord    ContentInfo,
    messageImprint    MessageImprint
}

```

The cryptoInfos field contains additional information that may be useful when verifying the archived data.

### **Archive Package**

Archive packages are defined as follows:

```

ArchivePackage ::= SEQUENCE
{
    archiveToken      ArchiveToken,
    packageData      [0] ArchivePackageData
                    OPTIONAL,
    pollReference     [1] OCTET STRING
                    OPTIONAL
}
ArchivePackageData ::= SEQUENCE
{
    digestAlgs        DigestAlgorithmIdentifiers,
    policy            OBJECT IDENTIFIER
                    OPTIONAL,
    archRecord        ArchiveRecord,
    cryptoInfos [0] CryptoInfos
                    OPTIONAL,
    archivedData      ArchivedData
}

```

## **4. Security Considerations**

This section provides an overview of a security analysis of the protocol.

### **Trust Anchors for Timestamp and Other Signature Verification on Archive Retrieval**

TAAs can provide all or some of the trust anchors upon retrieval. These may include all the trust anchors required to verify the various timestamps in the archive record and/or all the trust anchors known to the TAA at the time of the archive submission (i.e., the timestamp on the archived data). The latter set of trust anchors may be useful in digital signature verification on the archived data, if the data was signed.

Trust anchors provided by the TAA upon archive retrieval are transmitted securely since they are included in the signed envelope of the retrieval response. The relying party (i.e., the retrieval client) must use a trust anchor it trusts independent of the trust anchors provided by the TAA to verify the TAA signature on the retrieval response.

The relying party (i.e., the retrieval client) can trust the TAA provided trust anchors or can ignore them. In the latter case, only the TSA (and not the TAA) needs to be trusted for the integrity of the archived data. In other words, the relying party will be able to detect the modifications made to the archived data by the TAA. Refreshing the timestamp on the archived data before the latest (i.e., most current or outermost) timestamp expires ensures this.

### **Algorithm and Technology Advances**

In order to protect against algorithm (i.e., hashing and digital signature) compromise and/or computing technology advances, timestamps are periodically refreshed. For each timestamp token refresh, the

archived data is hashed using a current, secure hashing algorithm and a timestamp token generated using a current, secure digital signature algorithm.

### Security of TSA

TAA's must be able to obtain a trusted timestamp (either by implementing timestamp functionality or by access to a timestamp service). Timestamp-related security considerations apply (see [TSP]).

### ArchiveControls

ArchiveControls are optional request components that request server-side processing in addition to archiving, i.e. collection of certificates and CRLs. ArchiveControls that request alteration of the submitted data should define a response such that the timestamp contained in the archive token can be verified.

## 5. System Description

A trusted archive system using the requirements, concepts and protocol presented here has been developed to successfully demonstrate the concepts presented in this paper.

The components of the system are as follows

- A [TSP]-compliant Time Stamping Authority (TSA)
- A [TAP]-compliant TAA
- [TAP]-compliant TAA clients

The following diagram depicts the overall architecture of the implemented system.

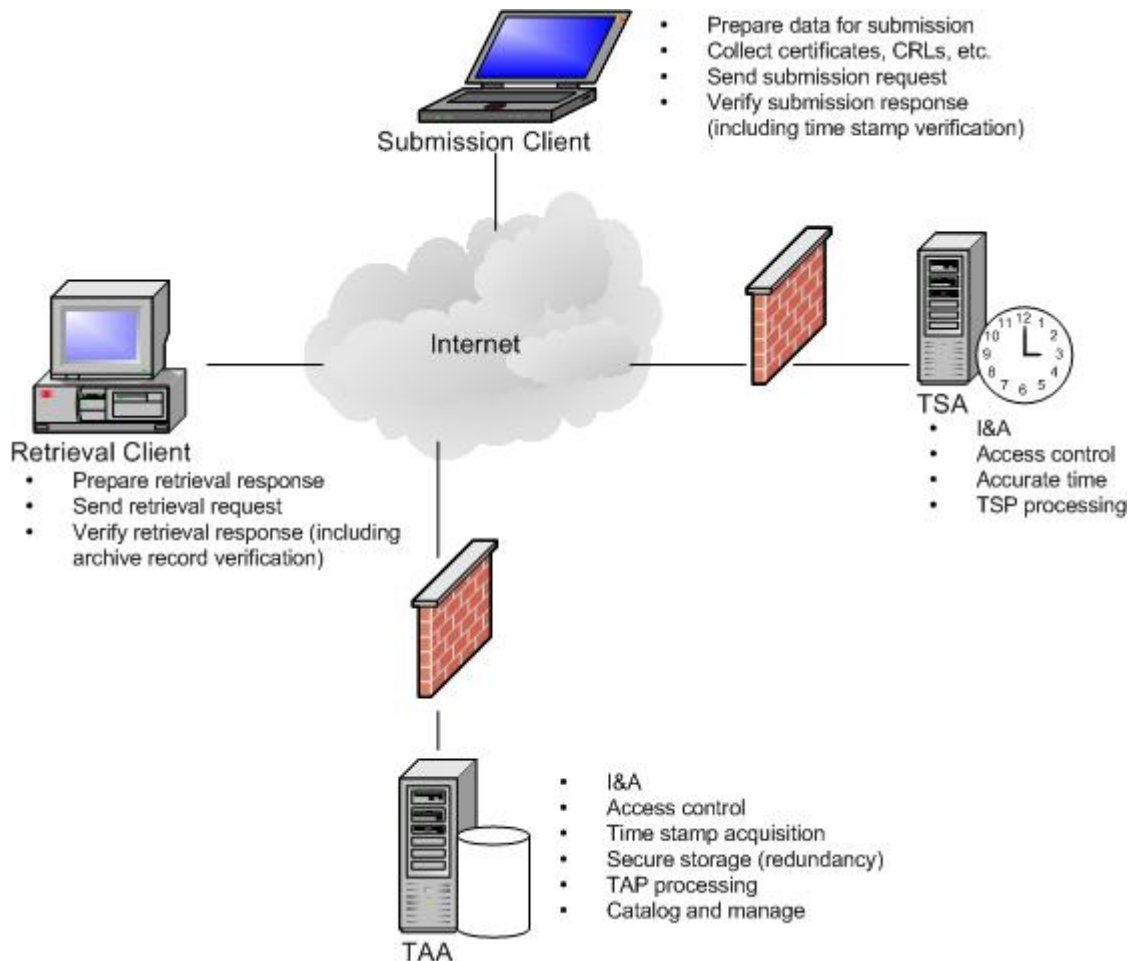


Figure 3 System Architecture

## 5.1 Time Stamp Authority (TSA)

The TSA is RFC 3161[TSP] compliant and is hosted on a PC running Windows 2000 Server. The TSA uses the National Institute of Standards and Technology (NIST) Internet based Network Time Protocol (NTP) service to set the Windows 2000 TSA Server system clock. The TSA interacts with TSA clients using HTTP. The TSA has a public key certificate issued by a PKI recognized by the TAA and, optionally, TAA clients.

## 5.2 Trusted Archive Authority (TAA)

The TAA is TAP I-D compliant and is hosted on a PC running Windows 2000 Server. The TAA has a public key certificate issued by the PKI recognized by the TAA clients. The TAA interacts with TAA clients using HTTP.

The TAA obtains the initial time stamp from the TSA upon submission of archived data. The TAA periodically refreshes the time stamps in accordance with the system concept and the TAP protocol.

The TAA catalogs archive data using the following attributes: submitter DN, time stamp token, submission date and time.

## 5.3 TAA Submission Client

The submission client operates on Windows workstations. The client validates the TAA signature in accordance with TAP and the time stamp token contained in the archive token in accordance with RFC 3161[TSP].

## 5.4 TAA Retrieval Client

The retrieval client operates on Windows workstations. The client provides an archive token to the TAA in order to retrieve the archived data. The client validates the TAA signature in accordance with TAP and the timestamp tokens contained in the archive record in accordance with TAP and RFC 3161[TSP].

A retrieval client unwinds the nested CMS package consisting of multiple nested time stamps. The retrieval client verifies the various time stamps as the CMS package is unwound using the time from the adjacent outer layer as the time of verification. The outermost layer is verified using the current time. The client determines the unwinding is complete when the innermost TSTInfo is reached. The

innermost TSTInfo is used to verify the archived data.

The retrieval client may use trust anchors provided by the TAA during archive record verification or trust anchors available locally.

## 5.5 Operational Considerations

The system described in this section is a proof of concept implementation. If this were an operational system additional security measures are recommended akin to the operations of a CA. It is desirable that the servers and workstation use FIPS 140-2 validated hardware cryptographic modules and Common Criteria validated operating systems and application software. The operational systems and services should use Physical, Procedural, and Personnel (P<sup>3</sup>) security controls commensurate with the security needs and perceived risks.

In addition to the computer security controls described for the TSA and TAA in System Description, appropriate boundary control product (e.g., validated to conform to [FWPP]) should be used to protect the TSA and TAA.

To enhance the security of the trusted archive service using the principle of separation of duties, consideration should be given where one TAA archives the data while another TAA trust anchors relevant to the verification of signatures on the archived data. Timestamps could be obtained from multiple TSAs to limit the damage resulting from TSA compromise. Redundant storage mechanisms should be employed to ensure that no archive data is lost due to device failure or catastrophe.

## 6. Lessons Learned

### 6.1 Metadata

One significant piece of information not included in the TAP protocol was filename and format of the archived data. This information is essential when working with material retrieved from an archive. Future versions of the protocol will include means of including a variety of metadata with an archive submission.

Metadata may also be useful in aggregating archived data over time. For example, to associate a refutation of a document with the original archived document or to associate data related by context, such as a various pieces of data in a criminal file.

## 6.2 Timestamp reliance

The archive record structure defined in TAP relies heavily on timestamps as defined in TSP. This has a number of potentially undesirable properties including:

- A new digital signature as part of the preservation of integrity of a digital signature
- A great degree of trust is invested in the TSA
- A one-to-one ratio of timestamps to documents to archived data objects makes development of high-performance applications difficult

Alternative timestamp structures have been defined that address these concerns by relying on the security of hash algorithms and the availability of published information, see [HOWTO] and [EFF].

## 6.3 Search features are important

TAP featured limited means for searching an archive. The retrieval interface was highly driven by hashes of archived data. While this works well if the data is stored with its archive token, such storage may not be the norm. Without the archive token, the effectiveness of a search is highly correlated with the submission volume of the original submitter. Search features should include means of searching based on content, metadata and/or keywords.

## 6.4 Auto-deletion

[TAP] defined no means for clients to define the period of time a TAA should preserve a data object. This leaves the burden on the submitter to stop the refresh process at some point in time. While this could be negotiated using the policy field, a better solution would be to provide a means for specifying the archivation period at submission time.

This leads to a need to manage the archivation period (or meta-data) post-submission. A better approach to the protocol may have been to specify a submission request, a management request and a single response type. The management request would be used to retrieve and delete archived data as well as to update meta-data, archivation period, etc. A single response format would simplify the handling of errors that are not request-specific ([TAP] defined two response types).

## 7. Future Directions

The LTANS WG has become quite active and is developing three standards in the area of trusted archive:

- Trusted Archive Requirements
- Evidence Record Syntax (based on [ATS])
- Trusted Archive Protocol (based on [TAP])

Another area of research is authentication and authorization for deletion and retrieval. The challenge of authentication and authorization validation in support of long-term non-repudiation can be summarized as follows:

- The identity of authorized parties may change over time. Generally, [TAP] was intended to support claims against data that occur within the memory of a person or institution where retrieval would be performed by the original submitter or by an authorized agent of an organization.
- The definition of authorization attributes may change over time and the naming of attributes may not prove to be unique in contexts that expand over time.
- The authorities such as CA or Attribute Authority (AA) may be no longer in existence.

Data formats, or data format migration, are another area of concern for long term archives. The formats of signed documents today may not be readily usable after a number of years.

Providing confirmation that a specific person generated a data item after a very long period of time is a very difficult problem. Over great periods of time it would be difficult to state with confidence that a particular signature was generated by a particular person. One approach may be to archive the information collected by a CA/TA to establish the binding between a person and a key. The need for this sort of demonstration may be very small. Notarization may be of assistance in this area. Rather than maintain evidence to demonstrate the binding of keys to members of the general population, it may be necessary to simply maintain evidence that binds keys to notaries, who generate attestations at a time when sufficient information is available to confirm the binding of a person to a key and a key to a signature. Biometric information provides another alternative for establishing a link between a specific individual and an archive record and/or an individual's key. This is an area that requires further consideration.

Other mechanisms for providing TAA functionality such as n of m splitting based on Shamir technique [SHA] that would provide a high degree of availability and integrity.

While these problems exist today in general, they are more likely to be encountered when one looks at 20 to 50 years and beyond. Solutions to these issues are a fertile area of research.

## References

[ACTS] NIST Automated Computer Time Service (ACTS), <http://tf.nist.gov/service/acts.htm>

[ATS] Brandner, R., Gondrom, T., Pordesch, U. and M. Tieleman, "Archive Time-Stamps Syntax", draft-brandner-et-al-ats-00.txt, July 2003.

[CMS] Housley, R., "Cryptographic Message Syntax", RFC 3369, August 2002.

[EFF] Bayer, D., Haber, S. and W.S. Stornetta, "Improving the Efficiency and Reliability of Digital Time-Stamping", *Sequences II: Methods in Communication, and Computer Science*, pp.329-334, March 1992.

[FWPP] U.S. Government Firewall Protection Profile for Medium Robustness Environments, Version 1.0, October 28, 2003.

[HOWTO] Haber, S. and W. S. Stornetta, "How to Time-Stamp a Digital Document", *Journal of Cryptology*, Vol. 3, No. 2, pp. 99-111, 1991.

[LTES] Pinkas, D., Ross, J., and N. Pope, "Electronic Signature Formats for long term electronic signatures", RFC 3126, September 2001.

[NTP] Network Time Protocol Specification, Implementation and Analysis, Internet RFC 1305, March 1992.

[OCSP] Myers, M., Ankney, R., Malpani, A., Galperin, S. and C. Adams, "X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP", RFC 2560, June 1999.

[SCVP] Malpani, A., Housley, R. and T. Freeman, "Simple Certificate Validation Protocol (SCVP)", draft-ietf-pkix-scvp-13.txt, October 2003.

[SHA] Shamir, A., "How to Share a Secret", *Communications of the ACM* 24 (1979), n. 11, (1979), 612-613.

[TAP] Chokhani, S. and C. Wallace, "Trusted Archive Protocol", draft-ietf-pkix-tap-00.txt, February 2003.

[TSP] Adams, C., Cain, P., Pinkas, D. and R. Zuccherato, "Internet X.509 Public Key Infrastructure Time-Stamp Protocol (TSP)", RFC 3161, August 2001.