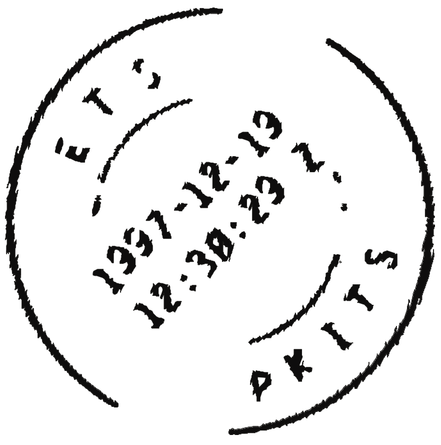


# PKITS
# Public Key Infrastructure with Time-Stamping Authority

**ETS PROJECT: 23.192**

# Deliverable D7a
# Description and Results of the Unstructured Data Time-Stamping Protocol Implementations

**Produced by:** ***UPC***
**Date of issue: 18*th December 1998***
**Revision Number:** ***1***

# TABLE OF CONTENTS

**1. EXECUTIVE SUMMARY .......... 4**

**2. REFERENCES .......... 4**

**3. DESCRIPTION OF THE UNSTRUCTURED DATA TIME-STAMPING TESTBED .......... 5**

3.1. SYSTEM ARCHITECTURE .......... 6
*3.1.1. System Architecture of the PKITS TimeStamping Server .......... 6*
*3.1.2. System Architecture of the PKITS TimeStamping Client .......... 9*
*3.1.3. Graphical Representation of the Service Architecture .......... 12*
3.2. PERFORMANCE .......... 13

**4. APPENDIX A: TIMESTAMP AND DOCUMENT STORAGE .......... 15**

**5. APPENDIX B: OBJECT IDENTIFIERS USED IN THE PROTOTYPE IMPLEMENTATION .......... 15**

**6. APPENDIX C: TIME STAMPING WITH LINKING PROTOCOL ASN.1 DATA STRUCTURES EXAMPLE .......... 16**

6.1. TIMESTAMPING REQUEST PROTOCOL DATA STRUCTURES .......... 16
*6.1.1. TimeStampRequest data structure .......... 16*
*6.1.2. PKCS#7 SignedData structure the requester submits to the PKITS TimeStamping Server . 17*
6.2. TIMESTAMPING RESPONSE PROTOCOL DATA STRUCTURES .......... 23
*6.2.1. PKCS#7 SignedData structure the server sends to the PKITS TimeStamping Client .......... 23*
*6.2.2. TimeStampToken data structure .......... 31*

**7. APPENDIX D: BIBLIOGRAPHY .......... 32**

## HISTORY

| version | date | Author | comment |
|---|---|---|---|
| 1 | 1998-12-18 | Eduard Bel<br>Manel Medina | 1st Version |

## GLOSSARY OF TERMS

| **Specifications:** | |
|---|---|
| SHALL | Essential requirement. A requirement must be fulfilled or a feature implemented wherever this term occurs. The designer is requested, however, to indicate if one or more “shall requirements” would increase the cost or time unreasonably in relation to the total cost or design cost, in which case the specification may have to be revised. |
| SHOULD | Important requirement. Shall be implemented without or with minimum extra cost. Valid reasons in particular circumstances may allow ignoring such requirements. |
| MAY | Optional requirement. From case to case, it should be decided whether implementing it or not, in any case without exceeding the budget planned for the related activity. |

| **Technical:** | |
|---|---|
| ASN.1 | Abstract Syntax Notation, One |
| CA | Certification Authority |
| NTP | Time Network Protocol |
| TS | Time-Stamping |
| TSA | Time-Stamping Authority |
| UTC | Coordinated Universal Time |

## 1. EXECUTIVE SUMMARY

This document describes the activities performed in order to validate and provide performance measurements of the time-stamping protocols and supporting algorithms developed within the PKITS project.

This document details the implementation of the time-stamping protocols for unstructured data prototype and it is organised as follows: Section 3 covers the system architecture and performance results. Appendixes are included to contain details of ASN.1 encoding. Section 4 formalises in ASN.1 syntax notation the way the client of the service shall store a timestamp and the related original unstructured document. On Section 5 we specify the OIDs we used to encode the prototype. Section 6 is an extract of the client software log file. It details the data structures interchanged by the client and the server applications.

## 2. REFERENCES

**[MIL85]** Mills, D.L. “Network Time Protocol (NTP)”. DARPA Network Working Group Report RFC-958, M/A-COM Linkabit, September 1985.

**[MIL89]** Mills, D.L. (09/89), “Network Time Protocol (version 2) - specification and implementation”. DARPA Network Working Group Report RFC-1119, University of Delaware.

**[MIL92]** Mills, D.L. (03/92), “Network Time Protocol (version 3) - Specification, Implementation and Analysis”. Network Working Group Report RFC-1119, University of Delaware.

**[PKITS D3]** Architecture of Time-Stamping Service and Scenarios of Use: Service and Features, Deliverable D3 of ETS Project 23.192 Public Key Infrastructure with Time-Stamping Authority, May, 1998.

**[PKITS D4]** Time-Stamping Service Functional Specification and Protocols for Unstructured Data, Deliverable D4 of ETS Project 23.192 Public Key Infrastructure with Time-Stamping Authority, July, 1998.

**[RFC 2068]** Hypertext Transfer Protocol – HTTP/1.1. R. Fielding et al. January 1997.

**[RFC 2314]** RSA Laboratories, “PKCS #10 – Certification Request Syntax Standard”, version 1.0, Nov. 1993. http://www.rsa.com/rsalabs/pubs/PKCS

**[RFC 2315]** RSA Laboratories, “PKCS #7 – Cryptographic Message Syntax Standard”, version 1.5, Nov. 1993. http://www.rsa.com/rsalabs/pubs/PKCS

**[X.680]** ITU-Rec. X.680/ISO-IEC I.S. 8824-1, "ASN.1: Specification of Basic Notation", 1994.

# 3. DESCRIPTION OF THE UNSTRUCTURED DATA TIME-STAMPING TESTBED

The time-stamping protocols for unstructured data prototype deals with unstructured data. As defined in previous deliverables [PKITS D3] and [PKITS D4], time-stamping protocols for unstructured data covers the provision of a time-stamping service when the time-stamping authority has no knowledge of the internal structure of the documents to be time-stamped. This is, documents are processed as raw data.

Time-stamping protocols for unstructured data were specified on ASN.1 syntax notation on [PKITS D4]. We defined time-stamping protocols for requesting timestamps, timestamp renewal protocols, timestamp verification protocols and TSAs synchronisation protocols. We also distinguished three time-stamping protocol schemes on [PKITS D3] and [PKITS D4]: simple protocol, linking protocol and distributed protocol.

The implementation prototype described in this document supports time-stamping protocols for requesting timestamps and timestamp renovation protocols in simple or linking protocol schemes, and they have been closely implemented as defined in previous deliverables.

## 3.1. SYSTEM ARCHITECTURE

The time-stamping service prototype for unstructured data is composed by two software packages. We have developed a server application that implements a Time Stamping Authority (PKITS TimeStamping Server) and a client application that adds time-stamping functionality to end-users (PKITS TimeStamping Client). Both software packages were developed with Visual C++ 5.0 and J++ASN Programming Language and Toolkit v.1.6085

On the following chapters we describe the main features of the two packages

### 3.1.1. System Architecture of the PKITS TimeStamping Server

The resulting prototype features the following requirements considered during the specification of the PKITS TimeStamping Server application:

#### 3.1.1.1. TimeStamping service infrastructure

We initially considered implement a standalone TSA that supports time-stamping protocols for requesting timestamps and timestamp renovation protocols in simple or linking protocol schemes.

Because of this, distributed protocol scheme nor synchronisation protocols defined on [PKITS D4] were implemented

#### 3.1.1.2. Hash functions, digital signatures and security certificates

As defined on these service protocols, a TSA shall use X509 v.3 certificates to digitally sign the responses sent to the requesters. Both server and client software packages uses X509 v3 certificates issued by the certification authority of the Universitat Politècnica de Catalunya (esCert-UPC) DEMO-CA.

When installing the PKITS TimeStamping Server it submits an on-line PKCS#10 certificate request to the DEMO-CA and this responds with a X509 v.3 certificates encapsulated in a PKCS#7 structure.

The PKITS TimeStamping Server uses RSA algorithm and 512 bits key length to digitally sign all the timestamps and error notifications sent to the requesters.

As defined on previous deliverables, a TSA shall define the supported hash algorithms. The PKITS TimeStamping Server supports MD-2, MD-5 and SHA-1 hash algorithms.

J++ASN Programming Language and Toolkit v.1.6085 is the software used to apply cryptographic operations (digitally signatures, hash computations) and to manage ASN.1 time-stamping protocols.

### 3.1.1.3. Time synchronisation

The PKITS TimeStamping Server periodically synchronises its clock with a time server through NTP (Network Time Protocol). In the prototype scheme, we use the time server of the Universitat Politècnica de Catalunya SIRIUS.AC.UPC.ES.

The Intellisoft TimeSync software is used to synchronise the PKITS TimeStamping Server clock.

The PKITS TimeStamping Server stamps UTC (Coordinated Universal Time) time values on the timestamps with a precision of a second.

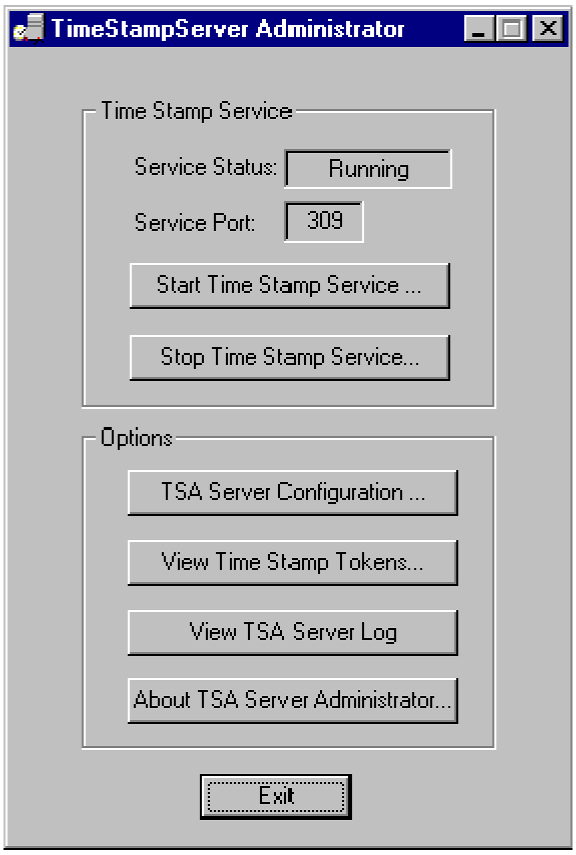
### 3.1.1.4. TimeStamps storage

The PKITS TimeStamping Server stores all the timestamps it issues in order to verify the correctness of the renewal request the users submitted and logs all its activity.

### 3.1.1.5. Supported platforms

PKITS TimeStamping Server was developed on Windows NT 4 Server-Intel Pentium-II platform.

It is fully integrated on Windows NT Server operating system: it runs as a NT service, logs its activity on the NT logs system and an administration application is supplied to configure and to maintain the service (PKITS TimeStamping Server Administrator). An install/uninstall application is also provided to automatically update the affected registry entries of the NT operating system.


TimeStampServer Administrator
Time Stamp Service
Start Time Stamp Service ...
Stop Time Stamp Service...
Options
TSA Server Configuration ...
View Time Stamp Tokens...
View TSA Server Log
About TSA Server Administrator...
Exit

Service Status: Running

Service Port: 309

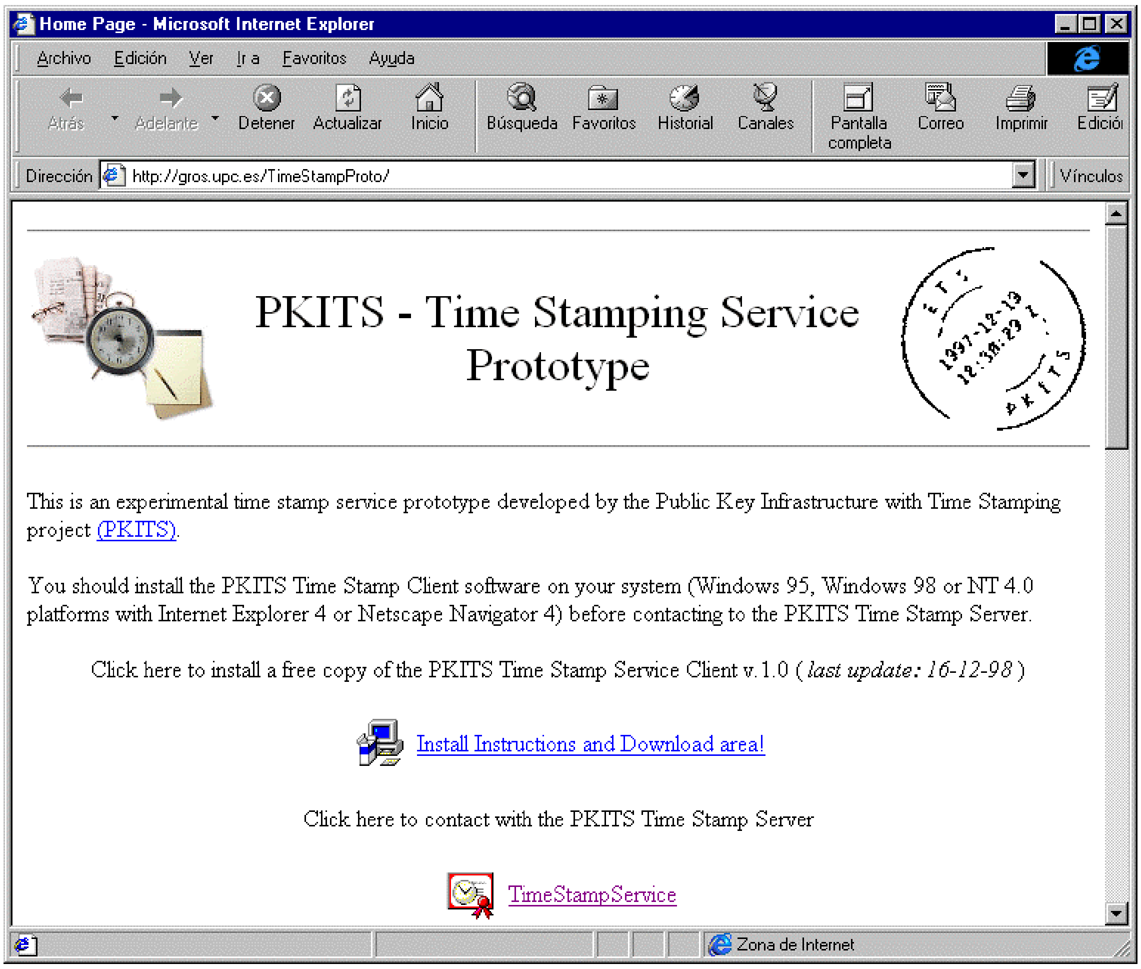
### 3.1.1.6. Communications and service access

PKITS TimeStamping Server uses TCP/IP protocols to interchange time-stamping protocols encapsulated on HTTP messages. It listens user requests on port 309 by default. However, it is also possible to indicate another service port.

In order to provide to the users an easy way to contact with the PKITS TimeStamping Server, the service is accessible from a WEB server. PKITS TimeStamping Service installation program automatically set-ups IIS 3 (Microsoft Internet Information Server 3) to support TimeStamping service. Other WEB servers shall be manually configured.

On this WEB server, an Internet user can retrieve the PKITS TimeStamping Server X509 v.3 certificate and the client software package that shall be used to interact with the service.

The service is currently installed on a Windows NT 4 Server-Intel Pentium platform (GROS.UPC.ES) at the Universitat Politècnica de Catalunya (UPC) with an open-access policy. The time-stamping service is accessible at http://gros.upc.es/TimeStampProtocol.


Home Page - Microsoft Internet Explorer
Archivo Edición Ver Ir a Favoritos Ayuda
Atrás Adelante Detener Actualizar Inicio Búsqueda Favoritos Historial Canales Pantalla completa Correo Imprimir Edición
Dirección http://gros.upc.es/TimeStampProto/
Vínculos
Zona de Internet

PKITS - Time Stamping Service Prototype

This is an experimental time stamp service prototype developed by the Public Key Infrastructure with Time Stamping project (PKITS).

You should install the PKITS Time Stamp Client software on your system (Windows 95, Windows 98 or NT 4.0 platforms with Internet Explorer 4 or Netscape Navigator 4) before contacting to the PKITS Time Stamp Server.

Click here to install a free copy of the PKITS Time Stamp Service Client v.1.0 ( *last update: 16-12-98* )

Install Instructions and Download area!

Click here to contact with the PKITS Time Stamp Server

TimeStampService

## 3.1.2.

The resulting prototype features the following requirements considered during the specification of the PKITS TimeStamping Client application:

**TimeStamping service infrastructure**

and timestamp renovation protocols in simple or linking protocol schemes. When renewing a timestamp, the user shall indicate if he wants to encapsulate the complete timestamp or only a

### 3.1.2.2.

The PKITS TimeStamping client software package uses X509 v3 certificates issued by the certification authority of the Universitat Politècnica de Catalunya (esCert-UPC) DEMO-CA.

When installing the PKITS TimeStamping Client it submits an on-line PKCS#10 certificate request to the DEMO-CA and it responds with a X509 v.3 certificates encapsulated in a PKCS#7 structure.

As the PKITS TimeStamping Server, the PKITS TimeStamping Client uses RSA algorithm and 512 bits key length to digitally sign all the timestamp requests submitted to the time-stamping server

The client package also supports MD-2, MD-5 and SHA-1 hash algorithms. Because of this, the users can encapsulate up to three different message digest of the document to be timestamped in a timestamp request.

J++ASN Programming Language and Toolkit v.1.6085 is also the software used to apply cryptographic operations (digitally signatures, hash computations) and to manage ASN.1 time-stamping protocols.

### 3.1.2.3. TimeStamps storage

When the PKITS TimeStamping Client software receives a valid timestamp from the PKITS TimeStamping Server, it encapsulates together the timestamp and the source document in a PKCS#7 SignedData structure and digitally signs it. This signed structure is then stored in a timestamp storage that can also be managed through the PKITS TimeStamping Client software. Timestamps stored can be renewed, deleted or save it to an external file in order to copy it to a removable device or to be attached in e-mails.

In order to manage timestamps from other users of the service, an auxiliary application PKITS TimeStamp Viewer is also provided to check and display the timestamp encapsulated in this PKCS#7 resulting structure, and also to extract the encapsulated document.

This PKCS#7 Signed Data that encapsulates the timestamp and the source document was not specified in previous deliverable [PKITS D4], and it is specified on appendix A.

### 3.1.2.4. Supported platforms

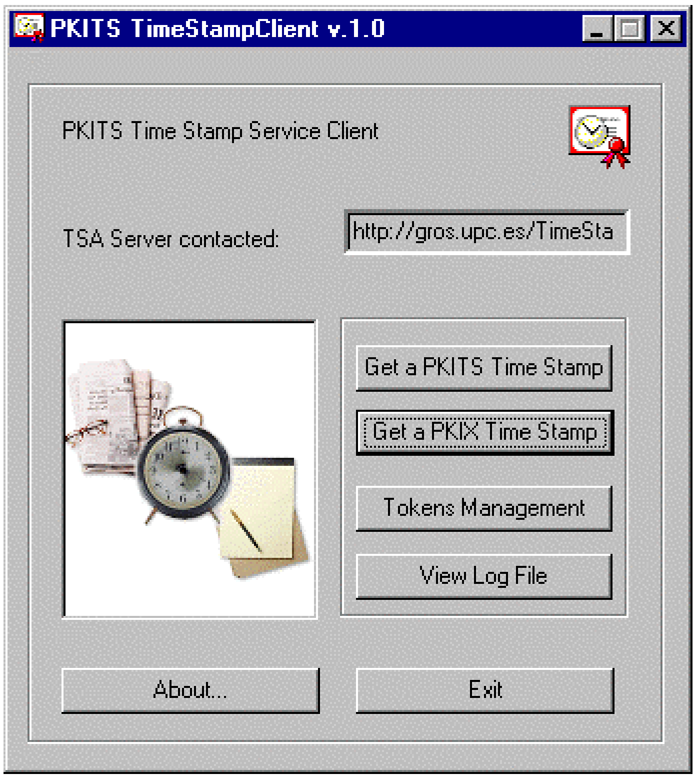
PKITS TimeStamping Client runs on Windows 95/98 and Windows NT 4 platforms. This software adds time-stamping functionality to the Microsoft Internet Explorer 4.0 and Netscape Communicator 4.0x and is fully integrated with them as a helper application.

An install/uninstall application is also provided to automatically update the affected registry entries of the Windows 95/98, NT operating system and Explorer and Communicator browsers.

### 3.1.2.5. Communications and service access

PKITS TimeStamping Client uses TCP/IP protocols to interchange time-stamping protocols encapsulated on HTTP messages.

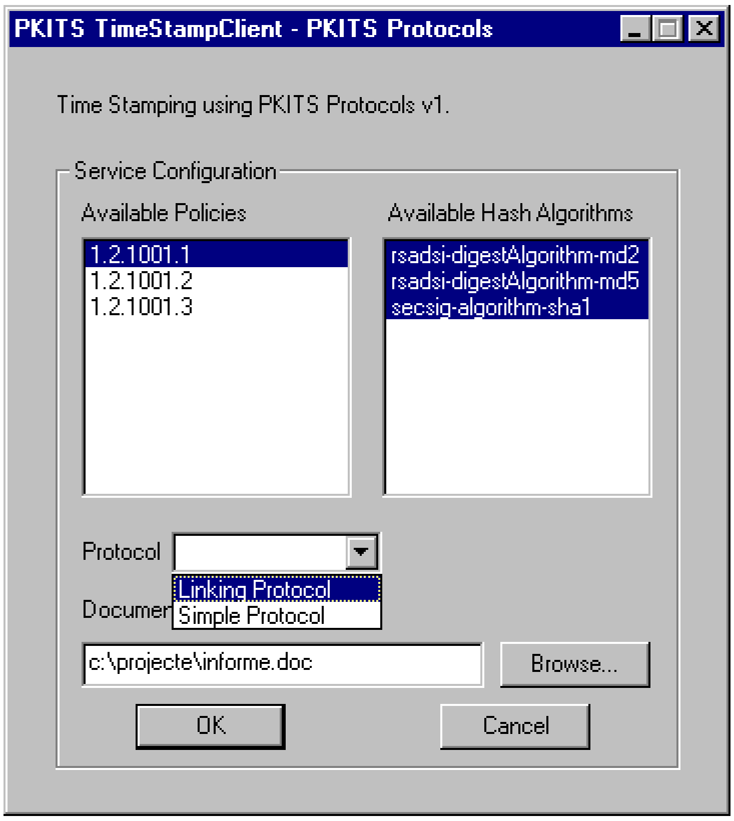
In order to contact to a PKITS TimeStamping Server, the user shall use Internet Explorer 4.0 or Netscape Communicator 4.0x to open the time-stamping service WEB page and follow the "TimeStampService" link. This will start and configure automatically the PKITS TimeStamping Client to operate with the contacted server and the following window will appear:


PKITS TimeStampClient v.1.0

PKITS Time Stamp Service Client

TSA Server contacted: http://gros.upc.es/TimeSta

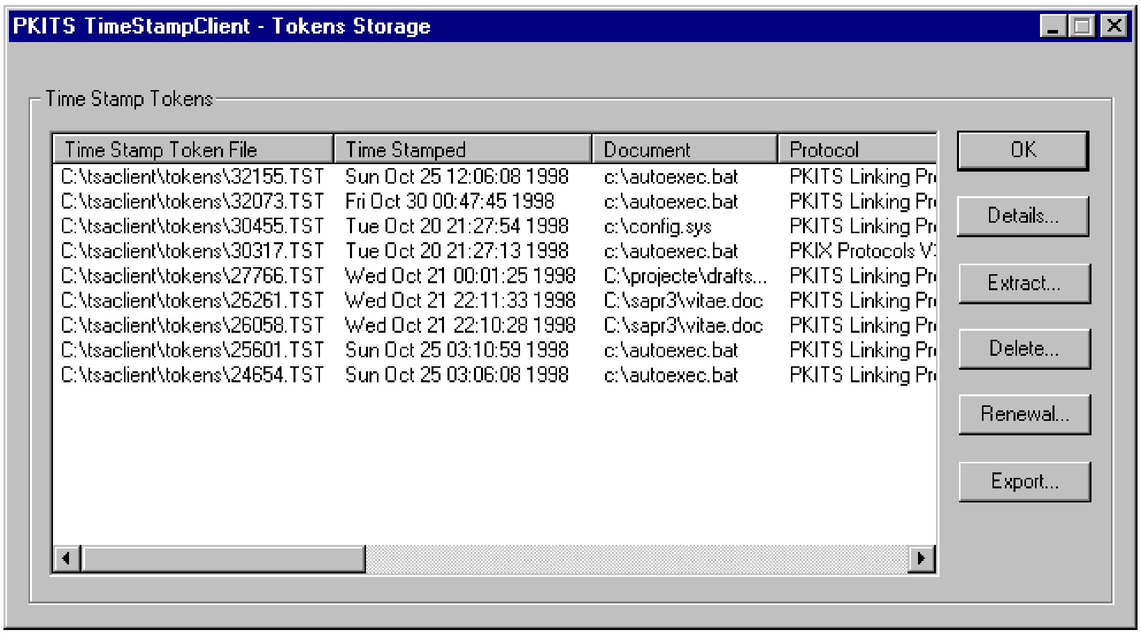
Clicking on the "Get a PKITS TimeStamp" button, the user can introduce the parameters of the timestamp request (hash algorithms and document to be time-stamped) in the following window:



When clicking "OK" button, the operations performed by the PKITS TimeStamping Client are:

1. A message digest is calculated for every hash algorithm the user selected
2. The resulting message digests and the protocol the user indicated are encapsulated in a timestamp request that is digitally signed with the private key of the client software. The signed request is then submitted to the PKITS TimeStamping Server
3. The PKITS TimeStamping Client waits for a response from the server
4. When receiving the response, it checks the digital signature from the PKITS TimeStamping Server and if no errors are encountered the timestamp received is stored in the timestamp client storage.

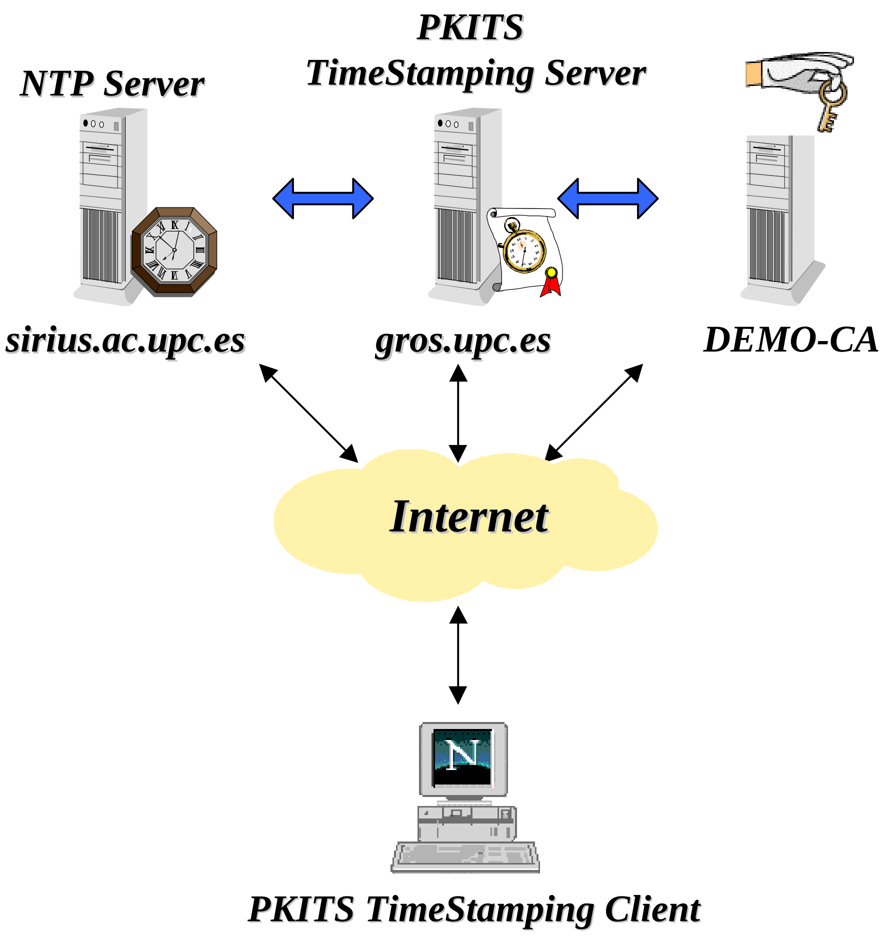
The stored timestamps can be managed clicking the "Tokens Management" button on the main window. The following window will appear:


PKITS TimeStampClient - Tokens Storage

Time Stamp Tokens

| Time Stamp Token File | Time Stamped | Document | Protocol |
|---|---|---|---|
| C:\tsaclient\tokens\32155.TST | Sun Oct 25 12:06:08 1998 | c:\autoexec.bat | PKITS Linking Pr |
| C:\tsaclient\tokens\32073.TST | Fri Oct 30 00:47:45 1998 | c:\autoexec.bat | PKITS Linking Pr |
| C:\tsaclient\tokens\30455.TST | Tue Oct 20 21:27:54 1998 | c:\config.sys | PKITS Linking Pr |
| C:\tsaclient\tokens\30317.TST | Tue Oct 20 21:27:13 1998 | c:\autoexec.bat | PKIX Protocols V |
| C:\tsaclient\tokens\27766.TST | Wed Oct 21 00:01:25 1998 | C:\projecte\drafts... | PKITS Linking Pr |
| C:\tsaclient\tokens\26261.TST | Wed Oct 21 22:11:33 1998 | C:\sapr3\vitae.doc | PKITS Linking Pr |
| C:\tsaclient\tokens\26058.TST | Wed Oct 21 22:10:28 1998 | C:\sapr3\vitae.doc | PKITS Linking Pr |
| C:\tsaclient\tokens\25601.TST | Sun Oct 25 03:10:59 1998 | c:\autoexec.bat | PKITS Linking Pr |
| C:\tsaclient\tokens\24654.TST | Sun Oct 25 03:06:08 1998 | c:\autoexec.bat | PKITS Linking Pr |

## 3.1.3. Graphical Representation of the Service Architecture



## 3.2. PERFORMANCE

The following table resumes a simple performance test applied to the PKITS TimeStamping Server and PKITS TimeStamping Client. The test consists on measuring the time that a complete timestamp transaction (linking protocol scheme) takes.

The involved steps are:

1. Compute a single hash digest from the original document
2. Generate a timestamp request (PKCS#7 SignedData) in Linking protocol scheme, digitally sign it and send it to the PKITS TimeStamping Server
3. Wait for the timestamp
4. Check the PKITS TimeStamping Server digital signature of the received timestamp
5. Check that the original document has not been modified during the data transmission between client and server
6. Store the original document and the related timestamp

We considered three different unstructured documents with the following sizes: 100Kb, 1024 Kb and 5120 Kb and the supported hash algorithms: MD-2, MD-5 and SHA-1

These are the results:

**<u>Test environment</u>**

PKITS TimeStamping Server

- ❑ Running on a Intel Pentium 200 Mhz 128 Mb RAM (GROS.UPC.ES)
- ❑ Windows NT Server 4

PKITS TimeStamping Client

- ❑ Running on a Intel Pentium II 233 Mhz 64 Mb RAM
- ❑ Windows NT Server 4
- ❑ We use a 33.6 Kbits modem to connect to the PKITS TimeStamping Server through a standard telephony Internet provider

**<u>Network Transmission</u>**

The data transfer in bytes between the client and server applications is independent of the original document size, because a message digest is transferred to the server instead of the complete document. Because MD-2 and MD-5 hash algorithms produce a 128 bits message digest and SHA-1 algorithm produces a160 bits message digest, the difference is only about 32 bits when sending the request to the server. When receiving the timestamp in linking protocol scheme, the linking information (Backward Link) affects the total amount of bytes received

| | **Using MD-2 or MD-5 as a hash algorithm** | **Using SHA-1 as a hash algorithm** |
|---|---|---|
| **Data transfer size** | 2279 byte sent<br>5267 byte received (aprox.) | 2284 byte sent<br>5314 bytes received (aprox.) |

## Transaction Time

***100 Kb file***

| | **Total Time** |
|---|---|
| **MD-2 hash algorithm** | *4.01 seconds* |
| **MD-5 hash algorithm** | *3.37 seconds* |
| **SHA-1 hash algorithm** | *3.01 seconds* |

***1024 Kb file***

| | **Total Time** |
|---|---|
| **MD-2 hash algorithm** | *20.56 seconds* |
| **MD-5 hash algorithm** | *4.66 seconds* |
| **SHA-1 hash algorithm** | *4.20 seconds* |

***5120 Kb file***

| | **Total Time** |
|---|---|
| **MD-2 hash algorithm** | *1 minute 33 seconds* |
| **MD-5 hash algorithm** | *12.82 seconds* |
| **SHA-1 hash algorithm** | *11.38 seconds* |

## 4. APPENDIX A: TIMESTAMP AND DOCUMENT STORAGE

In order to relate a timestamp with the original document the message digests encapsulated were computed of, the PKITS TimeStamp Client uses the PKCS#7 SignedData data structure.

This data structure is as follows:

- The ContentInfo data field shall have pkcs7-data type value and it shall encapsulate the original document.

- An authenticated attributed shall also be encapsulated in this PKCS#7 SignedData structure in order to introduce the related timestamp. This TimeStamp authenticated attribute is identified by the following OID:

  ```
  TimeStampId ::= OBJECT IDENTIFIER { to be supplied }
  ```

  The TimeStampId attribute type has the following ASN.1 data structure:

  ```
  TimeStamp ::=ContentInfo
  ```

- The PKITS TimeStamping Client digitally signs the resulting PKCS#7 SignedData structure

The signed structured is encoded (DER codification) into a file with .TST extension filename

## 5. APPENDIX B: OBJECT IDENTIFIERS USED IN THE PROTOTYPE IMPLEMENTATION

On [PKITS D4] we define some OID as OBJECT IDENTIFIER { to be supplied }. In order to codify the prototype, the following values were used:

| **Object Identifiers** | **OID Value** |
|---|---|
| TimeStampRequestId | { pkcs-7 2003 } |
| TimeStampTokenId | { pkcs-7 2004 } |
| TimeStampId | { pkcs-9 30 } |
| EncapsulatedTypeId | { pkcs-9 31 } |
| TimeStampProtocolId | { pkcs-9 32 } |
| TimeStampValidityId | { pkcs-9 34 } |
| TimeStampRequesterId | {pkcs-9 33 } |
| TimeStampSNId | { pkcs-9 35 } |
| TimeStampLinksId | { pkcs-9 38 } |
| TimeStampRenewalId | { pkcs-9 40 } |

Pkcs-7 is the OID {iso (1) member-body (2) US (840) rsadsi (113549) pkcs (1) 7 }
Pkcs-9 is the OID {iso (1) member-body (2) US (840) rsadsi (113549) pkcs (1) 9 }

# 6. APPENDIX C: TIME STAMPING WITH LINKING PROTOCOL ASN.1 DATA STRUCTURES EXAMPLE

This appendix is a commented extract of the log file generated by the PKITS TimeStamping Client in a timestamp request transaction with Linking protocol scheme. A user requested a new timestamp and the server responded with a new one in Linking protocol scheme.

There are four data structures, a pair related to the request the user submitted to the PKITS TimeStamping service and the other pair related to the response the server sent to the requester.

One data structure of each pair is the PKCS#7 SignedData structure that encapsulated the time-stamping protocol data structures and the authenticated attributes of the request the user submits to the server or the response the server sent to the user.

The other two protocol data structures are the TimeStampRequest and TimeStampToken the user submits to the server and the response the server sent to the user respectively.

## 6.1. TIMESTAMPING REQUEST PROTOCOL DATA STRUCTURES

### 6.1.1. TimeStampRequest data structure

```
PKITS Time Stamp Token request TimeStampRequest encapsulated
structure:
SEQUENCE  (81)  {
  version         INTEGER  (1)  0x00   (0)
  policy          SEQUENCE  (37)  {

    The user requests the security policy 1.2.1001.1 (demo value)

    policyIdentifier OBJECT IDENTIFIER  (4)  1.2.1001.1
    policyQualifiers SEQUENCE OF  (29)  {
      SEQUENCE  (27)  {
        policyQualifierId OBJECT IDENTIFIER  (8)  1.3.6.1.5.5.7.2.1
        qualifier         TYPE  (17)  with {

    Ficticious URL where additional security policy information could
    be found

          IA5String  (15)  "www.cps.tsa.com"
        }
      }
    }
  }
  nonce           INTEGER  (1)  0x00   (0)
  messageImprints SEQUENCE OF  (34)  {

    The user selected the MD-2 hash algorithm to compute a message
    digest from the original document

    SEQUENCE  (32)  {
      hashAlgorithm SEQUENCE  (12)  {
        algorithm  OBJECT IDENTIFIER  (8)  rsadsi-digestAlgorithm-md2
        parameters TYPE  (2)  with {
          NULL  (0)
        }
      }
      hashedMessage OCTET STRING  (16)  0094BCFB1A883AB71A0BAD936592
                                        6818
    }
  }
}
```

## 6.1.2. PKCS#7 SignedData structure the requester submits to the PKITS TimeStamping Server

This data structures corresponds to the field Content of the most external PKCS#7 ContentInfo data structure

```
PKITS Time Stamp Token request PKCS#7 SignedData structure:
SEQUENCE  (1151)  {
  version          INTEGER  (1)  0x01   (1)

  The PKITS TimeStamping Client uses the SHA-1 hash algorihtm to
  compute the message digest of the ContentInfo data field (related to
  PKCS#7 SignedData standard definition)

  digestAlgorithms SET OF  (11)  {
    SEQUENCE  (9)  {
      algorithm  OBJECT IDENTIFIER  (5)  secsig-algorithm-sha1
      parameters TYPE  (2)  with {
        NULL  (0)
      }
    }
  }

  contentInfo      SEQUENCE  (98)  {
    contentType OBJECT IDENTIFIER  (9)  pkcs7-data
    content     [0] TYPE  (85)  with {

      DER codification of the TimeStampRequest data structure (the
      previous one in this appendix)

      OCTET STRING (83)305102010030250604 2A876901301D301B0
                       6082B060105050702011 60F7777772E6370
                       732E7473612E636F6D02010030223020300
                       C06082A864886F70D020205 ...
    }
  }
  certificates     [0] IMPLICIT SET OF  (726)  {

  PKITS TimeStamping Client X509 v.3 certificate used to sign this
  PKCS#7 SignedData structure

    SEQUENCE  (722)  {
      toBeSigned        SEQUENCE  (636)  {
        version             [0] INTEGER  (1)  0x02   (2)

        Certificate serial number

        serialNumber        INTEGER  (2)  0x0381   (897)
        signature           SEQUENCE  (13)  {

          algorithm  OBJECT IDENTIFIER  (9)
                              pkcs1-sha1WithRsaSignature
          parameters TYPE  (2)  with {
```

```
    NULL  (0)
  }
}
```

***DEMO-CA is the root CA***

```
issuer               SEQUENCE OF  (86)  RDN {
  OU=DEMOCA-CERT, OU=esCERT, O=UPC, L=Barcelona, C=ES
}
validity             SEQUENCE  (30)  {
  notBefore UTCTime  (13)  "981020192645Z"
  notAfter  UTCTime  (13)  "981220202645Z"
}
```

***The owner of the certificate***

```
subject              SEQUENCE OF  (27)  RDN {
  CN=Eduard Bel Serra
}
```

***Certificate public key and signature algorithm***

```
subjectPublicKeyInfo SEQUENCE  (92)  {
  algorithm        SEQUENCE  (13)  {
    algorithm  OBJECT IDENTIFIER  (9)  pkcs1-rsaEncryption
    parameters TYPE  (2)  with {
      NULL  (0)
    }
  }
  subjectPublicKey BIT STRING  (75)  Encapsulates {
    TYPE  (74)  with {
      rSAPublicKey     SEQUENCE  (72)  {
        modulus  INTEGER  (65) 0x00BB692E402A88A937923B64949C
                               44532EFFBC517755A746A9D04B30BA
                               108A7818B9A6C0AA9D06178847FFA4
                               545E1EEFAE03E87CE4EF8B6EBF1151
                               279B8DFC2AD1
        exponent INTEGER  (3)  0x010001   (65537)
      }
    }
  }
}
issuerUId       [1] IMPLICIT BIT STRING  OPTIONAL NOT PRESENT
subjectUId      [2] IMPLICIT BIT STRING  OPTIONAL NOT PRESENT
```

***Netscape compatible certificate extensions***

```
extensions      [3] SEQUENCE OF  (361)  {
  extension SEQUENCE  (14)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-keyUsage
    critical  BOOLEAN  (1)  TRUE
    extnValue OCTET STRING  (4)  Encapsulates {
      TYPE  (4)  with {
        BIT STRING  (2)  01  F8
      }
    }
  }
  extension SEQUENCE  (12)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-basicConstraints
    critical  BOOLEAN  (1)  TRUE
    extnValue OCTET STRING  (2)  Encapsulates {
      TYPE  (2)  with {
        SEQUENCE  (0)  {
          cA                 BOOLEAN  (1)  FALSE DEFAULT
          pathLenConstraint INTEGER  OPTIONAL NOT PRESENT
        }
      }
    }
  }
  extension SEQUENCE  (31)  {
    extnId    OBJECT IDENTIFIER  (3)
                      id-ce-authorityKeyIdentifier
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (24)  Encapsulates {
      TYPE  (24)  with {
        SEQUENCE  (22)  {
          keyIdentifier       [0] IMPLICIT OCTET STRING  (20)
                                  39687F1F21160CC820B3BDB70F0E
                                  EA7E37553A1A
          authorityCertIssuer [1] IMPLICIT SEQUENCE OF
                                  OPTIONAL NOT PRESENT
          authorityCertSerialNumber [2] IMPLICIT INTEGER
                                  OPTIONAL NOT PRESENT
        }
      }
    }
  }
  extension SEQUENCE  (85)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-issuerAltName
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (78)  Encapsulates {
      TYPE  (78)  with {
        SEQUENCE OF  (76)  {
          generalName CHOICE  (16)   {
            rfc822Name  [1] IMPLICIT IA5String  (14)
                            "info@ca.upc.es"
          }
          generalName CHOICE  (29)   {
            uRI         [6] IMPLICIT IA5String  (27)
                            "http://ca.upc.es/~ca/democa"
          }
          generalName CHOICE  (31)   {
            uRI         [6] IMPLICIT IA5String  (29)
                            "https://ca.upc.es:6443/democa"
          }
        }
      }
    }
  }
  extension SEQUENCE  (52)  {
    extnId    OBJECT IDENTIFIER  (3)
                            id-ce-cRLDistributionPoints
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (45)  Encapsulates {
      TYPE  (45)  with {
        SEQUENCE OF  (43)  {
          SEQUENCE  (41)  {
            distributionPoint [0] CHOICE  (39)   {
              fullName      [0] IMPLICIT SEQUENCE OF  (37)  {
                generalName CHOICE  (37)   {
                  uRI         [6] IMPLICIT IA5String  (35)
                       "http://ca.upc.es/ice-c1/crl/crl.pem"
                  }
                }
              }
              reasons         [1] IMPLICIT BIT STRING OPTIONAL
                                  NOT PRESENT
              cRLIssuer       [2] IMPLICIT SEQUENCE OF OPTIONAL
                                  NOT PRESENT
            }
          }
        }
      }
    }
    extension SEQUENCE  (17)  {
      extnId    OBJECT IDENTIFIER  (9)  netscape-cert-type
      critical  BOOLEAN  (1)  FALSE DEFAULT
      extnValue OCTET STRING  (4)  Encapsulates {
        TYPE  (4)  with {
          BIT STRING  (2)  05  A0
        }
      }
    }
    extension SEQUENCE  (84)  {
      extnId    OBJECT IDENTIFIER  (9)  netscape-comment
      critical  BOOLEAN  (1)  FALSE DEFAULT
      extnValue OCTET STRING  (71)  Encapsulates {
        TYPE  (71)  with {
          IA5String  (69)  "Disclaimer: this is a DEMO CA, no
                              certification policy is guara..."
        }
      }
    }
    extension SEQUENCE  (50)  {
      extnId    OBJECT IDENTIFIER  (9)  netscape-ca-policy-url
      critical  BOOLEAN  (1)  FALSE DEFAULT
      extnValue OCTET STRING  (37)  Encapsulates {
        TYPE  (37)  with {
          IA5String  (35)  "http://ca.upc.es/ice-
                              c1/policy.html"
        }
      }
    }
  }
}
```

***DEMO-CA digitally signs this certificate using RSA algorithm***

```
signatureAlgorithm SEQUENCE  (13)  {
  algorithm  OBJECT IDENTIFIER  (9)  pkcs1-sha1WithRsaSignature
  parameters TYPE  (2)  with {
    NULL  (0)
  }
}
signature          BIT STRING  (65)  0071D468F69B8F1672573
                        AEE277602DBE2F97FFFAFD033E6C4D06F7
                        3475E1310ACF69D44366385B1BD07C1B0D
                        A8B603BF1B27947B1032D618A8F7BD91DF
                        0201618
  }
}
crls             [1] IMPLICIT SET OF  OPTIONAL NOT PRESENT
signerInfos      SET OF  (301)  {
```

***This PKCS#7 is signed by the PKITS TimeStamp Client with its private key.***

```
SEQUENCE  (297)  {
  version                  INTEGER  (1)  0x01   (1)
  issuerAndSerialNumber    SEQUENCE  (92)  {
```

***This issuer and serial Number refers to the certificate encapsulated in the previous certificates data field***

```
    issuer       SEQUENCE OF  (86)  RDN {
      OU=DEMOCA-CERT, OU=esCERT, O=UPC, L=Barcelona, C=ES
    }
    serialNumber INTEGER  (2)  0x0381   (897)
  }
```

***The PKITS TimeStamping Client uses the SHA-1 hash algorihtm to compute the message digest of the ContentInfo data field (related to PKCS#7 SignedData standard definition)***

```
  digestAlgorithm          SEQUENCE  (9)  {
    algorithm  OBJECT IDENTIFIER  (5)  secsig-algorithm-sha1
    parameters TYPE  (2)  with {
      NULL  (0)
    }
  }
```

***Authenticated attributes***

```
  authenticatedAttributes   [0] IMPLICIT SET OF  (79)  {
```

***PKCS#7 standard mandatory attribute. Identifies the data structure encapsulated in the ContentInfo data field***

```
    SEQUENCE  (24)  {
      attrType  OBJECT IDENTIFIER  (9)  pkcs9-contentType
      attrValue TYPE  (13)  with {
        SET OF  (11)  {
          OBJECT IDENTIFIER  (9)  pkcs7-data
        }
      }
    }
```

***PKCS#7 standard mandatory attribute.It encapsulates the message digest of the ContentInfo field computed with the specified algorithm (this is SHA-1)***

```
    SEQUENCE  (35)  {
      attrType  OBJECT IDENTIFIER  (9)  pkcs9-messageDigest
      attrValue TYPE  (24)  with {
        SET OF  (22)  {
          OCTET STRING  (20) 89F9E32898B4D9AE4A8671339726B03C
                             7B2A3EED
        }
      }
    }
```

***PKITS Time-Stamping service authenticated attribute: TimeStampProtocol.Value 2 represents Linking protocol scheme***

```
        SEQUENCE  (14)  {
          attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.32
          attrValue TYPE  (3)  with {
            ENUMERATED  (1)  0x02   (2)
          }
        }
      }
```

***PKITS TimeStamping Client digitally signs with RSA algorithm***

```
      digestEncryptionAlgorithm SEQUENCE  (13)  {
        algorithm  OBJECT IDENTIFIER  (9)  pkcs1-rsaEncryption
        parameters TYPE  (2)  with {
          NULL  (0)
        }
      }
      encryptedDigest           OCTET STRING  (64)8B2BCC0E2CEA1C151
                                    9C4B746488221D4AA5891F42CAC40D1
                                    B22296C44AE9DBC0292D8DE20AAE9FE
                                    A6C38795A9CF94EF1292E60DD1FE70A
                                    652E21112CB79A82F6
```

***Unauthenticated attribute: Encapsulated Type.***
***Value OID {pkcs-7 2003} specifies that the DER codification stored in the Content field corresponds to a TimeStampRequest data structure***

```
      unauthenticatedAttributes [1] IMPLICIT SET OF  (25)  {
        SEQUENCE  (23)  {
          attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.31
          attrValue TYPE  (12)  with {
            OBJECT IDENTIFIER  (10)  1.2.840.113549.1.7.2003
          }
        }
      }
    }
  }
}
```

## 6.2. TIMESTAMPING RESPONSE PROTOCOL DATA STRUCTURES

### 6.2.1. PKCS#7 SignedData structure the server sends to the PKITS TimeStamping Client

```
Receiving response from TSAServer...
PKCS#7 SignedData structure received from the TSAServer:
SEQUENCE  (1637)  {
  version          INTEGER  (1)  0x01   (1)

  The PKITS TimeStamping Server also computes the message imprint of
  the Content data field with the SHA-1 algorithm

  digestAlgorithms SET OF  (11)  {
    SEQUENCE  (9)  {
      algorithm  OBJECT IDENTIFIER  (5)  secsig-algorithm-sha1
      parameters TYPE  (2)  with {
        NULL  (0)
      }
    }
  }

  DER codification of the TimeStampToken data structure encapsulated

  contentInfo      SEQUENCE  (143)  {
    contentType OBJECT IDENTIFIER  (9)  pkcs7-data
    content     [0] TYPE  (129)  with {
      OCTET STRING  (127)  307D020100302506042A876901301D301B06082B
                           06010505070201160F7777772E6370732E747361
                           2E636F6D3019020100801454494D452043455254
                           49464943 ...
    }
  }

  PKITS TimeStamping Server certificate

  certificates     [0] IMPLICIT SET OF  (726)  {
    SEQUENCE  (722)  {
      toBeSigned        SEQUENCE  (636)  {
        version             [0] INTEGER  (1)  0x02   (2)
        serialNumber        INTEGER  (2)  0x037E   (894)
        signature           SEQUENCE  (13)  {
          algorithm  OBJECT IDENTIFIER  (9)
                       pkcs1-sha1WithRsaSignature
          parameters TYPE  (2)  with {
            NULL  (0)
          }
        }

        DEMO-CA is the root CA

        issuer              SEQUENCE OF  (86)  RDN {
          OU=DEMOCA-CERT, OU=esCERT, O=UPC, L=Barcelona, C=ES
        }
        validity            SEQUENCE  (30)  {
          notBefore UTCTime  (13)  "981020134941Z"
          notAfter  UTCTime  (13)  "981220144940Z"
        }
        subject              SEQUENCE OF  (27)  RDN {
          CN=PKITS TSA Server
        }
```

***PKITS TimeStamping Server public key***

```
subjectPublicKeyInfo SEQUENCE  (92)  {
  algorithm       SEQUENCE  (13)  {
    algorithm  OBJECT IDENTIFIER  (9)  pkcs1-rsaEncryption
    parameters TYPE  (2)  with {
      NULL  (0)
    }
  }
  subjectPublicKey BIT STRING  (75)  Encapsulates {
    TYPE  (74)  with {
      rSAPublicKey     SEQUENCE  (72)  {
        modulus  INTEGER  (65)  0x00C85520508ACE73F92988938
                                EEF881B2F3C642DA67AB313F256
                                A1F08E6A02C5D99A1D714126AF6
                                490C47B00DD370AE8516FF72164
                                FBA4ABE7C9E789B1717C9A9F
        exponent INTEGER  (3)  0x010001   (65537)
      }
    }
  }
}
issuerUId            [1] IMPLICIT BIT STRING OPTIONAL
                            NOT PRESENT
subjectUId           [2] IMPLICIT BIT STRING OPTIONAL
                            NOT PRESENT
```

***Netscape compatible certificate extensions***

```
extensions           [3] SEQUENCE OF  (361)  {
  extension SEQUENCE  (14)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-keyUsage
    critical  BOOLEAN  (1)  TRUE
    extnValue OCTET STRING  (4)  Encapsulates {
      TYPE  (4)  with {
        BIT STRING  (2)  01  F8
      }
    }
  }
  extension SEQUENCE  (12)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-basicConstraints
    critical  BOOLEAN  (1)  TRUE
    extnValue OCTET STRING  (2)  Encapsulates {
      TYPE  (2)  with {
        SEQUENCE  (0)  {
          cA                BOOLEAN  (1)  FALSE DEFAULT
          pathLenConstraint INTEGER  OPTIONAL NOT PRESENT
        }
      }
    }
  }
  extension SEQUENCE  (31)  {
    extnId    OBJECT IDENTIFIER  (3)
                id-ce-authorityKeyIdentifier
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (24)  Encapsulates {
      TYPE  (24)  with {
        SEQUENCE  (22)  {
          keyIdentifier     [0] IMPLICIT OCTET STRING  (20)
                                 39687F1F21160CC820B3BDB70F0E
                                 EA7E37553A1A
          authorityCertIssuer       [1] IMPLICIT SEQUENCE OF
                                        OPTIONAL NOT PRESENT
          authorityCertSerialNumber [2] IMPLICIT INTEGER
                                        OPTIONAL NOT PRESENT
        }
      }
    }
  }
  extension SEQUENCE  (85)  {
    extnId    OBJECT IDENTIFIER  (3)  id-ce-issuerAltName
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (78)  Encapsulates {
      TYPE  (78)  with {
        SEQUENCE OF  (76)  {
          generalName CHOICE  (16)   {
            rfc822Name  [1] IMPLICIT IA5String  (14)
                            "info@ca.upc.es"
          }
          generalName CHOICE  (29)   {
            uRI         [6] IMPLICIT IA5String  (27)
                            "http://ca.upc.es/~ca/democa"
          }
          generalName CHOICE  (31)   {
            uRI         [6] IMPLICIT IA5String  (29)
                            "https://ca.upc.es:6443/democa"
          }
        }
      }
    }
  }
  extension SEQUENCE  (52)  {
    extnId    OBJECT IDENTIFIER  (3)
                    id-ce-cRLDistributionPoints
    critical  BOOLEAN  (1)  FALSE DEFAULT
    extnValue OCTET STRING  (45)  Encapsulates {
      TYPE  (45)  with {
        SEQUENCE OF  (43)  {
          SEQUENCE  (41)  {
            distributionPoint [0] CHOICE  (39)   {
              fullName      [0] IMPLICIT SEQUENCE OF  (37)  {
                generalName CHOICE  (37)   {
                  uRI         [6] IMPLICIT IA5String  (35)
                      "http://ca.upc.es/ice-c1/crl/crl.pem"
                }
              }
            }
            reasons           [1] IMPLICIT BIT STRING
                                  OPTIONAL NOT PRESENT
            cRLIssuer         [2] IMPLICIT SEQUENCE OF
                                  OPTIONAL NOT PRESENT
          }
        }
      }
    }
  }
  extension SEQUENCE  (17)  {
          extnId    OBJECT IDENTIFIER  (9)  netscape-cert-type
          critical  BOOLEAN  (1)  FALSE DEFAULT
          extnValue OCTET STRING  (4)  Encapsulates {
            TYPE  (4)  with {
              BIT STRING  (2)  06  40
            }
          }
        }
        extension SEQUENCE  (84)  {
          extnId    OBJECT IDENTIFIER  (9)  netscape-comment
          critical  BOOLEAN  (1)  FALSE DEFAULT
          extnValue OCTET STRING  (71)  Encapsulates {
            TYPE  (71)  with {
              IA5String  (69)  "Disclaimer: this is a DEMO CA, no
                           certification policy is guara..."
            }
          }
        }
        extension SEQUENCE  (50)  {
          extnId    OBJECT IDENTIFIER  (9)  netscape-ca-policy-url
          critical  BOOLEAN  (1)  FALSE DEFAULT
          extnValue OCTET STRING  (37)  Encapsulates {
            TYPE  (37)  with {
              IA5String  (35)  "http://ca.upc.es/ice-
                                 c1/policy.html"
            }
          }
        }
      }
    }
```

***DEMO-CA digitally signed this certificate***

```
    signatureAlgorithm SEQUENCE  (13)  {
      algorithm  OBJECT IDENTIFIER  (9)  pkcs1-sha1WithRsaSignature
      parameters TYPE  (2)  with {
        NULL  (0)
      }
    }
    signature          BIT STRING  (65) 00BE73A289F3A47AADD130182
                                        4DBCD3D4DCC2AE317067FA34E
                                        CFDE1A5CDF1A9F67DDC176C18
                                        2C2872C395E5098EF08DC3806
                                        78C7711019158C66B1FB72154
                                        850D
  }
}
crls             [1] IMPLICIT SET OF  OPTIONAL NOT PRESENT
signerInfos      SET OF  (741)  {
```

***The PKITS TimeStamping Server digitally signs this PKCS#7 SignedData***

```
  SEQUENCE  (737)  {
    version                  INTEGER  (1)  0x01   (1)
```

***This issuer and serialNumber refers to the certificate encapsulated in the Certificates field***

```
    issuerAndSerialNumber     SEQUENCE  (92)  {
      issuer       SEQUENCE OF  (86)  RDN {
```

```
    OU=DEMOCA-CERT, OU=esCERT, O=UPC, L=Barcelona, C=ES
  }
  serialNumber INTEGER  (2)  0x037E   (894)
}

digestAlgorithm           SEQUENCE  (9)  {
  algorithm  OBJECT IDENTIFIER  (5)  secsig-algorithm-sha1
  parameters TYPE  (2)  with {
    NULL  (0)
  }
}
```

***Authenticated atributes***

```
authenticatedAttributes   [0] IMPLICIT SET OF  (517)  {
```

***PKITS TimeStamping service authenticated attribute: TimeStampProtocol.Value 2 represents Linking protocol scheme***

```
  SEQUENCE  (14)  {
    attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.32
    attrValue TYPE  (3)  with {
      ENUMERATED  (1)  0x02   (2)
    }
  }
```

***PKITS TimeStamping service authenticated attribute: TimeStampSN. This TimeStamp has serial number 17***

```
  SEQUENCE  (14)  {
    attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.35
    attrValue TYPE  (3)  with {
      INTEGER  (1)  0x11   (17)
    }
  }
```

***PKCS#7 standard mandatory attribute. Identifies the data structure encapsulated in the ContentInfo data field***

```
  SEQUENCE  (24)  {
    attrType  OBJECT IDENTIFIER  (9)  pkcs9-contentType
    attrValue TYPE  (13)  with {
      SET OF  (11)  {
        OBJECT IDENTIFIER  (9)  pkcs7-data
      }
    }
  }
```

***PKITS TimeStamping service authenticated attribute: TimeStampValidity.This timestamp not valid after 25th October 2000***

```
  SEQUENCE  (28)  {
    attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.34
    attrValue TYPE  (17)  with {
      CHOICE  (17)   {
        GeneralizedTime  (15)  "20001025011059Z"
      }
    }
  }
```

***PKCS#7 standard mandatory attribute.It encapsulates the message digest of the ContentInfo field computed with the specified algorithm (this is SHA-1)***

```
SEQUENCE  (35)  {
  attrType  OBJECT IDENTIFIER  (9)  pkcs9-messageDigest
  attrValue TYPE  (24)  with {
    SET OF  (22)  {
      OCTET STRING  (20)  25A430E5CA96ED902BB1CF1EECBC
                          A582CE923832
    }
  }
}
```

***PKITS TimeStamping service authenticated attribute: TimeStampRequester.The timestamp requester was the user with the certificate with serial number 897***

```
SEQUENCE  (105)  {
  attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.33
  attrValue TYPE  (94)  with {
    SEQUENCE  (92)  {
      issuer       SEQUENCE OF  (86)  RDN {
        OU=DEMOCA-CERT, OU=esCERT, O=UPC, L=Barcelona, C=ES
      }
      serialNumber INTEGER  (2)  0x0381   (897)
    }
  }
}
```

***PKITS TimeStamping service authenticated attribute: TimeStampLink.***

```
SEQUENCE  (281)  {
  attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.38
  attrValue TYPE  (270)  with {
    SEQUENCE  (266)  {
      version      INTEGER  (1)  0x00   (0)
```

***BackwardLink: The previous timestamp issued by the PKITS TimeStamping Server has serial number 16, the requester was the user with a certificate with serial number 898 and it was issued on 25th October 1998 at 02:06:08 UTC***

```
      backwardLink SEQUENCE  (255)  {
        serialNumber    INTEGER  (1)  0x10   (16)
        requester       SEQUENCE  (92)  {
          issuer       SEQUENCE OF  (86)  RDN {
            OU=DEMOCA-CERT,OU=esCERT,O=UPC,L=Barcelona,C=ES
          }
          serialNumber INTEGER  (2)  0x0382   (898)
        }
        stampedTime     GeneralizedTime  (15)
                            "19981025020608Z"
```

***Hashs encapsulated in the previous TimeStamp***

```
        messageImprints SEQUENCE OF  (34)  {
          SEQUENCE  (32)  {
            hashAlgorithm SEQUENCE  (12)  {
              algorithm  OBJECT IDENTIFIER  (8)
```

```
                          rsadsi-digestAlgorithm-md5
              parameters TYPE  (2)  with {
                NULL  (0)
              }
            }
            hashedMessage OCTET STRING  (16)
                          4075AEBFE5D6A7130438C9399287E89F
          }
        }

        previous        SEQUENCE OF  (103)  {
          SEQUENCE  (32)  {
            hashAlgorithm SEQUENCE  (12)  {
              algorithm  OBJECT IDENTIFIER  (8)
                          rsadsi-digestAlgorithm-md2
              parameters TYPE  (2)  with {
                NULL  (0)
              }
            }
            hashedMessage OCTET STRING  (16)
                          5EE8CBD875C34E3D0FEDF2250C4CD0EE
          }
          SEQUENCE  (32)  {
            hashAlgorithm SEQUENCE  (12)  {
              algorithm  OBJECT IDENTIFIER  (8)
                          rsadsi-digestAlgorithm-md5
              parameters TYPE  (2)  with {
                NULL  (0)
              }
            }
            hashedMessage OCTET STRING  (16)
                          319CBE896BB009FAAC053E46E405715C
          }
          SEQUENCE  (33)  {
            hashAlgorithm SEQUENCE  (9)  {
              algorithm  OBJECT IDENTIFIER  (5)
                          secsig-algorithm-sha1
              parameters TYPE  (2)  with {
                NULL  (0)
              }
            }
            hashedMessage OCTET STRING  (20)
                59FD72CE86CE212E1E11EE1E44F305089A42F219
          }
        }
      }
```

***Forward Link: serial number of the following timestamp issued by the PKITS TimeStamping Server***

```
      forwardLink  SEQUENCE  (3)  {
        serialNumber INTEGER  (1)  0x12   (18)
      }
    }
  }
 }
}
```

***The PKITS TimeStamping Server digitally signs this PKCS#7 SignedData with the RSA algorithm***

```
      digestEncryptionAlgorithm SEQUENCE  (13)  {
        algorithm  OBJECT IDENTIFIER  (9)  pkcs1-rsaEncryption
        parameters TYPE  (2)  with {
          NULL  (0)
        }
      }
      encryptedDigest           OCTET STRING  (64)0862587BC5D
                                85A18262F8F41B629B6ED0F16CC45
                                E9EA4DD1A27999C35D3A2E5BD0F9C
                                0BD6D4A44BBD088EF02C2F3EFA502
                                CAA2C6592A2BBF2402571747EF15A
```

***Unauthenticated attribute: Encapsulated Type.***
***Value OID {pkcs-7 2004} specifies that the DER codification stored in the Content field corresponds to a TimeStampToken data structure***

```
      unauthenticatedAttributes [1] IMPLICIT SET OF  (25)  {
        SEQUENCE  (23)  {
          attrType  OBJECT IDENTIFIER  (9)  1.2.840.113549.1.9.31
          attrValue TYPE  (12)  with {
            OBJECT IDENTIFIER  (10)  1.2.840.113549.1.7.2004
          }
        }
      }
    }
  }
}
```

## 6.2.2. TimeStampToken data structure

The TimeStampToken encapsulated in its DER codification in the previous PKCS#7 Signed Data in the ContentInfo data field is as follows:

```
TimeStampToken TimeStampToken encapsulated structure:
SEQUENCE  (125)  {
  version         INTEGER  (1)  0x00   (0)

  Security policy the user requested (demo value) in the
  TimeStampRequest data structure

  policy          SEQUENCE  (37)  {
    policyIdentifier OBJECT IDENTIFIER  (4)  1.2.1001.1
    policyQualifiers SEQUENCE OF  (29)  {
      SEQUENCE  (27)  {
        policyQualifierId OBJECT IDENTIFIER  (8)  1.3.6.1.5.5.7.2.1
        qualifier         TYPE  (17)  with {
          IA5String  (15)  "www.cps.tsa.com"
        }
      }
    }
  }

  TimeStamp status

  status          SEQUENCE  (25)  {
    status       INTEGER  (1)  0x00   (0)
    statusString CHOICE  (22)   {
      iA5String    [0] IMPLICIT IA5String  (20)"TIME CERTIFICATE OK!"
    }
    failInfo     BIT STRING  OPTIONAL NOT PRESENT
  }

  Stamped Time in UTC format: 25th October 1998 at 02:10:59.

  stampedTime     GeneralizedTime  (15)  "19981025021059Z"
  nonce           INTEGER  (1)  0x00   (0)

  Message digest the user submited to the PKITS TimeStamping Server

  messageImprints SEQUENCE OF  (34)  {
    SEQUENCE  (32)  {
      hashAlgorithm SEQUENCE  (12)  {
        algorithm  OBJECT IDENTIFIER  (8)  rsadsi-digestAlgorithm-md2
        parameters TYPE  (2)  with {
          NULL  (0)
        }
      }
      hashedMessage OCTET STRING (16) 0094BCFB1A883AB71A0BAD9365
                                      926818
    }
 }
```

# 7. APPENDIX D: BIBLIOGRAPHY

**[ISO 8824]** ITU-Rec. X.680/ISO-IEC I.S. 8824-1, "ASN.1: Specification of Basic Notation", 1994.

**[J++ASN 1]** J++ASN Programming Language and Toolkit: ASN.1 Class Reference. © Francisco Jordan 96/11/15

**[J++ASN 2]** J++ASN Programming Language and Toolkit: Changes and Extensions to J++ASN v09. © Francisco Jordan 97/05/19

**[J++ASN 3]** J++ASN Programming Language and Toolkit: Programmer's Guide. © Francisco Jordan 96/11/15

**[J++ASN 4]** J++ASN Programming Language and Toolkit: Secure Class Reference. © Francisco Jordan 96/11/15

**[J++ASN 5]** J++ASN Programming Language and Toolkit: Error handling Reference. © Francisco Jordan 96/11/15

**[J++ ASN 6]** J++ASN Programming Language and Toolkit: Input/Output Class Reference.
© Francisco Jordan 96/11/15

**[PKIX]** PKIX Working Group, "Internet Public Key Infrastructure - Part IX.509

http://www.ietf.org/html.charters/pkix-charter.html

**[PKITS D3]** Architecture of Time-Stamping Service and Scenarios of Use: Service and Features, Deliverable D3 of ETS Project 23.192 Public Key Infrastructure with Time-Stamping Authority, May, 1998.

**[PKITS D4]** Time-Stamping Service Functional Specification and Protocols for Unstructured Data, Deliverable D4 of ETS Project 23.192 Public Key Infrastructure with Time-Stamping Authority, July, 1998.

**[RFC 1305]** D.L. Mills, "1305 Network Time Protocol (Version 3) Specification,

ftp://ds.internic.net/rfc/rfc1305.txt

**[RFC 1319]** B. Kaliski, "The MD2 Message-Digest Algorithm", Apr. 1992. ftp://ds.internic.net/rfc/rfc1319.txt

**[RFC 1321]** R. Rivest, "The MD5 Message-Digest Algorithm", Apr. 1992. ftp://ds.internic.net/rfc/rfc1321.txt

**[RFC 2068]** Hypertext Transfer Protocol – HTTP/1.1. R. Fielding et al. January 1997.

**[RFC 2314]** RSA Laboratories, "PKCS #10 – Certification Request Syntax Standard", version 1.0, Nov. 1993.
http://www.rsa.com/rsalabs/pubs/PKCS

**[RFC 2315]** RSA Laboratories, "PKCS #7 – Cryptographic Message Syntax Standard", version 1.5, Nov. 1993.
http://www.rsa.com/rsalabs/pubs/PKCS

**[RFC 781]** Su, Z. A specification of the Internet protocol (IP) timestamp option. DARPA Network Working Group Report RFC-781. SRI International, May 1981.
ftp://ds.internic.net/rfc/rfc781.txt

**[X.680]** ITU-Rec. X.680/ISO-IEC I.S. 8824-1, "ASN.1: Specification of Basic Notation", 1994.