INTERNATIONAL STANDARD

ISO/IEC 18014-3

First edition
2004-02-15

Information technology — Security techniques — Time-stamping services —

Part 3: Mechanisms producing linked tokens

Technologies de l'information — Techniques de sécurité — Services d'horodatage —

Partie 3: Mécanismes produisant des jetons liés

Reference number
ISO/IEC 18014-3:2004(E)

© ISO/IEC 2004

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2004

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

© ISO/IEC 2004 – All rights reserved

Contents

Page

Foreword ... iv

Introduction ... v

1 Scope ... 1

2 Normative references ... 1

3 Terms and definitions ... 1

4 General discussion ... 3

5 Operations ... 3
5.1 Linking ... 3
5.2 Aggregation ... 3
5.3 Publishing ... 3

6 Data structures ... 4
6.1 Object identifiers ... 4
6.2 Node ... 4
6.3 Link ... 4
6.4 Chain ... 5
6.5 BindingInfo ... 5
6.6 Extensions ... 6
6.6.1 Request and Token extensions ... 7
6.6.2 BindingInfo extensions ... 8

7 Time-stamp request ... 9

8 Time-stamp response ... 9
8.1 DigestedData encapsulation ... 10
8.2 SignedData encapsulation ... 11

9 Time-stamp verification ... 12

Annex A (normative) **ASN.1 Module for time-stamping** ... 13

Annex B (normative) **Additional Discussion** ... 20
B.1 Discussion ... 20
B.2 Linking ... 20
B.2.1 Linear chain linking ... 20
B.2.2 Anti-monotone binary linking ... 21
B.2.3 Threaded tree linking ... 21
B.3 Aggregation ... 21
B.3.1 Unity aggregation ... 21
B.3.2 Merkle tree aggregation ... 22
B.3.3 One-way accumulator aggregation ... 22
B.4 Publishing ... 23
B.4.1 No publishing ... 23
B.4.2 Single link publishing ... 23
B.4.3 Merkle tree publishing ... 23

Annex C (informative) **Data structures** ... 24
C.1 Time-stamps using DigestedData encapsulation ... 24
C.2 Time-stamps using SignedData encapsulation ... 25
C.3 Linear Chain Linking with Merkle Tree Aggregation ... 26

Bibliography ... 28

© ISO/IEC 2004 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO/IEC 18014-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 18014 consists of the following parts, under the general title *Information technology — Security techniques — Time-stamping services*:

— *Part 1: Framework*

— *Part 2: Mechanisms producing independent tokens*

— *Part 3: Mechanisms producing linked tokens*

Further parts may follow.

--`,```-`-`,,`,,`,`,,`---

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Introduction

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this International Standard may involve the use of patents.

The ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured the ISO and IEC that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with the ISO and IEC. Information may be obtained from:

ISO/IEC JTC 1/SC 27 Standing Document 8 (SD 8) "Patent Information"

SD 8 is publicly available at: http://www.ni.din.de/sc27

Attention is drawn to the possibility that some of the elements of this International Standard may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Information technology — Security techniques — Time-stamping services —

Part 3: Mechanisms producing linked tokens

1 Scope

This part of ISO/IEC 18014

- describes a general model for time-stamping services producing linked tokens;
- describes the basic components used to construct a time-stamping service of this type;
- defines the data structures used to interact with a time-stamping service of this type;
- describes specific instances of such time-stamping services.

2 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 9594-8:2001, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks*

ISO/IEC 10118 (all parts), *Information technology — Security techniques — Hash-functions*

ISO/IEC 18014-1:2002, *Information technology — Security techniques — Time-stamping services — Part 1: Framework*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
aggregation
the process of generating a proxy data item for a group of data items that are linked together at once, producing a verifiable cryptographic link between each data item and the rest of the group.

© ISO/IEC 2004 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

3.2
chain
a verifiable cryptographic binding between a data item and the rest of the data items participating in an aggregation. A chain consists of a sequence of operations that authenticate the participation of the selected data item in the aggregation.

3.3
collision-resistant hash-function
a hash-function satisfying the following property: it is computationally infeasible to find any two distinct inputs which map to the same output. [ISO/IEC 9796-3: 2000]

3.4
data items' representation
a data item or its respective hash value. [ISO/IEC 18014-1: 2002]

3.5
hash-function
a function which maps strings of bits to fixed-length strings of bits, satisfying the following two properties: it is computationally infeasible to find for a given output, an input which maps to this output; it is computationally infeasible to find for a given input, a second input which maps to the same output. [ISO/IEC 9796-3: 2000]

3.6
hash value
the string of bits which is the output of a hash-function. [ISO/IEC 9796-3: 2000]

3.7
link
a data item providing an irreversible and collision-resistant cryptographic statement attesting to the existence of at least two other data items.

3.8
node
a data item representing input data or a previously computed link.

3.9
time-stamp token
a data structure containing a verifiable cryptographic binding between a data items' representation and a time-value. A time-stamp token may also include additional data items in the binding.

3.10
time-stamping authority (TSA)
a trusted third party trusted to provide a time-stamping service. [ISO/IEC 18014-1: 2002]

3.11
time-stamping service
a service providing evidence that a data item existed before a certain point in time. [ISO/IEC 18014-1: 2002]

3.12
trusted third party (TTP)
a security authority, or its agent, trusted by other entities with respect to security related activities. [ISO/IEC 11770-3: 1999]

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

4 General discussion

This part of ISO/IEC 18014 describes methods and processes used to form time-stamp tokens that are related, or “linked”, to other tokens produced by those methods and processes. A time stamping authority (TSA) utilises these methods and processes to provide a secure, verifiable cryptographic binding between a certain point in time and data values, and to enhance the security of the resulting tokens by reducing the trust level required in the TSA. The trustworthiness of the tokens computed in this scheme depends on the integrity of the repository of previously linked tokens maintained by the TSA. The integrity of the repository and of the linking operations performed by the TSA can be verified algorithmically, and do not depend on the trustworthiness of a PKI, nor require trusting that a particular private key has not been compromised.

This part of ISO/IEC 18014 describes how time-stamping systems utilising linked tokens can interoperate through standard tokens, operations, and data formats.

Note: For a complete treatment of time-stamping with linked tokens, see [HS91], [BHS93], [HS97], [BLLV98], and [BLS00]. For a proof of security of the linking technique, see [HS97] Section 3.1, Theorem 1.

5 Operations

In general, a TSA producing linked tokens uses the cryptographic properties of hash-functions to “link” a time-stamp token to other tokens previously generated by the TSA. Hash-functions defined in ISO/IEC 10118 that are both irreversible and collision-resistant are suitable for use in forming linked tokens: irreversibility conceals the content of the original data item from the TSA, while collision resistance ensures a false token cannot be inserted into an existing collection of linked tokens.

5.1 Linking

The linking operation consists of forming a verifiable, irreversible and collision-resistant binding between the time-stamp token and links previously produced by the TSA. The most recently produced such link provides a cryptographic summary of all time-stamp tokens that ever participated in the linking process. A common algorithm for computing a link between two events is to concatenate the data representation of the events and use the resulting concatenation as input to a hash-function. The resulting hash value provides a verifiable cryptographic link between the two input events due to the collision-resistant nature of the hash-function.

5.2 Aggregation

A conformant TSA may perform linking operations on groups of time-stamp tokens, rather than for each time-stamp token individually, in order to improve computational efficiency or achieve a higher degree of service scalability. Linking multiple events together at once is referred to as "aggregation,” and all time-stamp tokens involved in an aggregation shall be assigned the same time-value. Each aggregation scheme takes a group of events as inputs and produces a verifiable cryptographic link between each event and the rest of the group. The unique resulting value associated with the aggregation is further linked by the TSA, in a way similar to the case of the single time-stamp token.

Note: Common forms of aggregation include Merkle hash trees [M80], one-way accumulators [BD93], or other binary linking schemes [BLLV98].

5.3 Publishing

The links produced by a conformant TSA are published in such a way as to make them “widely-witnessed.” The manner most commonly used is through periodic publication on some well-known media, e.g., web pages or offline media. The value published should depend on all tokens generated by the TSA since the previous publishing event. By linking the tokens to "widely-witnessed" events, the TSA in effect generates verifiable statements of when each token was generated by the system.

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

6 Data structures

A conformant TSA produces linked time-stamp tokens using linking and aggregation processes that utilise the data types defined in the following sub-clauses.

6.1 Object identifiers

The following root object identifier arc is defined to support allocation of further object identifiers within this standard:

```
tsp-lt ::= OBJECT IDENTIFIER { iso(1) standard(0) time-stamp(18014) lt(3) }
```

Subsequent object identifiers defined in this document originate from this root identifier, except where declared otherwise.

6.2 Node

A single input element to a linking operation or to a step of an aggregation is represented by the following data structure:

```
Imprint ::= OCTET STRING
Imprints::=  SEQUENCE SIZE (1..MAX) OF Imprint
Node ::= CHOICE {
    imprints       [0] Imprints,
    reference      [1] INTEGER
}
Nodes ::= SEQUENCE SIZE (1..MAX) OF Node
```

Data field	Description
imprints	input data items for a link
reference	local identifier of a link from which to obtain input data items

If the 'imprints' field is present, the node contains the actual data items to be used as input to the current operation. If the 'reference' field is present, the node identifies a 'Link' structure (defined below) whose result is to be used as input to the current operation. In this case the value of the 'reference' field corresponds to the value of the 'identifier' field of the 'Link' structure whose results are to be included in the current operation. A value of zero in the 'reference' field indicates the data item to be used as input is obtained from a source external to the local group of links being computed.

6.3 Link

A single linking operation or a single step of an aggregation is represented by the following data structure:

```
Link ::= SEQUENCE {
    algorithm      [0] LinkAlgorithmIdentifier OPTIONAL,
    identifier     [1] INTEGER OPTIONAL,
    members        Nodes
}
Links ::= SEQUENCE SIZE (1..MAX) OF Link
```


© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Data field	Description
algorithm	object identifier of algorithm used to compute the link
identifier	local link identifier, when link is referred to by other links
members	data items or identifiers of data items to be linked

A link is computed by executing the algorithm specified in the 'algorithm' field on the input values contained in (or referenced by) the nodes in the 'members' field. If the results of this operation will be used as input to other operations, the local identifier assigned to this link is stored in the 'identifier' field.

If the 'algorithm' field is not present, the algorithm used to compute the link is specified at a higher level, for example in the 'Chain' structure (defined below). Clause B.2 in Annex B contains additional discussion about the algorithms that may be used.

6.4 Chain

A sequence of operations representing an aggregation or publishing process is contained in the following structure:

```
Chain ::= SEQUENCE {
    algorithm      ChainAlgorithmIdentifier,
    links          Links
}
Chains ::= SEQUENCE SIZE (1..MAX) OF Chain
```

Data field	Description
algorithm	object identifier of algorithm used to compute the chain of links
links	links that form the elements of the chain

A chain of links is computed by executing the algorithm specified in the 'algorithm' field on the sequence of links contained in the 'links' field. For example, a chain of links may be constructed to represent the result of a Merkle hash tree [M80] calculation; in this case the sequence of links represent the computational path from a specific leaf node to the common root. Clause B.3 in Annex B contains additional discussion about the algorithms that may be used.

6.5 BindingInfo

Data items generated by a TSA as a result of linking a time-stamp token to previously generated tokens are encapsulated by the following structure:

```
MessageImprints ::= SEQUENCE SIZE (1..MAX) OF MessageImprint
BindingInfo ::= SEQUENCE {
```

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
    version         Version,
    msgImprints     MessageImprints,
    aggregate       [0] Chains OPTIONAL,
    links           Links,
    publish         [1] Chains OPTIONAL,
    extensions      [2] Extensions OPTIONAL
}
```

Data field	Description
version	version number of this data structure
msgImprints	digest or digests computed on encapsulated TSTInfo structure, as defined in ISO/IEC 18014-1
aggregate	results of aggregating 'msgImprints' field with other members of the aggregation
links	links from prior time values
publish	results of aggregating time-stamp token with publishing data, if present
extensions	additional extension fields, if present

The ‘links’ field contains the results of linking operations (from prior time values) that were used as input to the linking operation for the time value in the encapsulated ‘TSTInfo’ digested in ‘msgImprints’. It always includes the result of the linking operation for the immediately preceding time value, which represents the running summary of the cumulative linking operations so far. Given a ‘BindingInfo’ structure, the result of the linking operation for the time value in the encapsulated ‘TSTInfo’ may be computed as follows: if no aggregation exists, the contents of the ‘links’ field are linked along with the contents of the 'msgImprints' field; otherwise the contents of the ‘links’ field is linked to the result of the computation of the 'aggregate' field.

The ‘aggregate’ field may contain multiple groups of aggregation data when multi-tiered aggregation schemes are supported.

6.6 Extensions

A time-stamping service may include extensions in the tokens it generates. The TSA computes the contents of the extension and inserts the results in an extension structure, defined as follows:

```
Extension ::= SEQUENCE {
    extnId          EXTENSION.&id ({ExtensionSet}),
    critical        BOOLEAN DEFAULT FALSE,
    extnValue       OCTET STRING
    -- contains a DER encoding of a value of type
    -- &ExtnType  for the extension object identified by extnId
}
ExtensionSet EXTENSION ::= { ... }
```


--`,,`-`-`,,`,,`,`,,`---
© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

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

Each extension defined below is encoded in an identical fashion. The data associated with the extension is encoded using DER encoding rules, and the resulting octets are inserted in the 'extnValue' field as an octet string.

The extensions declared below are those known at time of writing; future versions of this standard may include additional extensions.

6.6.1 Request and Token extensions

The following extensions fields are identified to be carried either by the ‘TimeStampReq’ or the ‘TimeStampToken’.

6.6.1.1 ExtHash extension

A requester of time-stamping services may wish to submit for time-stamping more than one hash value derived from a single data item. To enable the submission of multiple hash values the following extension is defined:

```
ExtHash ::= SEQUENCE SIZE(1..MAX) OF MessageImprint

extHash EXTENSION ::= {
    SYNTAX ExtHash IDENTIFIED BY tsp-ext-hash
}

tsp-ext-hash OBJECT IDENTIFIER ::= { tsp-ext 1 }
```

This extension is carried in the "extensions" field of both the ‘TimeStampReq’ message sent by a requester to a TSA and in the "extensions" field of the resulting ‘TimeStampToken’ structure formed by the TSA and returned to the requester. The TSA reproduces the contents of this extension unmodified.

6.6.1.2 ExtMethod extension

A requester of time-stamping services may wish to indicate to a specific TSA which time-stamping method to use when forming the eventual time-stamp token. To enable a requester to indicate to a specific TSA which time-stamping method to use in forming the resulting time-stamp token, the following extension is defined:

```
ExtMethod ::= SEQUENCE SIZE(1..MAX) OF Method

Method ::= METHOD.&id({Methods})

tsp-ext-meth OBJECT IDENTIFIER ::= { tsp-ext 2 }

extMethod EXTENSION ::= {
    SYNTAX ExtMethod IDENTIFIED BY tsp-ext-meth
}
```

This extension is carried in the "extensions" field of both the TimeStampReq message sent by a requester to a TSA and in the "extensions" field of the resulting TimeStampToken structure formed by the TSA and returned to the requester.

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

If this extension is present and the TSA is able to process it, then the TSA must attempt to fulfil the request for the specified method, or return an error indicating the method is not available. If the requester specifies more than one possible method, the TSA may select one of the suggested methods for use in forming the time-stamp token. If this extension is not present, the TSA uses its default time-stamping mechanism.

6.6.1.3 ExtRenewal extension

A requester of time-stamping services may wish to indicate to the TSA that the current time-stamping request is a renewal of a time-stamp on data that were already time-stamped in the past, so that the validity period of the old time-stamp is effectively extended. For this purpose, the following extension is defined:

```
ExtRenewal ::= TimeStampToken

tsp-ext-renewal OBJECT IDENTIFIER ::= { tsp-ext 3 }

extRenewal EXTENSION ::= {
    SYNTAX ExtRenewal IDENTIFIED BY tsp-ext-renewal
}
```

This extension is carried in the “extensions” field of both the TimeStampReq message sent by a requester to a TSA and in the “extensions” field of the resulting TimeStampToken structure formed by the TSA and returned to the requester. The TSA reproduces the contents of this extension unmodified.

6.6.2 BindingInfo extensions

The following extensions fields are identified to be carried by the ‘BindingInfo’.

6.6.2.1 ExtName extension

A TSA may identify each step of the overall time-stamping process by a distinct name, for auditing or record-keeping purposes. This additional name is carried as an extension defined as follows:

```
ExtName ::= GeneralName

tsp-ext-name ::= OBJECT IDENTIFIER { tsp-lt name (5) }

extName EXTENSION ::= {
    SYNTAX ExtName IDENTIFIED BY tsp-ext-name
}
```

A TSA may include this extension, for example, when the overall time-stamping process is implemented by a co-operating set of distinct processes.

6.6.2.2 ExtTime extension

A TSA may record the time value at which each step in the overall time-stamping process occurred, again for auditing or record-keeping purposes. This additional time value is carried as an extension defined as follows:

```
ExtTime ::= GeneralizedTime

tsp-ext-time ::= OBJECT IDENTIFIER { tsp-lt time (6) }

extTime EXTENSION ::= {
    SYNTAX ExtTime IDENTIFIED BY tsp-ext-time
}
```

A TSA may include this extension, for example, when the overall time-stamping process is implemented by a co-operating set of distinct processes.

--`,,```-`-`,,`,,`,`,,`---

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

7 Time-stamp request

The requester shall use the ExtMethod extension that is specified in the extensions field of the time-stamp request message- with the following object identifier-:

```
tsp-req-link OBJECT IDENTIFIER ::= {tsp-lt link(1)}
```

for requesting time-stamp tokens using the DigestedData encapsulation as described in subclause 8.1.

Alternatively, the requester shall use the Extmethod extension that is specified in the extensions field of the time-stamp request message with the following object identifier:

```
tsp-req-link-ds OBJECT IDENTIFIER ::= {tsp-lt link-ds(2)}
```

for requesting time-stamp tokens using the SignedData encapsulation as described in subclause 8.2.

8 Time-stamp response

The time-stamp response returned by a conformant TSA may include a 'TimeStampToken' structure, defined as follows:

```
TimeStampToken ::= SEQUENCE {
    contentType    CONTENT.&id({Contents}),
    content    [0] EXPLICIT CONTENT.&Type({Contents}{@contentType})
}
```

A conformant TSA produces linked time-stamp tokens according to the following steps.

Time-stamp information is first generated by the TSA according to the 'TSTInfo' structure defined in ISO/IEC 18014-1. This structure contains the message digest of the data to be time-stamped, the time value assigned by the TSA, and other relevant information. The 'tsa' and 'extensions' fields of TSTInfo shall be present. The 'tsa' field is required for time-stamp verification. The 'extensions' field is required to carry the identification of the time-stamping mechanism used.

The TSA then encapsulates the 'TSTInfo' contents, generates one or more hash values on the resulting bit strings, and links the resulting data with data previously generated and persistently stored by the TSA using the set of linking and publishing operations chosen by the TSA. The resulting linking, aggregation, and publishing information is inserted in a 'BindingInfo' structure as defined in clause 6.5.

Finally, a conformant time-stamp token is formed by encapsulating a 'TSTInfo' structure and its corresponding 'BindingInfo' structure in a 'TimeStampToken' structure. The TSA may perform either of two types of encapsulation:

- 'DigestedData' encapsulation, in which case the contentType component shall contain the value id-digestedData and the content component shall contain a value of type 'DigestedData; this encapsulation shall be used when the time-stamp request includes the ExtMethod extension with value the object identifier tsp-link.

- 'SignedData' encapsulation, in which case the contentType shall contain the value id-signedData and the content component shall contain a value of type 'SignedData'; this encapsulation shall be used when the time-stamp request includes the ExtMethod extension with value the object identifier tsp-link-ds.

The following definitions apply for the encapsulation of the 'TSTInfo' structure:

```
EncapsulatedContentInfo::= SEQUENCE {
    eContentType     CONTENT.&id({EContents}),
```

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
    eContent       [0] EXPLICIT CONTENT.&Type({EContents} {@eContentType})
}
EContents CONTENT ::= {
    { ETSTInfo IDENTIFIED BY id-ct-TSTInfo }
    ...
}
ETSTInfo ::= OCTET STRING (CONTAINING TSTInfo ENCODED BY der)
```

A TSA shall first encode the generated 'TSTInfo' structure using DER encoding rules, and store the resulting octets in the 'eContent' field of an 'EncapsulatedContentInfo' structure.

The TSA shall then compute one or more hash values on this 'eContent' field using one or more hash-functions. The resulting hash value or values are inserted in the 'msgImprints' field of the 'BindingInfo' structure defined in clause 6.5.

The TSA shall then use the 'msgImprints' field in linking operations, possibly in aggregation with 'msgImprints' fields from other concurrent time-stamp requests, and shall generate values for the 'links', 'aggregate' and optionally for the 'publish' fields of the 'BindingInfo' structure for this time-stamp request.

The following sub-clauses describe the remaining steps for the two possible types of encapsulation.

8.1 DigestedData encapsulation

The following object identifier identifies the digested data content type:

```
id-digestedData OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 5
}
```

The 'DigestedData' structure is defined in [CMS] as follows:

```
DigestedData ::= SEQUENCE {
    version             CMSVersion,
    digestAlgorithm     DigestAlgorithmIdentifier,
    encapContentInfo    EncapsulatedContentInfo,
    digest              Digest
}
```

with the following according to ISO/IEC 18014-3 Annex A:

```
DigestAlgorithmIdentifier ::= AlgorithmIdentifier {{ DigestAlgorithms }}
Digest ::= OCTET STRING
```

A TSA using the 'DigestedData' encapsulation method shall first perform the steps described in Clause 8 and place the resulting 'EncapsulatedContentInfo' structure in the 'encapContentInfo' field of the 'DigestedData' structure. Once all fields of the 'BindingInfo' structure have been computed, the TSA shall encode the 'BindingInfo' structure using DER encoding rules, and store the resulting octets in the 'digest' field of the 'DigestedData' structure.

Finally, the TSA will insert the object identifier corresponding to this encapsulation method into the 'digestAlgorithm' field. The object identifier for this encapsulation method is defined as follows:

```
tsp-digestedData ::= OBJECT IDENTIFIER { tsp-lt digestedData(8) }
```


© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

8.2 SignedData encapsulation

A TSA may integrate digital signatures with a time-stamping system producing linked tokens. By linking together tokens signed independently the TSA can verifiably demonstrate the order in which the individually signed tokens were issued.

The following object identifier identifies the signed data content type:

```
id-signedData OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7) 2
}
```

The ‘SignedData’ structure as defined in ISO/IEC 18014-1 Annex A appears as follows:

```
SignedData    ::= SEQUENCE {
    version             CMSVersion,
    digestAlgorithms    DigestAlgorithmIdentifiers,
    encapContentInfo    EncapsulatedContentInfo,
    certificates        [0] IMPLICIT CertificateSet OPTIONAL,
    crls                [1] IMPLICIT CertificateRevocationLists OPTIONAL,
    signerInfos         SignerInfos
}
```

with

```
SignerInfos ::= SET OF SignerInfo
SignerInfo ::= SEQUENCE {
    version             CMSVersion,
    sid                 SignerIdentifier,
    digestAlgorithm     DigestAlgorithmIdentifier,
    signedAttrs         [0] IMPLICIT SignedAttributes OPTIONAL,
    signatureAlgorithm  SignatureAlgorithmIdentifier,
    signature           SignatureValue,
    unsignedAttrs       [1] IMPLICIT UnsignedAttributes OPTIONAL
}
SignedAttributes ::= SET SIZE (1..MAX) OF Attribute
UnsignedAttributes ::= SET SIZE (1..MAX) OF Attribute
Attribute ::= SEQUENCE {
    attrType            OBJECT IDENTIFIER,
    attrValues          SET OF AttributeValue
}
AttributeValue      ::= ANY
```

© ISO/IEC 2004 - All rights reserved
11
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

The 'SignedData' structure is used to encapsulate both the signature data and linking data in this case. A 'SignedData' structure contains one or more 'SignerInfo' structures, which in turn may contain additional signed or unsigned attributes.

A TSA using the 'SignedData' encapsulation method shall first perform the steps described in Clause 8 and place the resulting 'EncapsulatedContentInfo' structure in the 'encapContentInfo' field of the 'SignedData' structure. Once all fields of the 'BindingInfo' structure have been computed, the TSA shall encode the 'BindingInfo' structure as an octet string using DER encoding rules, and insert it as an element of the 'attrValues' field in an 'Attribute' structure. The TSA shall also insert the object identifier corresponding to this encapsulation method into the 'attrType' field. The object identifier for this encapsulation method is defined as follows:

```
tsp-signedData ::= OBJECT IDENTIFIER { tsp-lt signedData(9) }
```

The TSA shall then insert the resulting 'Attribute' structure in the 'signedAttrs' field of a 'SignerInfo' structure, and complete the signing process.

It is important to note that inserting the 'BindingInfo' structure as a signed attribute preserves the validity of the 'EncapsulatedContentInfo' structure beyond the lifetime of the digital certificate used to form the digital signature.

9 Time-stamp verification

The validity of a time-stamp token may be verified by checking that:

- the time-stamp token is syntactically well-formed;
- the value of the messageImprint component of the time-stamp token matches the value of a messageImprint evaluated over the document subject to scrutiny;
- the value of the messageImprint component of every ExtHash extension in the time-stamp token matches the value of a messageImprint evaluated over the document subject to scrutiny, and
- the policy under which the token is issued is acceptable for the intended usage.

In the case of a time-stamp token with 'DigestedData' encapsulation, the verifier shall also carry out a verification protocol exchange with the issuing TSA or other third-party service that has access to the issuing TSA's summary links, using a data integrity and data origin authentication protected channel. If there is no summary link value in the TSA's or third-party service's repository of summary link values with the same time value as the one in the submitted time-stamp token's encapsulated 'TSTInfo' structure, the verification fails. Otherwise, the issuing TSA or third-party service shall proceed to verify that the 'msgImprints' in the 'BindingInfo' field match the hash values computed over the encapsulated 'TSTInfo' structure. If not, the verification fails; otherwise, if the 'aggregate' field is present in the 'BindingInfo' structure, the issuing TSA or third-party service shall carry out the computation in the 'aggregate' field (assuming that a zero 'reference' value within a 'Node' refers to the 'msgImprints' field of 'BindingInfo'), and apply the linking algorithm on this result and the 'links' field in order to verify whether the result matches the archived summary link for the time value in the time-stamp token's encapsulated 'TSTInfo'; if the 'aggregate' field is missing, the issuing TSA or third-party service shall apply the linking algorithm on the 'msgImprints' and 'links' fields of the 'BindingInfo' structure in the same manner, and carry over the same verification step on the resulting value. The linking algorithm used shall be the one that was used to generate the summary link value in the summary link value repository for the same time value.

In the case of a time-stamp token with 'SignedData' encapsulation, the verifier may carry out the same verification protocol as the one described in the case of 'DigestedData' encapsulation, or perform verification in the same manner as for time-stamp tokens with digital signatures described in ISO/IEC 18014-2.

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex A
(normative)

ASN.1 Module for time-stamping

This module contains correct ASN.1 based on the current ASN.1 standards that has passed successfully through a reliable syntax checker used by the ITU-T ASN.1 Project.

```
TimeStampProtocol-3 {
    iso(1) standard(0) time-stamp(18014) modules(0) part3(3)}
    DEFINITIONS IMPLICIT TAGS ::= BEGIN
-- EXPORTS All; --
IMPORTS
   -- ISO/IEC 9594-8 | ITU-T Rec. X.509 AuthenticationFramework --
    EXTENSION, Extensions
              FROM AuthenticationFramework {
                   joint-iso-itu-t ds(5) module(1) authenticationFramework(7) 4 }
    -- ISO/IEC 9594-8 | ITU-T Rec. X.509 CertificateExtensions --
    GeneralName
              FROM CertificateExtensions {
                   joint-iso-itu-t ds(5) module(1) certificateExtensions(26) 4 }
    SignedData
              FROM CryptographicMessageSyntax {
                   iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs-9(9)
                   smime(16) modules(0) cms(1) };
TimeStampReq ::=  SEQUENCE {
    version          Version,
    messageImprint   MessageImprint,
    reqPolicy        TSAPolicyId OPTIONAL,
    nonce            Nonce OPTIONAL,
    certReq          BOOLEAN DEFAULT FALSE,
    extensions       [0] Extensions OPTIONAL
}
MessageImprint ::= SEQUENCE  {
    hashAlgorithm    DigestAlgorithmIdentifier,
    hashedMessage    OCTET STRING
}
DigestAlgorithmIdentifier ::= AlgorithmIdentifier {{ DigestAlgorithms }}
```

Copyright International Organization for Standardization · All rights reserved
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
DigestAlgorithms ALGORITHM ::= {
    { OID sha1 PARMS NULL },
    --
    ... -- Expect additional digest algorithms --
}
TSAPolicyId ::= POLICY.&id({TSAPolicies})
TSAPolicies POLICY ::= {
      --
    ...  -- Any supported TSA policy --
}
TimeStampResp ::= SEQUENCE {
    status          PKIStatusInfo,
    timeStampToken  TimeStampToken OPTIONAL
}
PKIStatusInfo ::=   SEQUENCE {
    status          PKIStatus,
    statusString    PKIFreeText OPTIONAL,
    failInfo        PKIFailureInfo OPTIONAL
}
PKIStatus ::= INTEGER {
    granted             (0),   -- the request is completely granted
    grantWithMods       (1),   -- modifications were necessary, the requester
                               -- is responsible for asserting the differences
    rejection           (2),   -- the request could not be fulfilled, the failure
                               -- code delivers additional information
    waiting             (3),   -- the request is not processed, the requester
                               -- receives a receipt that the request
                              -- has been received
    revocationWarning          (4),   -- a revocation is imminent
    revocationNotification     (5)    -- notification that a revocation
                                      -- has occurred
}
PKIFreeText ::= SEQUENCE SIZE(1..MAX) OF UTF8String
PKIFailureInfo ::=  BIT STRING {
    badAlg              (0),   -- unrecognized or unsupported Algorithm Identifier
    badRequest          (2),   -- transaction not permitted or supported
    badDataFormat       (5),   -- data submitted has the wrong format
    timeNotAvailable    (14),  -- the TSAs service is not available
    unacceptedPolicy    (15),  -- the requested TSA policy is not supported
    unacceptedExtension (16),  -- the requested TSA extension is not supported,
```


© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
    addInfoNotAvailable (17), -- the additional information requested
                              -- could not be understood or is not available,
    systemFailure       (25)   -- System Failure
}
TimeStampToken ::= SEQUENCE {
    contentType         CONTENT.&id({Contents}),
    content          [0] EXPLICIT CONTENT.&Type({Contents}{@contentType})
}
Contents CONTENT ::= {
    time-stamp-mechanism-linking-with-digest  |
    time-stamp-mechanism-linking-with-signature,
     --
    ...  -- Expect additional time-stamp mechanisms --
}
-- Time-stamp mechanism information objects --
time-stamp-mechanism-linking-with-digest CONTENT ::=
    { DigestedData IDENTIFIED BY id-digestedData }
time-stamp-mechanism-linking-with-signature CONTENT ::=
    { SignedData IDENTIFIED BY id-SignedData }
ETSTInfo ::= OCTET STRING (CONTAINING TSTInfo ENCODED BY der)
TSTInfo ::=  SEQUENCE {
    version          Version,
    policy           TSAPolicyId,
    messageImprint   MessageImprint,
    serialNumber     SerialNumber,
    genTime          GeneralizedTime,
    accuracy         Accuracy OPTIONAL,
    ordering         BOOLEAN DEFAULT FALSE,
    nonce            Nonce OPTIONAL,
    tsa              [0] EXPLICIT GeneralName OPTIONAL,
    extensions       [1] Extensions OPTIONAL
}
Version ::= INTEGER { v1(1) }
SerialNumber ::= INTEGER  -- Expect large values
Accuracy ::= SEQUENCE {
   seconds           INTEGER  OPTIONAL,
   millis            [0] INTEGER(1..999)  OPTIONAL,
   micros            [1] INTEGER(1..999)  OPTIONAL
}
 (ALL EXCEPT({ -- no components present -- }))
```

© ISO/IEC 2004 - All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
Ordering ::= BOOLEAN
Nonce ::= INTEGER  -- Expect large values
tsp OBJECT IDENTIFIER ::= { iso(1) standard(0) time-stamp(18014) }

-- Time-stamping extensions --
TSExtensions EXTENSION ::= {
    extHash    |
    extMethod  |
    extRenewal,
     --
    ...  -- Expect additional extensions --
}
extHash EXTENSION ::= {SYNTAX ExtHash IDENTIFIED BY tsp-ext-hash}
ExtHash ::= SEQUENCE SIZE(1..MAX) OF MessageImprint
extMethod EXTENSION ::= {SYNTAX ExtMethod IDENTIFIED BY tsp-ext-meth}
ExtMethod ::= SEQUENCE SIZE(1..MAX) OF Method
Method ::= METHOD.&id({Methods})

Methods METHOD ::= {
    --
    ...  -- Any time-stamping method  --
}
extRenwal EXTENSION ::= {SYNTAX ExtRenewal IDENTIFIED BY tsp-ext-renewal}
ExtRenewal ::= TimeStampToken
tsp-ext OBJECT IDENTIFIER ::= { tsp ext(1) }
tsp-ext-hash OBJECT IDENTIFIER ::= { tsp-ext 1 }
tsp-ext-meth OBJECT IDENTIFIER ::= { tsp-ext 2 }
tsp-ext-renewal OBJECT IDENTIFIER ::= { tsp-ext 3 }
EncapsulatedContentInfo::= SEQUENCE {
    eContentType     CONTENT.&id({EContents}),
    eContent         [0] EXPLICIT CONTENT.&Type({EContents}{@eContentType})
}
EContents CONTENT ::= {
    { ETSTInfo IDENTIFIED BY id-ct-TSTInfo },
    --
    ...     -- Expect additional content types --
}
-- Supporting definitions
AlgorithmIdentifier { ALGORITHM:IOSet } ::= SEQUENCE {
    algorithm       ALGORITHM.&id({IOSet}),
```


© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
    parameters      ALGORITHM.&Type({IOSet}{@algorithm})  OPTIONAL
}
ALGORITHM ::= CLASS {
    &id             OBJECT IDENTIFIER  UNIQUE,
    &Type           OPTIONAL
}
WITH SYNTAX { OID &id [PARMS &Type] }
CONTENT ::= TYPE-IDENTIFIER      -- ISO/IEC 8824-2, Annex A
OIDS ::= CLASS {
    &id             OBJECT IDENTIFIER  UNIQUE
}
WITH SYNTAX { OID &id }
POLICY ::= OIDS  -- Supported TSA policies
METHOD ::= OIDS  -- TSA Methods

der OBJECT IDENTIFIER ::= {
    joint-iso-itu-t asn1(1) ber-derived(2) distinguished-encoding(1) }
sha1 OBJECT IDENTIFIER ::= {
    iso(1) identified-organization(3) oiw(14) secsig(3) 2 26 }
md5 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) digestAlgorithm(2) 5}
pkcs7 OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1) pkcs7(7)
}
id-digestedData OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840)
    rsadsi(113549) pkcs(1) pkcs7(7) 5
}
id-signedData OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs7(7) signedData(2) }
id-ct-TSTInfo OBJECT IDENTIFIER ::= {
    iso(1) member-body(2) us(840) rsadsi(113549) pkcs(1)
    pkcs-9(9) smime(16) ct(1) 4 }
-- verification of a timestamp token
VerifyReq ::=     SEQUENCE {
    version       Version,
    tst           TimeStampToken,
    requestID     [0] OCTET STRING OPTIONAL
}
```

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
VerifyResp ::=      SEQUENCE {
    version       Version,
    status        PKIStatusInfo,
    tst           TimeStampToken,
    requestID     [0] OCTET STRING OPTIONAL
}
     -- ISO 18014-3 specific types
tsp-lt  OBJECT IDENTIFIER ::= { tsp lt(3) }
tsp-req-link OBJECT IDENTIFIER ::= {tsp-lt link(1)}
tsp-req-link-ds OBJECT IDENTIFIER ::= {tsp-lt link-ds(2)}
tsp-ext-name OBJECT IDENTIFIER ::= { tsp-lt name (5) }
tsp-ext-time OBJECT IDENTIFIER ::= { tsp-lt time(6) }
tsp-digestedData OBJECT IDENTIFIER ::= { tsp-lt digestedData(8) }
tsp-signedData OBJECT IDENTIFIER ::= { tsp-lt signedData(9) }

Digest ::= OCTET STRING -- (CONTAINING BindingInfo ENCODED BY der)
DigestedData ::= SEQUENCE {
    version             CMSVersion,
    digestAlgorithm     DigestAlgorithmIdentifier,
    encapContentInfo    EncapsulatedContentInfo,
    digest              Digest
}
MessageImprints ::= SEQUENCE SIZE (1..MAX) OF MessageImprint
BindingInfo ::= SEQUENCE {
    version             Version,
    msgImprints         MessageImprints,
    aggregate           [0] Chains OPTIONAL,
    links               Links,
    publish             [1] Chains OPTIONAL,
    extensions          [2] Extensions OPTIONAL
}
Imprint ::= OCTET STRING
Node ::= CHOICE {
    imprints  [0] Imprints,
    reference [1] INTEGER
}
Imprints ::=  SEQUENCE SIZE (1..MAX) OF Imprint
Nodes ::= SEQUENCE SIZE (1..MAX) OF Node
Chain ::= SEQUENCE {
    algorithm ChainAlgorithmIdentifier,
```


© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

```
    links     Links
}
ChainAlgorithmIdentifier ::= AlgorithmIdentifier {{ ChainAlgorithms }}
ChainAlgorithms ALGORITHM ::= {
    --
    ... -- Expect additional chain algorithms --
}
Chains ::= SEQUENCE SIZE (1..MAX) OF Chain
Link ::= SEQUENCE {
    algorithm        [0] LinkAlgorithmIdentifier OPTIONAL,
    identifier       [1] INTEGER OPTIONAL,
    members          Nodes
}
LinkAlgorithmIdentifier ::= AlgorithmIdentifier {{ LinkAlgorithms }}
LinkAlgorithms ALGORITHM ::= {
    --
    ... -- Expect additional link algorithms --
}
Links ::= SEQUENCE SIZE (1..MAX) OF Link
-- BindingInfo extensions
ExtName ::= GeneralName
extName EXTENSION ::= { SYNTAX ExtName IDENTIFIED BY tsp-ext-name }
ExtTime ::= GeneralizedTime
extTime EXTENSION ::= { SYNTAX ExtTime IDENTIFIED BY tsp-ext-time }

END  -- TimeStampProtocol 3 --
```

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex B
(normative)

Additional Discussion

B.1 Discussion

This annex provides additional explanatory text for use by conformant TSAs when choosing specific processes and techniques for computing links, aggregations and/or additional publishing information.

B.2 Linking

Linking is the means whereby a hash value representing either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures (containing the same time value) is cryptographically linked to other hash value representations, in order to both express the causal ordering of time-stamping events and to serve as a witness to all previously-linked events. Linking functions must be cryptographically strong, in the sense that once the link is made it cannot be changed or deleted without detection. Linking successive hash values together demonstrates the sequence in which they were generated, as a means for verifying the correct application of time values to individual 'TSTInfo' structures.

By linking each hash value together with all previously-linked hash values, the linking process also serves to make each newly linked value witness to all preceding values, in the sense that previously-linked values cannot be altered without successive calculations being affected. Each linked value identifies an event which, when widely witnessed, can be used to verify the correctness of all previously submitted events.

It is recommended that multiple hash-functions be used concurrently in the linking process so that the calculation of links be insulated from the cryptographic failure of any single hash function. For example, a TSA may perform a linking operation by first computing the MD5 hash value over the input data of the linking operation, then computing the SHA1 hash value over the same input data, and finally concatenating the two computed hash values together to generate the result of the linking operation. As noted in the 'BindingInfo' type definition in Clause 6.5 of this document, multiple hash values representing a single encapsulated 'TSTInfo' structure may be stored in the 'msgImprints' field of the 'BindingInfo' structure and may be used as input to the linking or aggregation process. In subsequent sub-clauses, it is assumed that whenever a single hash value representing some data is mentioned, a TSA may instead utilise multiple hash values representing the same data, computed using distinct hash-functions.

B.2.1 Linear chain linking

The hash value representing either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value, is linked to all previously-linked links in the repository maintained by the TSA, such that the most recently generated link in effect forms a running summary of all previously linked hash values. Each link in the linear chain is formed by concatenating the previous running summary link with the current hash value to be linked, and using the resulting bitstring as input to a hash-function. The current hash value may represent either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value. The output of the hash-function becomes the new running summary link and is stored in a repository by the TSA; the previous running summary link is returned to the requester in the 'links' field of the corresponding 'BindingInfo' structure. Each value produced by the linking process is retained in the repository, to support future verification requests, as well as publishing operations.

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

B.2.2 Anti-monotone binary linking

Using this method of linking, the repository of link values maintained by the TSA can be represented as an anti-monotone directed acyclic graph with each vertex having in-degree equal to 2 [BLS00]. The hash value representing either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value, is linked to all previously-linked hash values, such that the most recently generated link in effect forms a running summary of all previously linked hash values. Each link in the anti-monotone directed graph is formed by concatenating two previous links of the directed graph maintained by the TSA (one of them the immediately previous running summary link) with the current hash value to be linked, and using the resulting bitstring as input to a hash-function. The current hash value may represent either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value. The output of the hash-function becomes the new running summary link, and is stored in a repository by the TSA; the two previous links that were used in the linking operation are returned to the requester in the 'links' field of the corresponding 'BindingInfo' structure. Each value produced by the linking process is retained in the repository, to support future verification requests, as well as publishing operations.

B.2.3 Threaded tree linking

This method of linking uses threaded authentication trees, which are similar to the Merkle hash trees[M80], as augmented by additional edges amongst the vertices [BLS00]. The hash value representing either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value, is linked to all previously-linked hash values, such that the most recently generated link in effect forms a running summary of all previously linked hash values. Each link in the threaded tree is formed by concatenating appropriate number of previous links of the directed graph maintained by the TSA (one of them the immediately previous running summary link) with the current hash value to be linked, and using the resulting bitstring as input to a hash-function. The current hash value may represent either a single encapsulated 'TSTInfo' structure or an aggregation of multiple encapsulated 'TSTInfo' structures, containing the same time value. The output of the hash-function becomes the new running summary link, and is stored in a repository by the TSA; the previous links that were used in the linking operation are returned to the requester in the 'links' field of the corresponding 'BindingInfo' structure. Each value produced by the linking process is retained in the repository, to support future verification requests, as well as publishing operations.

B.3 Aggregation

Aggregation is the means whereby a group of hash values representing multiple encapsulated 'TSTInfo' structures, containing the same time value, are used to compute a single output value, which can serve as a direct proxy to the original 'TSTInfo' structures in the subsequent linking process.

The aggregation process combines groups of hash values into a single aggregate value, where each input hash value contributes in some manner to the contents of the resulting aggregate value. The techniques used to combine multiple hash values into a single aggregate value rely on the cryptographic characteristics of one-way hash-functions or other accumulation algorithms. The aggregation process also produces additional data required to authenticate the membership of a single hash value in the group of hash values used to produce a given aggregate value. This data is inserted in the 'aggregate' field of the corresponding 'BindingInfo' structure.

B.3.1 Unity aggregation

The simplest form of aggregation is no aggregation at all. That is, while there are significant operational and security benefits to aggregating hash values representing encapsulated 'TSTInfo' structures together, the time-stamping process is not required to aggregate them. It is possible for hash values produced from individual encapsulated 'TSTInfo' structures to simply go directly to the linking step without being aggregated with any other hash values. While this approach simplifies the time-stamping process, it also greatly increases the load on the linking process. Time-stamp tokens produced without aggregation also do not benefit from the additional security provided by allowing each member of an aggregate group to directly

© ISO/IEC 2004 - All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

witness the time-stamping event for all other members of the group. Verifying a time-stamp token formed with unity aggregation requires no additional work beyond verifying the associated linking step.

B.3.2 Merkle tree aggregation

Hash value representations of encapsulated 'TSTInfo' structures can be aggregated using the authentication tree originally proposed by Merkle [M80] for use in forming directories of cryptographic keys. In this application the binary tree construct from [M80] is used to aggregate the hash values representing a group of individual encapsulated 'TSTInfo' structures rather than cryptographic keys. This aggregation process is implemented by selecting a group of encapsulated 'TSTInfo' structures to aggregate, and computing hash representations of them. The resulting hash values are treated as leaves of a binary tree. Each intermediate node in the tree is computed by concatenating the child hash values for that node and treating the concatenation as input to a hash-function. The output of the hash calculation becomes the value of that node. The computation proceeds in this manner, computing successive nodes at higher levels in the tree, until a single root hash value has been calculated. As all leaf values contribute to the final calculation of the root value, that root value can be used as a direct proxy to all leaf values.

When this binary tree construct is used to aggregate hash value representations of encapsulated 'TSTInfo' structures, the TSA must return in the 'aggregate' field of the 'BindingInfo' structure corresponding to each encapsulated 'TSTInfo' structure those intermediate hash values required to compute the path from the original leaf hash value to the root hash value. For an aggregation of N encapsulated 'TSTInfo' structures, only $\log_2 N$ intermediate hash values are required to compute the path from the leaf hash value to the root hash value. Verification of the 'aggregate' field of the corresponding 'BindingInfo' structure requires $\log_2 N$ hash calculations to authenticate the leaf hash value against the generated aggregate value.

B.3.3 One-way accumulator aggregation

Another form of aggregation is through application of a one-way accumulator. This form of aggregation seeks to compute an aggregation such that verifying membership in the aggregation requires a single constant-time computation, regardless of the number of values in the aggregation. One technique proposed for realising such an accumulator is that proposed by Benaloh and de Mare in [BD93], which uses successive exponentiations modulo an RSA modulus of unknown factorisation to compute a single accumulated value from multiple input values. The same technique is also used to compute verifiable statements for each input value that demonstrates membership of the input value in the aggregate value.

For a collection of hash values $Y = (y_1, y_2, ..., y_N)$ representing encapsulated 'TSTInfo' structures, an RSA modulus $n=pq$ (where p and q are primes) and the starting seed (a number modulo n) denoted by x, the accumulation algorithm computes:

$$A(Y) = x^{y_1 y_2 \cdots y_N} \bmod n$$

The proof construction algorithm P constructs the proof for the element y_i as part of the aggregation to be the value

$$P(y_i, Y) = x^{y_1 y_2 \cdots y_{i-1} y_{i+1} \cdots y_N} \bmod n$$

and inserts it in the 'aggregate' field of the corresponding 'BindingInfo' structure, encapsulated in a 'Chain' structure.

Verifying the aggregation step amounts to checking that

$$P(y_i, Y)^{y_i} = A(Y)$$

Thus, verification of the aggregation step for any participating hash value requires computation time that is independent of N.

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

B.4 Publishing

An event that is witnessed by many independent observers cannot be effectively repudiated. By exposing an event to multiple observers and having them maintain sufficient records about the event, it becomes impractical to modify all existing records of the event, or to collude with all observers to modify all existing records of the event, or to collude with all observers to modify the event. Witnessing can occur at multiple levels, in order to provide multiple layers of assurance. Witnessing in general means that links produced by the TSA are made widely available, such that any attempt to modify those links after the fact is impractical. A witnessed set of events may also be audited by parties external to the TSA, providing additional assurance of integrity.

Publishing is a means whereby the linking process itself can become a “widely-witnessed” event. The manner most commonly used to ensure the linking process produces widely-witnessed events is through the periodic publication of linked hash values on some well-known media. For example, a publishing process may periodically insert hash values from the linking process into web pages, where those hash values will thus be widely disseminated throughout the Internet via web caching and indexing agents. Another publishing process may choose to publish linking information in offline media, such as classified ads in newspapers with global distribution. Other publishing processes may insert linking information into email messages or online newsgroups, as again a means for wide dissemination of information.

B.4.1 No publishing

A time-stamping process may choose to bypass the publication process, for example if operational concerns preclude the practical use of publication processes. Such systems cannot, however, claim to be “widely-witnessed.”

B.4.2 Single link publishing

The simplest publishing process is one that selects a single linked hash value from the linear chain of linked values for publication. Verifying the correctness of a link in such a system requires the linking process to retain both the set of linked hash values and the individual or aggregate hash values used to form those links. A party wishing to verify their linking data against the published value first computes their place in an aggregation (if any), then obtains the linked hash values and input hash values needed to compute the links from the published value to their place in the linear chain. The number of hash values a requestor must retrieve in order to validate a token against a published hash value grows linearly with the number of links generated in a chosen publication interval.

B.4.3 Merkle tree publishing

Another approach to publication is to accumulate a group of linked values together using another Merkle hash tree [M80] computation. In this case the linked hash values for some chosen interval of time are aggregated together, and the aggregate value is then published. As with the aggregation process for encapsulated ‘TSTInfo’ structures described earlier, each leaf linked hash value in a publication Merkle tree contributes to the computation of the root value, and hence the root value can be used as a direct proxy to all the leaf values. Using binary trees to compute publication values ensures that the number of hash values that must be retrieved in order to validate a time-stamp token against the published value is $\log_2 N$, where N is the number of linked hash values generated in a chosen publication interval.

© ISO/IEC 2004 – All rights reserved

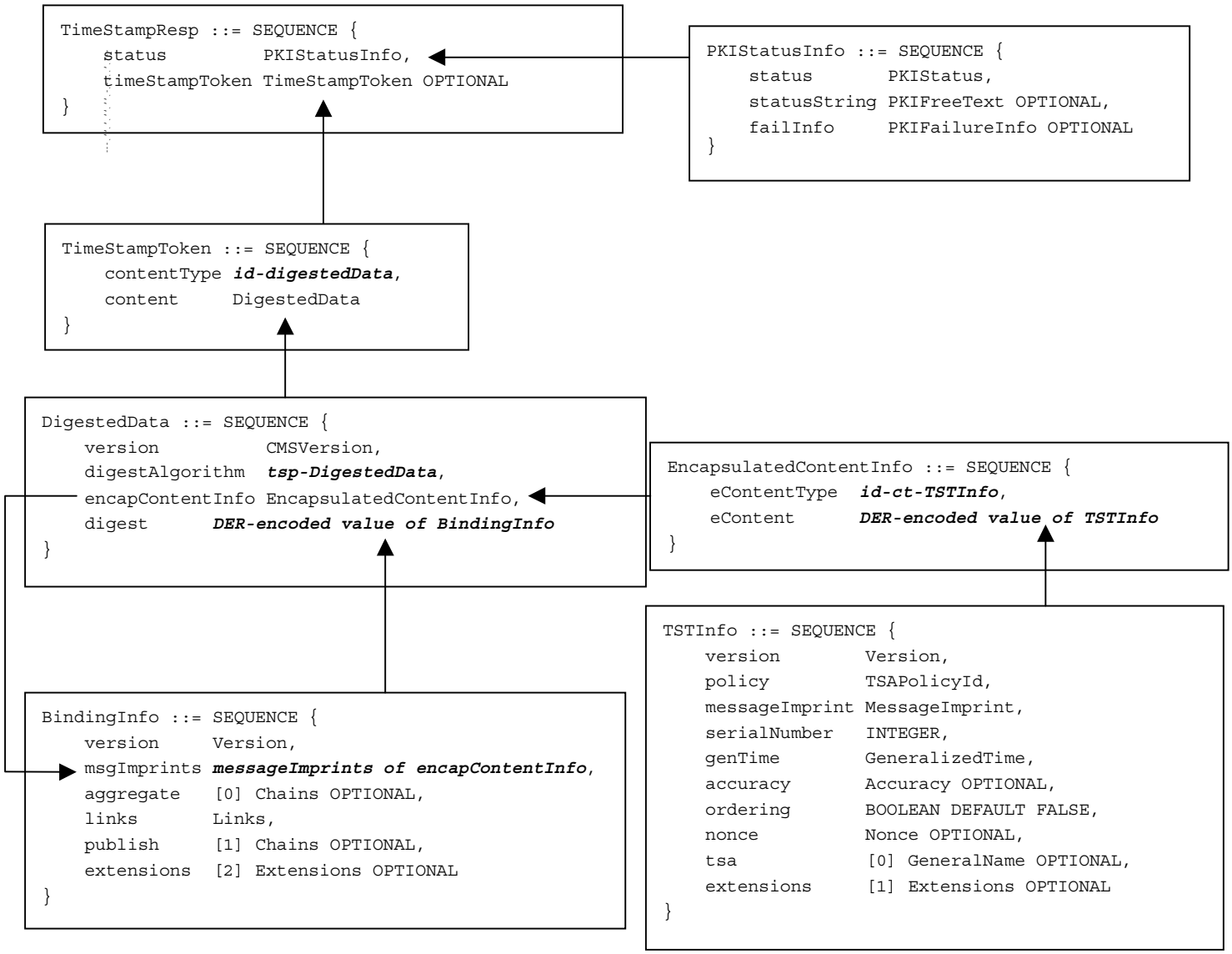
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Annex C
(informative)

Data structures

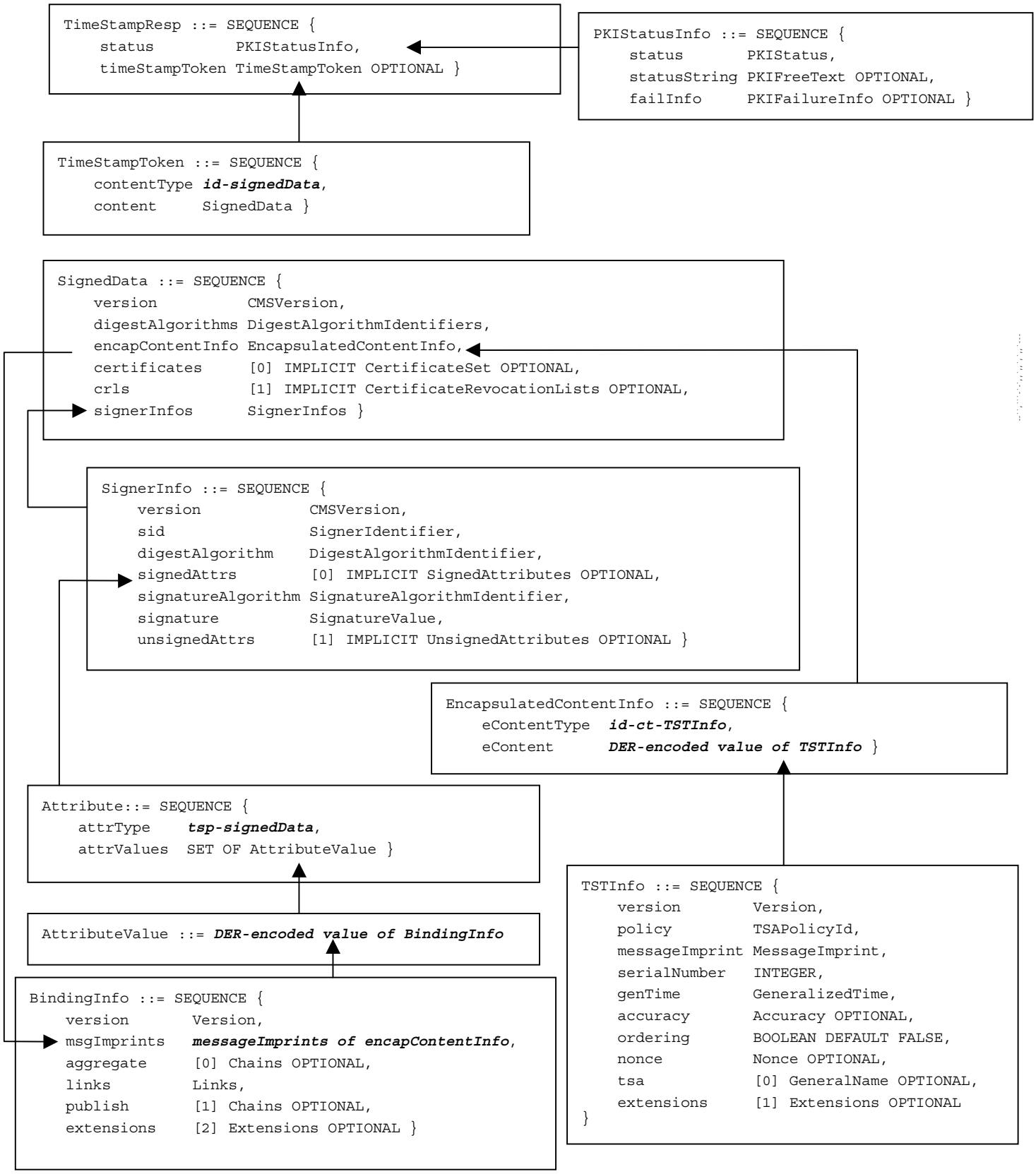
The following diagrams complement the ASN.1 descriptions in the normative text.

C.1 Time-stamps using DigestedData encapsulation



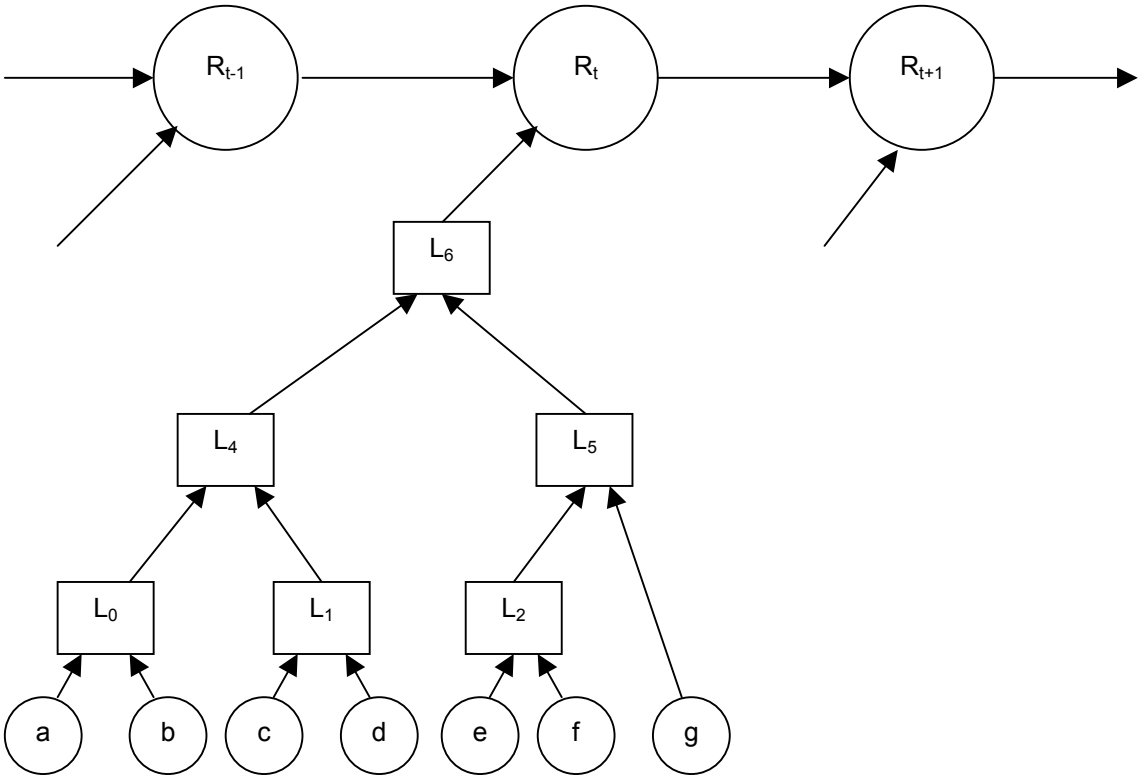
© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.2 Time-stamps using SignedData encapsulation



© ISO/IEC 2004 - All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

C.3 Linear Chain Linking with Merkle Tree Aggregation



The input nodes at time *t* are: a, b, c, d, e, f, g.

The Merkle tree for the aggregation step at time t generates link values: L_0, L_1, L_2, L_4, L_5, L_6.

The link value of the linear chain in the repository at time t-1 is R_{t-1}.

The newly calculated link value of the linear chain in the repository at time t is R_t.

Let H be the algorithm used in all steps. The following equations hold true.

$L_0 = H(a, b)$ $L_1 = H(c, d)$ $L_2 = H(e, f)$

$L_4 = H(L_0, L_1)$ $L_5 = H(L_2, g)$

$L_6 = H(L_4, L_5)$

$R_t = H(R_{t-1}, L_6)$

The aggregation chains corresponding to the input nodes are as follows:

$C_a = \{a, b, L_1, L_5\}$ $C_b = \{a, b, L_1, L_5\}$

$C_c = \{c, d, L_0, L_5\}$ $C_d = \{c, d, L_0, L_5\}$

$C_e = \{e, f, g, L_4\}$ $C_f = \{e, f, g, L_4\}$

$C_g = \{L_2, g, L_4\}$

© ISO/IEC 2004 — All rights reserved
Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

All these aggregation chains produce the value L_6 when the computation steps are carried out.

The computation of a chain consists in applying the algorithm H repeatedly to the elements of the chain chaining until a single value remains. For example:

$$\begin{aligned}
\text{Compute}(C_d) &= \text{Compute}(\{c, d, L_0, L_5\}) \\
&= \text{Compute}(\{H(c, d), L_0, L_5\}) \\
&= \text{Compute}(\{L_1, L_0, L_5\}) \\
&= \text{Compute}(H(L_0, L_1), L_5\}) \\
&= \text{Compute}(\{L_4, L_5\}) \\
&= \text{Compute}(\{H(L_4, L_5)\}) \\
&= \text{Compute}(\{L_6\}) \\
&= L_6
\end{aligned}$$

--`,,```-`-`,,`,,`,`,,`---

© ISO/IEC 2004 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS
Not for Resale

Bibliography

[BD91] “Efficient Broadcast Time-Stamping,” J. Benaloh, M. de Mare, TR 91-1, Clarkson University Department of Mathematics and Computer Science, 1991.

[BD93] “One-Way accumulators: A Decentralized Alternative to Digital Signatures,” J. Benaloh, M. de Mare, Advances in Cryptology - EUROCRYPT 1993, Lecture Notes in Computer Science, vol.765, pp. 274-285, Springer-Verlag, 1994.

[BHS93] “Improving the Efficiency and Reliability of Digital Time-Stamping,” D. Bayer, S. Haber, W. Stornetta, Sequences II: Methods in Communication, Security and Computer Science, pp. 329-334, Springer-Verlag, 1993.

[BLLV98] “Time-Stamping with Binary Linking Schemes,” A. Buldas, P. Laud, H. Lipmaa, J. Villemson, Advances in Cryptology - CRYPTO 1998, Lecture Notes in Computer Science, vol. 1462, pp. 486-501, Springer-Verlag, 1998.

[BLS00] “Optimally Efficient Accountable Time-Stamping,” A. Buldas, H. Lipmaa, B. Schoenmakers, Public Key Cryptography 2000, pp. 293-305, Springer-Verlag, January 2000.

[HS91] “How to Time-Stamp a Digital Document,” S. Haber, W. Stornetta, Journal of Cryptology, Vol. 3, No. 2, pp. 99-111, 1991.

[HS97] “Secure Names for Bit-Strings,” S. Haber, W. Stornetta, Proceedings of the 4th ACM Conference on Computer and Communication Security, pp. 28-35, ACM Press, 1997.

[J98] ”Some Timestamping Protocol Failures”, M. Just, Proceedings of the Internet Society Symposium on Network and Distributed System Security, p. 5, Internet Society, 1998.

[M80] “Protocols for Public Key Cryptosystems,” R. Merkle, Proceedings of the IEEE Symposium on Security and Privacy, pp. 122-133, IEEE Computer Society, 1980.

[PKIXCP] "Internet X.509 Public Key Infrastructure Certificate and CRL Profile" R. Housley, W. Ford, W. Polk, D. Solo, IETF RFC 3280, 2002.

[CMS] “Cryptographic Message Syntax”, R. Housley, IETF RFC 3369, 2002.

ISO/IEC 8824 (all parts), *Information technology — Abstract Syntax Notation One (ASN.1)*

ISO/IEC 8825 (all parts), *Information technology — ASN.1 encoding rules*

ISO/IEC 9796-3, *Information technology — Security techniques — Digital signature schemes giving message recovery — Part 3: Discrete logarithm based mechanisms*

ISO/IEC 11770-3, *Information technology — Security techniques — Key management — Part 3: Mechanisms using asymmetric techniques*

© ISO/IEC 2004 — All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

--`,,```-`-`,,`,,`,`,,`---

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale

ICS 35.040

Price based on 28 pages

© ISO/IEC 2004 – All rights reserved

Copyright International Organization for Standardization
Reproduced by IHS under license with ISO
No reproduction or networking permitted without license from IHS

Not for Resale