INTERNATIONAL STANDARD

ISO/IEC 18014-2

Second edition
2009-12-15

Information technology — Security techniques — Time-stamping services —

Part 2: Mechanisms producing independent tokens

Technologies de l'information — Techniques de sécurité — Services d'horodatage —
Partie 2: Mécanismes produisant des jetons indépendants

Reference number
ISO/IEC 18014-2:2009(E)

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

Contents

Page

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 18014-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

This second edition cancels and replaces the first edition (ISO/IEC 18014-2:2002). The text has been revised to clarify the presentation, and a new time-stamping mechanism has been added.

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*

Information technology — Security techniques — Time-stamping services —

Part 2: Mechanisms producing independent tokens

1 Scope

This part of ISO/IEC 18014 presents a general framework for the provision of time-stamping services.

Time-stamping services may generate, renew and verify time-stamp tokens.

Time-stamp tokens are associations between data and points in time, and are created in a way that aims to provide evidence that the data existed at the associated date and time. In addition, the evidence may be used by non-repudiation services.

This part of ISO/IEC 18014 specifies mechanisms that generate independent time-stamps: in order to verify an independent time-stamp token, verifiers do not need access to any other time-stamp tokens. That is, time-stamp tokens are not linked, as is the case for the token types defined in ISO/IEC 18014-3.

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 8824-1:1998, *Information technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation*

ISO/IEC 9594-8:2005, *Information technology — Open Systems Interconnection — The Directory: Public-key and attribute certificate frameworks*

ISO/IEC 18014-1:2008, *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
asymmetric key pair
pair of related keys where the private key defines the private transformation and the public key defines the public transformation

NOTE Adapted from ISO/IEC 9798-1.

3.2
asymmetric signature system
system based on asymmetric techniques whose private transformation is used for signing and whose public transformation is used for verification

NOTE Adapted from ISO/IEC 9798-1.

3.3
authentication
provision of assurance of the claimed identity of an entity

NOTE Adapted from ISO/IEC 18028-4.

3.4
certification authority
CA
authority trusted by one or more users to create and assign public-key certificates

NOTE Optionally, the certification authority can create the users' keys.

[ISO/IEC 9594-8:2005, definition 3.3.16]

3.5
cryptography
discipline that embodies principles, means, and mechanisms for the transformation of data in order to hide its information content, prevent its undetected modification and/or prevent its unauthorized use

[ISO/IEC 7498-2:1989, definition 3.3.20]

3.6
data integrity
property that data has not been altered or destroyed in an unauthorized manner

[ISO/IEC 7498-2:1989, definition 3.3.21]

3.7
data origin authentication
corroboration that the source of data received is as claimed

[ISO/IEC 7498-2:1989, definition 3.3.22]

3.8
digital signature
data appended to, or a cryptographic transformation (see cryptography) of, a data unit that allows a recipient of the data unit to prove the source and integrity of the data unit and protect against forgery, e.g. by the recipient

[ISO/IEC 7498-2:1989, definition 3.3.26]

3.9
distinguished encoding rules
DER
encoding rules that may be applied to values of types defined using the ASN.1 notation

NOTE 1 As defined in the introduction to ISO/IEC 18028-4.

NOTE 2 Application of these encoding rules produces a transfer syntax for such values. It is implicit that the same rules are also to be used for decoding. The DER is more suitable if the encoded value is small enough to fit into the available memory and there is a need to rapidly skip over some nested values.

3.10
hash-function
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

NOTE Computational feasibility depends on the specific security requirements and environment.

[ISO/IEC 10118-1:2000, definition 3.5]

3.11
hash-value
string of bits which is the output of a hash-function

NOTE Adapted from hash-code as defined in ISO/IEC 10118-1.

3.12
message authentication code
MAC
string of bits which is the output of a MAC algorithm

NOTE A MAC is sometimes called a cryptographic check value (see for example ISO 7498-2).

[ISO/IEC 9797-1:1999, definition 3.2.4]

3.13
message authentication code (MAC) algorithm
algorithm for computing a function which maps strings of bits and a secret key to fixed-length strings of bits, satisfying the following two properties:

— for any key and any input string the function can be computed efficiently

— for any fixed key, and given no prior knowledge of the key, it is computationally infeasible to compute the function value on any new input string, even given knowledge of the set of input strings and corresponding function values, where the value of the ith input string may have been chosen after observing the value of the first $i-1$ function values

NOTE 1 A MAC algorithm is sometimes called a cryptographic check function (see for example ISO 7498-2).

NOTE 2 Computational feasibility depends on the user's specific security requirements and environment.

[ISO/IEC 9797-1:1999, definition 3.2.6]

3.14
private key
that key of an entity's asymmetric key pair which should only be used by that entity

[ISO/IEC 9798-1:1997, definition 3.3.17]

3.15
public key
that key of an entity's asymmetric key pair which can be made public

NOTE In the case of an asymmetric signature scheme the public key defines the verification transformation. In the case of an asymmetric encipherment system the public key defines the encipherment transformation. A key that is 'publicly known' is not necessarily globally available. The key may only be available to all members of a pre-specified group.

[ISO/IEC 11770-3:2008, definition 3.32]

3.16
public key certificate
public key information of an entity signed by the certification authority and thereby rendered unforgeable

[ISO/IEC 11770-3:2008, definition 3.33]

3.17
time-stamp requester
entity which possesses data it wants to be time-stamped

[ISO/IEC 18014-1:2008, definition 3.14]

3.18
time-stamp token
TST
data structure containing a verifiable cryptographic binding between a data item's representation and a time-value

NOTE A time-stamp token can also include additional data items in the binding.

[ISO/IEC 18014-1:2008, definition 3.15]

3.19
time-stamp verifier
entity which possesses data and wants to verify that it has a valid time-stamp bound to it

NOTE The verification process can be performed by the verifier itself or by a trusted third party.

[ISO/IEC 18014-1:2008, definition 3.16]

3.20
time-stamping authority
TSA
trusted third party trusted to provide a time-stamping service

[ISO/IEC 18014-1:2008, definition 3.17]

3.21
time-stamping policy
named set of rules that indicates the applicability of a time-stamp token to a particular community or class of application with common security requirements

3.22
time-stamping service
TSS
service providing evidence that a data item existed before a certain point in time

[ISO/IEC 18014-1:2008, definition 3.18]

3.23
time-stamping unit
TSU
set of hardware and software which is managed as a unit and generates time-stamp tokens

3.24
trusted third party
security authority, or its agent, trusted by other entities with respect to security related activities

[ISO/IEC 10181-1:1996, definition 3.3.30]

4 Notation, symbols and abbreviated terms

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

∧	logical conjunction, i.e. the *and* operator of Boolean algebra
ASN.1	Abstract Syntax Notation One
CA	certification authority
DER	Distinguished Encoding Rules
$H_i(D)$	hash-value calculated by applying the hash-function H_i to the data string D
HMAC	hash message authentication code
isValid (TST(t), t_V)	predicate (i.e. true or false) indicating whether or not the token TST(t) is valid at time t_V
MAC	message authentication code
OID	object identifier
PKI	public key infrastructure
TSA	time-stamping authority
TSS	time-stamping service
TST	time-stamp token
TST(t)	time-stamp token created at time t
TSU	time-stamping unit
t, t_V	points in time

Hash-functions are specified in the multi-part standard ISO/IEC 10118.

MAC functions are specified in the multi-part standard ISO/IEC 9797.

The HMAC function is specified in ISO/IEC 9797-2.

5 Time-stamping services

Time-stamping services may generate, renew, and verify time-stamp tokens.

Time-stamp tokens are associations between data and points in time, and are created in a way that aims to provide evidence that the data existed at the associated date and time. In addition, the evidence may be used by non-repudiation services.

Time-stamping services involve the following entities (defined in ISO/IEC 18014-1):

— the time-stamp requester, that has a document to time-stamp;

— the Time-Stamping Authority (TSA), that generates time-stamp tokens (TSTs);

— the time-stamp verifier, that verifies time-stamps bound to documents.

A time-stamping service (TSS) provides three specific services:

— time-stamp generation, where the requester submits data items and receives a time-stamp generated by the TSA;

— time-stamp renewal, a special case of time-stamp generation, where the requester submits an existing time-stamp and related data items and receives a new time-stamp generated by the TSA, such that the validity period of the existing time-stamp is extended by the new time-stamp;

— time-stamp verification, in which the verifier validates the time-stamp.

Time-stamping services are provided by means of two protocols, as defined in ISO/IEC 18014-1:

— time-stamp request protocol: the requester requests the TSA to time-stamp a document or renew an existing time-stamp for a document, and

— time-stamp verification protocol: the verifier submits a time-stamp token to be verified.

6 Time-stamp tokens

6.1 Contents

A time-stamp token is a data structure containing a verifiable binding between a data item's representation and a time-value. A time-stamp token may also bind additional items to the data item's representation and the time-value.

A time-stamp token shall contain

— one or more hash-values of the data that is to be time-stamped; hash-functions are specified in the multi-part standard ISO/IEC 10118;

— a point in time (a time-value);

— a reference to the policy under which the time-stamp token is generated.

together with any additional information that may be regarded as helpful for the practical provision of the service, such as

— identification of the time-stamping service provider (to help verifiers in looking for further evidence);

— an indication of the accuracy of the time point (that is, the maximum error in the time representation);

— an indication of ordering (that is, whether the service provider guarantees the relative ordering of generated tokens);

— identification of the version of the format (foreseeing syntax changes in the future);

— a serial number (to enable reference to be made to the token);

— a reference to the user's request[1)], to help users in matching requests and responses.

1) Often referred to as a 'nonce', a number or bit string used only once, so that there is no ambiguity about what it is referring to.

6.2 Notation

Let

$H_i(D)$

be a hash-value computed on data D using hash-function H_i.

Let

$TST(t)$

be a time-stamp token issued at the point in time t.

The time-stamp token $TST(t)$ may be further decomposed into its parts:

$TST(t) := < \{ H_i(D) \}, t , P >$ [2)]

where $\{ H_i(D) \}$ is the set of one or more hash-values[3)] on data D. P indicates the policy under which the token was generated.

6.3 Verification

Let t_V be the moment when the time-stamp token is verified, where t_V is measured by the entity performing the validity check.

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

- the time-stamp token is syntactically well-formed;
- $t < t_V$; [4)]

- the value of every component $H_i(D)$ of the time-stamp token matches the hash-value of D evaluated at t_V over the document subject to scrutiny, using the same hash-function H_i;
- at least one of the hash-functions H_i is not broken at t_V;
- the protection of the time-stamp token is technically sound when the time-stamp token is verified at time t_V; that is, the protection mechanism is not broken;
- the issuing policy P is acceptable for the verifier's purposes.

If all the previous conditions hold, we say that the time-stamp token is valid at t_V. The following notation is used for the predicate that evaluates whether a time-stamp token TST(t) is valid at t_V:

$isValid\ (TST(t), t_V) = true$

The verifier may request additional assurance that is outside the scope of this standard.

---

2) The notation <a, b, c, …> denotes a tuple, that is a sequence of values called the components of the tuple.

3) Each hash-value $H_i(D)$ shall describe both the hash value and the hash-function used to derived it, altogether with any additional information that might be needed for recreate the hash value in the future (e.g. hash-function parameters). Hash-functions are standardised in ISO/IEC 10118. Use of hash-functions chosen from amongst those specified in ISO/IEC 10118 is recommended.

4) The notation $t_1 < t_2$ indicates that 'time t_1 is previous to time t_2' according to the clock of the validating entity. Strict precedence is not always mandatory, and a verifier may specify a tolerance or accepted error margin in time values; if such a tolerance is used, it shall be a positive number, and it shall be stated in the verifier's practice statement. In such a case, the mathematical formula becomes $t_1 - t_2 <$ tolerance.

6.4 Renewal

A time-stamp generated at t_0 is theoretically valid forever. However, in practice, a time limit should apply, for example for one of the following reasons:

— the strength of any of the underlying cryptographic primitives is under suspicion and is no longer trusted;

— the TSA's signing key is about to expire;

— the TSA is about to cease provision of a time-stamping service;

— the policy specifies a time limit that is about to expire.

In such a case, a new time-stamp token is needed to extend the validity beyond the practical limits of the original token. This new token, generated at t_1, may extend the previous bound t_0 if generated using the renewal architecture described below; that is, the new time-stamp token binds the point in time t_0 to the data, and is valid beyond t_1. In general, several time-stamp tokens may be part of a renewal chain

$$[\ TST(t_0)\ \ TST(t_1)\ \ TST(t_2)\ \ \ldots\ \ TST(t_i)\ \ \ldots\ ] \qquad \text{where } t_0 < t_1 < t_1 < \ldots < t_i < \ldots$$

that extends the validity of the binding to t_0 an unlimited number of times.

In order to achieve this objective:

— the new time-stamp token at t_i shall be generated before the previous time-stamp token expires;

— the time-stamp token $TST(t_i)$ incorporates time-stamp token $TST(t_{i-1})$ as part of the protected information;

— the time-stamp request makes explicit the previous time-stamp token so that it can be incorporated into the response;

— the time-stamp verification shall extend over the chain of renewals.

6.5 Renewal verification

Let

$$[\ TST(t_0), TST(t_1), \ldots, TST(t_n)\ ]$$

be a renewal chain, that is, an ordered list of time-stamp tokens:

— which all refer to the same data item D;

— for which the generation time is ordered; that is, $t_0 < t_1 < \ldots < t_n$.

Let t_v be the moment when the time-stamp chain is verified.

The validity extension property states that

$$isValid\ (\ [TST(t_0), TST(t_1)], t_v) = isValid\ (TST(t_0), t_1) \wedge isValid(\ TST(t_1), t_v)$$

where

$$t_0 < t_1 < t_v$$