INTERNATIONAL STANDARD

ISO/IEC 18014-2

Third edition
2021-09

Information security — Time-stamping services —

Part 2: Mechanisms producing independent tokens

Sécurité de l'information — Services d'horodatage —

Partie 2: Mécanismes produisant des jetons indépendants

Reference number
ISO/IEC 18014-2:2021(E)

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives or www.iec.ch/members_experts/refdocs).

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. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

This third edition cancels and replaces the second edition (ISO/IEC 18014-2:2009), which has been technically revised.

The main changes compared to the previous edition are as follows:

— updated the definition of a hash function to a collision-resistant hash-function;

— application of style and editorial changes.

A list of all parts in the ISO/IEC 18014 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html and www.iec.ch/national-committees.

Information security — Time-stamping services —

Part 2:
Mechanisms producing independent tokens

1 Scope

This document specifies mechanisms that generate, renew, and verify independent time-stamps. In order to verify an independent time-stamp token, time-stamp verifiers do not need access to any other time-stamp tokens. That is, such time-stamp tokens are not linked.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements 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 18014-1, *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.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at https://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
time-stamp token
TST
data structure containing a verifiable binding between a data items' representation and a time-value

[SOURCE: ISO/IEC 18014-1:2008, 3.15, modified – Note to entry has been removed.]

3.2
time-stamping service
TSS
service providing evidence that a data item existed before a certain point in time

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

3.3
time-stamping policy
set of rules that indicates the applicability of a *time-stamp token* (3.1) to a particular community and/or class of application with common security requirements

[SOURCE: ISO/IEC 18014-1:2008, 3.23]

3.4
time-stamp requester
entity which possesses data it wants to be time-stamped

[SOURCE: ISO/IEC 18014-1:2008, 3.14, modified – Note to entry has been removed.]

3.5
time-stamp verifier
entity which possesses data and wants to verify that it has a valid time-stamp bound to it

Note 1 to entry: The verification process can be performed by the verifier itself or by a trusted third party.

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

3.6
time-stamping authority
TSA
trusted third party trusted to provide a *time-stamping service* (3.2)

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

3.7
time-stamping unit
TSU
set of hardware and software which is managed as a unit and generates *time-stamp tokens* (3.1)

3.8
data origin authentication
corroboration that the source of data received is as claimed

Note 1 to entry: Data origin is sometimes called data source.

[SOURCE: ISO 7498-2:1989, 3.3.22, modified — In the definition, the initial article "the" has been removed. Note 1 to entry has been added.]

3.9
data integrity
property that data has not been altered or destroyed in an unauthorized manner

[SOURCE: ISO/IEC 9797-1:2011, 3.4]

3.10
asymmetric key pair
pair of related keys where the *private key* (3.11) defines the private transformation and the *public key* (3.12) defines the public transformation

[SOURCE: ISO/IEC 9798-1:2010, 3.3]

3.11
private key
key of an entity's *asymmetric key pair* (3.10) that is kept private

Note 1 to entry: The security of an *asymmetric signature system* (3.15) depends on the privacy of this key.

[SOURCE: ISO/IEC 11770-1:2010, 2.35, modified — The definition was restricted to asymmetric signature system.]

3.12
public key
key of an entity's *asymmetric key pair* (3.10) which can usually be made public without compromising security

[SOURCE: ISO/IEC 11770-1:2010, 2.36]

3.13
public key certificate
public key (3.12) information of an entity signed by the certification authority

[SOURCE: ISO/IEC 11770-1:2010, 2.37]

3.14
public-key infrastructure
PKI
infrastructure able to support the management of public keys able to support authentication, encryption, integrity or non-repudiation services

[SOURCE: ISO/IEC 9594-8:2020, 3.5.60, modified — In the definition, the initial article "the" has been removed.]

3.15
asymmetric signature system
system based on asymmetric cryptographic techniques whose private transformation is used for signing and whose public transformation is used for verification

[SOURCE: ISO/IEC 9798-1:2010, 3.4]

3.16
digital signature
data appended to, or a cryptographic transformation of, a data unit that allows a recipient of the data unit to prove the *source* (3.8) and *integrity* (3.9) of the data unit and protect against forgery, e.g. by the recipient

[SOURCE: ISO/IEC 9798-1:2010, 3.11]

3.17
collision-resistant hash-function
hash-function satisfying the following property: it is computationally infeasible to find any two distinct inputs which map to the same output

[SOURCE: ISO/IEC 10118-1:2016, 3.1, modified — Note 1 to entry has been removed.]

3.18
hash-code
string of bits which is the output of a *collision-resistant hash-function* (3.17)

Note 1 to entry: The definition in ISO/IEC 10118-1 does not require collision-resistance. In this document, all hash functions are collision-resistant hash functions.

[SOURCE: ISO/IEC 10118-1:2016, 3.3, modified — In the definition, "collision-resistant" has been added. Note 1 to entry has been replaced.]

3.19
message authentication code algorithm
MAC algorithm
algorithm for computing a function which maps strings of bits and a *MAC algorithm key* (3.20) to fixed-length strings of bits, satisfying the following two properties:

— for any *MAC algorithm key* (3.20) and any input string, the function can be computed efficiently;

— for any fixed *MAC algorithm key* (3.20), and given no prior knowledge of the *MAC algorithm key* (3.20), it is computationally infeasible to compute the function value on any new input string, even given knowledge of a set of input strings and corresponding function values, where the value of the i-th input string might have been chosen after observing the value of the first $i - 1$ function values

[SOURCE: ISO/IEC 9797-1:2011, 3.10, modified — In the definition, "secret key" has been replaced with "MAC algorithm" and "key" has been replaced with "MAC algorithm key". At the end, the parentheses have been removed.]

3.20
MAC algorithm key
key that controls the operation of a *MAC algorithm* (3.19)

[SOURCE: ISO/IEC 9797-1:2011, 3.8]

3.21
message authentication code
MAC
string of bits which is the output of a *MAC algorithm* (3.19)

[SOURCE: ISO/IEC 9797-1:2011, 3.9, modified — Note 1 to entry has been removed.]

3.22
distinguished encoding rules
DER
encoding rules that may be applied to values of types defined using the ASN.1 notation

Note 1 to entry: 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.

4 Notation, symbols and abbreviated terms

ASN.1	abstract syntax notation one Annex A provides the ASN.1 definitions described in this document, which shall be used to identify the OIDs of ASN.1 module for time-stamping.
H_i	collision-resistant hash-function
$H_i(X)$	hash-code computed on data X
isValid(TST(t), t_v)	predicate (i.e. True or False) indicating whether or not the token TST(t) is valid at point in time t_v
OID	object identifier
TST(t)	time-stamp token created at point in time t
t, t_v	points in time
$t_1 < t_2$	The time-stamp verifier considers t_1 to be an earlier point in time than t_2. Strict precedence is not always mandatory, and a time-stamp verifier may specify a tolerance or accepted error margin in time units. When such tolerance is permitted, the allowed value ε shall be a positive number, and it shall be stated in the time-stamp verifier's practice statement. In such a case, the time-stamp verifier shall accept t_1 as being earlier than t_2 as long as no more than ε time units have elapsed from t_2 to t_1.
<*a*, *b*, *c*>	a triplet, that is a sequence of values called the components of the triplet
$\wedge$	logical conjunction, i.e. the *and* operator of Boolean algebra

5 Time-stamp tokens

5.1 Contents

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

A time-stamp token shall contain:

— one or more hash-codes of the data that is to be time-stamped;

— a point in time;

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

together with any additional information that can be regarded as helpful for the practical provision of the time-stamping service, such as:

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

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

— an indication of ordering (that is, whether the time-stamping authority 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.

5.2 Generation

Let $\mathrm{H_i}(D)$ be a hash-code computed on data D using a collision-resistant hash-function $\mathrm{H_i}$.

The time-stamp token TST(t) consists of the triplet

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

where $\{ \mathrm{H_i}(D) \}$ is a set of one or more hash-codes on data D. P indicates the time-stamping policy under which the time-stamping token was generated. Each hash-code $\mathrm{H_i}(D)$ shall describe both the hash-code and the collision-resistant hash-function used to derive it, together with any additional information that is needed to recreate the hash-code in the future (e.g. collision-resistant hash-function parameters).

NOTE Collision-resistant hash-functions are standardized in the ISO/IEC 10118 series.

5.3 Verification

Let t_v be the point in time when the time-stamp token is verified, where t_v is measured by the time-stamp verifier.

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

— the time-stamp token t is syntactically well-formed;

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

— $t<t_v$;

— the value of every component in { $H_i(D)$ } of the time-stamp token matches the hash-code of D evaluated at t_v over the data subject to scrutiny, using the same collision-resistant hash-function H_i together with any additional information that was used to generate the hash-code;

— the issuing time-stamping policy P is acceptable for the time-stamp verifier's purposes.

If all these conditions hold, it is said that the time-stamp token is valid at t_v. Otherwise, the time-stamp token is said to be invalid. 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   if the time-stamp token is valid;

isValid (TST(t), t_v) = False otherwise.

The time-stamp verifier may request additional assurances that are outside the scope of this document.

5.4 Renewal

A time-stamp generated at t_0 is theoretically valid indefinitely. However, in practice, a time limit (i.e. a point in time after which a token generated at t_0 is no longer valid) should apply, for example for one of the following reasons:

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

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

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

— the time-stamping policy specifies a point in time limit after which the time-stamp expires.

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

[TST(t_0), TST(t_1), TST(t_2), ..., TST(t_i), ...]

where $t_0 < t_1 < t_2 < ... < t_i < ...$.

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

— the time-stamp token TST(t_i) shall incorporate the time-stamp token TST(t_{i-1}) as part of the protected information.

5.5 Renewal verification

Let [TST(t_0), TST(t_1), ..., TST(t_n)] be a renewal chain, i.e. 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, $t_0 < t_1 < ... < t_n$.

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

