INTERNATIONAL STANDARD

ISO/IEC 10118-1

Third edition
2016-10-15

Information technology — Security techniques — Hash-functions —

Part 1:
General

Technologies de l'information — Techniques de sécurité — Fonctions de hachage —

Partie 1: Généralités

Reference number
ISO/IEC 10118-1:2016(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

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.

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).

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).

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

For an explanation on 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 the following URL: www.iso.org/iso/foreword.html.

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

This third edition cancels and replaces the second edition (ISO/IEC 10118-1:2000), which has been technically revised.

A list of all parts in the ISO/IEC 10118 series can be found on the ISO website.

Information technology — Security techniques — Hash-functions —

Part 1: General

1 Scope

ISO/IEC 10118 (all parts) specifies hash-functions and is therefore applicable to the provision of authentication, integrity and non-repudiation services. Hash-functions map strings of bits of variable (but usually upper bounded) length to fixed-length strings of bits, using a specified algorithm. They can be used for

— reducing a message to a short imprint for input to a digital signature mechanism, and

— committing the user to a given string of bits without revealing this string.

NOTE The hash-functions specified in ISO/IEC 10118 (all parts) do not involve the use of secret keys. However, these hash-functions may be used, in conjunction with secret keys, to build message authentication codes. Message Authentication Codes (MACs) provide data origin authentication as well as message integrity. Techniques for computing a MAC using a hash-function are specified in ISO/IEC 9797-2 [1].

This document contains definitions, symbols, abbreviations and requirements that are common to all the other parts of ISO/IEC 10118. The criteria used to select the algorithms specified in subsequent parts of ISO/IEC 10118 are defined in Annex B of this document.

2 Normative references

There are no normative references in this document.

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:

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

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

3.1
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

Note 1 to entry: Computational feasibility depends on the specific security requirements and environment. Refer to Annex C.

3.2
data string
data
string of bits which is the input to a hash-function

3.3
hash-code
string of bits which is the output of a hash-function

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as hash-code. Modification Detection Code, Manipulation Detection Code, digest, hash-result, hash-value and imprint are some examples.

3.4
hash-function
function which maps strings of bits of variable (but usually upper bounded) length to fixed-length strings of bits, satisfying the following two properties:

— for a given output, it is computationally infeasible to find an input which maps to this output;

— for a given input, it is computationally infeasible to find a second input which maps to the same output

Note 1 to entry: Computational feasibility depends on the specific security requirements and environment. Refer to Annex C.

3.5
initializing value
value used in defining the starting point of a hash-function

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as initializing value. Initialization vector and starting value are examples.

3.6
output transformation
transformation or mapping of the output of the iteration stage to obtain the hash-code

3.7
padding
appending extra bits to a data string

3.8
round-function
function $\phi(.,.)$ that transforms two binary strings of lengths L_1 and L_2 to a binary string of length L_2 that is used iteratively as part of a hash-function, where it combines a data string of length L_1 with the previous output of length L_2 or the initializing value

Note 1 to entry: The literature on this subject contains a variety of terms that have the same or similar meaning as round-function. Compression function and iterative function are some examples.

4 Symbols and abbreviated terms

4.1 General symbols

For ISO/IEC 10118 (all parts), the following symbols and abbreviations are used:

B_i	a byte
D	data
D_i	a block derived from the data string D after the padding process
h	hash-function

H hash-code

H_i a string of L_2 bits which is used in the hashing operation to store an intermediate result

IV initializing value

L_1 length (in bits) of the first of the two input strings to the round-function

L_2 length (in bits) of the second of the two input strings to the round-function, the output string from the round-function, and of the initializing value

L_X length (in bits) of a string of bits X

ϕ round-function (phi)

T an output transformation function, e.g. truncation

$X \,||\, Y$ concatenation of strings of bits X and Y in the indicated order

$X \oplus Y$ exclusive-or of strings of bits X and Y (where $L_X = L_Y$)

4.2 Symbols specific to this document

For the purpose of this document, the following symbol applies:

q number of blocks in the data string after the padding and splitting process

4.3 Coding conventions

In contexts where the terms “most significant bit/byte” and “least significant bit/byte” have a meaning (e.g. where strings of bits/bytes are treated as numerical values), the leftmost bits/bytes of a block shall be the most significant.

5 Requirements

The use of a hash-function requires that the parties involved shall operate upon precisely the same bit string, even though the representation of the data may be different in each entity’s environment. This may require one or more of the entities to convert the data into an agreed bit-string representation prior to applying a hash-function.

Some of the hash-functions specified in ISO/IEC 10118 (all parts) require padding, so that the data string is of the required length. Several padding methods are presented in Annex A of this document; additional padding methods may be specified in each part of ISO/IEC 10118 where padding is needed.

6 General model for hash-functions

6.1 General

The hash-functions specified in ISO/IEC 10118 (all parts) require the use of a round-function ϕ. In subsequent parts of ISO/IEC 10118, several alternatives for the function ϕ are specified.

The hash-functions which are specified in subsequent parts of ISO/IEC 10118 provide hash-codes of length L_H, where L_H is less than or equal to the value of L_2 for the round-function ϕ being used.

6.2 Hashing operation

6.2.1 General

Let ϕ be a round-function and IV be an initializing value of length L_2. For the hash-functions specified in subsequent parts of ISO/IEC 10118, the value of the IV shall be fixed for a given hash-function ϕ. The hash-code H of the data D shall be calculated using the following four steps.

6.2.2 Step 1 (padding)

The data string D is padded in order to ensure that its length is an integer multiple of L_1. See Annex A for more information.

6.2.3 Step 2 (splitting)

The padded version of the data string D is split into L_1-bit blocks $D_1, D_2, \ldots, D_q$, where D_1 represents the first L_1 bits of the padded version of D, D_2 represents the next L_1 bits, and so on. The padding and splitting processes are illustrated in Figure 1.

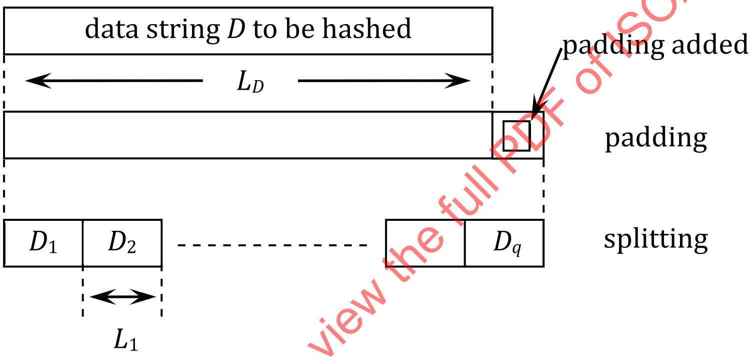

Figure 1 — Padding and splitting processes

NOTE Sometimes, it is more efficient to have the splitting occur before the padding. The padding is then done on the last block.

6.2.4 Step 3 (iteration)

Let $D_1, D_2, \ldots, D_q$ be the L_1-bit blocks of the data after padding and splitting. Let H_0 be a bit string equal to IV. The L_2-bit strings $H_1, H_2, \ldots, H_q$ are calculated iteratively in the following way.

```
for $i$ from 1 to $q$:
    $H_i = \phi(D_i, H_{i-1})$.
```

6.2.5 Step 4 (output transformation)

The hash-code H is derived by performing a transformation T on H_q, the output of step 3, to obtain the L_H bits of the final hash-code.

EXAMPLE The transformation T may be a truncation operation.

6.3 Use of the general model

In subsequent parts of ISO/IEC 10118, examples of hash-functions based on the general model are specified. Specification of an individual hash-function will in each case require the following to be defined:

— parameters L_1, L_2;

— the padding method;

— the initializing value IV;

— the round-function ϕ;

— the output transformation T.

Practical use of a hash-function defined using the general model will also require the choice of the parameter L_H.

Annex A
(normative)

Padding methods

A.1 General

The calculation of a hash-code, as specified in other parts of ISO/IEC 10118, may require the selection of a padding method. The padding method will always output a padded data string whose length (in bits) is a multiple of L_1. Two methods are presented in this annex.

The padding bits (if any) need not be stored or transmitted with the data. The verifier shall know whether or not the padding bits have been stored or transmitted, and which padding method is in use.

A.2 Method 1

The data for which the hash-code is to be calculated is appended with a single "1" bit. The resulting data are then appended with as few (possibly zero) "0" bits as are necessary to obtain the required length.

NOTE Method 1 always requires the addition of at least one padding bit.

A.3 Method 2

This padding method requires the selection of a parameter r (where $r \leq L_1$), e.g. $r = 64$, and a method for encoding the bit length of the data D, i.e. L_D, as a bit string of length r. The choice for r will limit the length of D, in that $L_D < 2^r$.

The data D for which the hash-code is to be calculated is padded using the following procedure.

a) D is concatenated with a single "1" bit.

b) The result of the previous step is concatenated with between zero and $L_1 - 1$ "0" bits, such that the length of the resultant string is congruent to $L_1 - r$ modulo L_1. The result will be a bit string whose length will either be r bits short of an integer multiple of L_1 bits, or in the case of $r = L_1$, will be a bit string whose length is an exact multiple of L_1 bits.

c) Append an r-bit encoding of L_D, using the selected encoding method, yielding the padded version of D.

Annex B
(normative)

Criteria for submission of hash-functions for possible inclusion in ISO/IEC 10118 (all parts)

B.1 Guidelines used for selecting hash-functions

The hash-functions included in subsequent parts of ISO/IEC 10118 have been selected from the large variety of such techniques published and in use. The exclusion of particular hash-functions does not necessarily imply that these techniques are insecure. The hash-functions specified represent a small set of techniques chosen according to the following criteria (where the order of presentation of the criteria is not of significance). SC 27/WG 2 Standing Document 5 (WG2 SD5) describes a process that ISO/IEC JTC 1/SC 27 can follow when deciding on the inclusion of new hash-functions in ISO/IEC 10118 (all parts).

Selection is made with respect to the following aspects of the hash-function.

a) The *security* of the hash-function, i.e. selected algorithms shall be resistant to cryptanalytic attacks. Attack goals, generic attacks and impact of cryptanalytic attacks are defined in Annex C of this document. The existence of a proof of security or security reduction is regarded as a significant argument in favour of a hash-function, depending on the security model and the proof assumptions. The nature of any evaluations is also of great importance, especially those conducted by widely recognized evaluation organizations.

b) The *performance* of the hash-function on a variety of typical platforms. This includes not only issues such as time and space efficiency, but also whether or not it has characteristics that give it advantages over other techniques.

c) The nature of any *licensing issues* affecting the hash-function.

d) The *maturity* of the hash-function. The *maturity* of the hash-function is evaluated in terms of how extensively it is used, how widely any analysis has been published, and how much the hash-function has been scrutinized. National standards are considered to have high maturity.

e) The degree to which the hash-function is *endorsed* by a recognized organization (e.g. a standards body, government security agency, etc.), or is under investigation and/or analysis for endorsement by such a body.

f) The existing *level of adoption* of the hash-function. Unless other considerations over-ride such a decision, hash-functions that are de facto standards are to be favoured over less well-used techniques.

g) In general, the *number* of hash-functions to be standardized in each part of ISO/IEC 10118 should be as small as possible. Three exceptions to this principle exist.

   1) Where two hash-functions have different characteristics, e.g. hash-functions with different hash-code lengths or hash-functions with widely differing computational and space implementation requirements, and both sets of characteristics have practical significance, hash-functions of both types are likely to be standardized.

   2) It is generally desirable to have multiple standardized hash-functions which are based on different fundamental principles, so that if one hash-function becomes vulnerable to cryptanalytic attack, another hash-function has a good chance of remaining secure.

3) It is generally desirable to have standardized hash-functions with large security margins for an arbitrary application's security level.

B.2 Minimum qualification criteria for the submission of new hash-functions

The criteria set out in Clause B.2 are meant for the submission of hash-functions not already included in subsequent parts of ISO/IEC 10118. In order for a hash-function to be considered for inclusion in subsequent parts of ISO/IEC 10118, the hash-function shall comply with the following requirements.

a) Cryptanalysis is performed and results are known: There shall be no known cryptanalytic attacks that break the hash-function. Annex C provides further information on attacks as related to hash-functions.

b) Public domain: The hash-function description shall have been published for a minimum period of 3 years in the public domain. Examples of acceptable publications and conferences for hash-function presentation include but are not limited to the following:

   1) IACR conferences and workshops:

      i) International Conference on the Theory and Application of Cryptology and Information Security (Asiacrypt)

      ii) International Cryptology Conference (Crypto)

      iii) International Conference on the Theory and Applications of Cryptographic Techniques (Eurocrypt)

      iv) International Workshop on Fast Software Encryption (FSE)

      v) International Workshop on Cryptographic Hardware and Embedded Systems (CHES)

   2) IEEE annual conferences:

      i) Symposium on Security and Privacy

      ii) Symposium on the Foundations of Computer Science (FOCS)

   3) ACM annual conferences

      i) Symposium on Theory of Computing (ACM-STOC)

      ii) Computer and Communication Security (ACM-CCS)

   4) Well-known international conferences which have a more than 15-year history with available proceedings:

      i) USENIX Security

      ii) European Symposium on Research in Computer Security (ESORICS)

      iii) Australasian Conference on Information Security and Privacy (ACISP)

      iv) Financial Cryptography and Data Security (FC)

      v) International Conference on Information Security and Cryptography (ICISC)

      vi) Conference on Selected Areas in Cryptography (SAC)

   5) Well-known journals:

      i) ACM

         I) Journal of the ACM

II) Communications of the ACM

ii) Elsevier

I) Computer Communications

II) Information and Computation

III) Journal of Computer and System Sciences (JCSS)

IV) Journal of Discrete Algorithms

iii) IEEE

I) IEEE Transactions on Information Theory

II) IEEE Transactions on Computers

III) IEEE Security & Privacy

iv) IEICE

I) IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences

II) IEICE Transactions on Information and Systems

v) SIAM Journal on Computing

vi) Springer

I) Combinatorica

II) Cryptography and Communications

III) Designs, Codes and Cryptography

IV) Journal of Cryptology

V) International Journal of Information Security

6) Official publication as a standard in English that is made available to the general public by a recognized standardization organization.

7) An international competition with the sole purpose of choosing a new state-of-the-art hash-function which is run for a minimum of 2 years, and where analysis and publications are open to the general public. Submissions to this international competition can be considered as published.

c) Cryptanalysis documentation exists: Prior to inclusion, a hash-function shall have cryptanalysis papers published in peer-reviewed journals or conferences such as those listed in b).

d) Industry adoption: Robust evidence shall be provided of commercial applications using the hash-function and possible world-wide deployments of the applications.

e) Performance: For a pre-determined length of the hash-code, performance measurements shall be provided for many different vectors such as bits/cycle or bits/watt. Robust evidence shall be provided that the hash-function offers acceptable performance on the performance vectors that are optimized for the intended applications compared to existing hash-functions already in the standard.

Annex C
(informative)

Security considerations

C.1 Attack goals

Various attack goals exist related to hash-functions (see, for example, Reference [2]). The following are of particular importance.

Collision search — The attack goal is to find two different data strings M_1, M_2 such that $h(M_1) = h(M_2)$.

Preimage search — Given a bit string H of appropriate length, the attack goal is to find a data string M such that $h(M) = H$.

Second preimage search — Given a data string M, the attack goal is to find another data string M' such that $h(M') = h(M)$ and $M' \neq M$.

Length-extension attack — Given a bit string $h(M)$ for some unknown non-empty data string M, the attack goal is to find any data string M' and the value of $h(M \,||\, M')$.

NOTE The current state-of-the-art in cryptanalysis of hash-functions involves the use of a large variety of attack goals including some not listed above. These attack goals are taken into consideration during the standardization process, but have auxiliary importance. Moreover, in applications, it is not usually required that the hash-function is resistant to all attack goals. Typically, some particular subset of those requirements is only considered.

C.2 Generic attacks

A generic attack is an attack that is applicable to all hash-functions and does not rely on the design of the hash-function.

EXAMPLE One example of a generic attack is a brute force search for a preimage. Given a value of a hash-code, the attacker evaluates the values of $h(M)$, trying different possible data strings M and comparing the resulting hash-code with the given one. If the two hash-codes match, the preimage search goal is achieved.

C.3 Impact of cryptanalytic attack

In ISO/IEC 10118 (all parts), the resistance of a hash-function to possible attack goals is specified in terms of “computational infeasibility” of meeting the goal. As noted in the definitions, the meaning of computational infeasibility depends on the specific security requirements and environment. One meaning commonly used by security practitioners is that the computational infeasibility is a property for a task demanding an amount of computing resource beyond what is generally available.

A more rigorous approach is to compare the efficiency of a particular attack against generic attacks with the same attack goal. If all known cryptanalytic attacks for a given attack goal are not more efficient than the corresponding generic attacks, then the hash-function is said to be resistant to that attack goal. However, if there is a cryptanalytic attack substantially more efficient than the corresponding generic attacks, then the hash-function is said to be broken.

The efficiency of a cryptanalytic attack is determined by three parameters: attack complexity, amount of required memory and probability of success.