INTERNATIONAL STANDARD

ISO/IEC 9796-2

First edition
1997-09-01

Information technology — Security techniques — Digital signature schemes giving message recovery —

Part 2:
Mechanisms using a hash-function

Technologies de l'information — Techniques de sécurité — Schémas de signature numérique rétablissant le message —

Partie 2: Mécanismes utilisant une fonction de hachage

Reference number
ISO/IEC 9796-2:1997(E)

Contents

Page

Annexes

ISO/IEC Copyright Office • Case postale 56 • CH-1211 Genève 20 • Switzerland

Printed in Switzerland

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. 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.

ISO/IEC 9796 consists of the following parts, under the general title *Information technology — Security techniques — Digital signature schemes giving message recovery.*

— *Part 1: Mechanisms using redundancy*

— *Part 2: Mechanisms using a hash-function*

International Standard ISO/IEC 9796-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT security techniques*.

Annexes A and B of this part of ISO/IEC 9796 are for information only.

Introduction

Digital signatures can be used to provide the electronic equivalent of a handwritten signature, for implementing services such as message authentication, message integrity and non-repudiation. These services apply to digital messages which are strings of bits, e.g., concatenations of data elements or concatenations of data objects.

Most digital signature schemes are based upon a particular public-key system. Any public-key system includes three basic operations:

- — a process producing pairs of keys, a private key and a public key,
- — a process using a private key,
- — a process using a public key.

In any public-key digital signature scheme, the private key is used in a signature process for signing messages; the public key is used in a verification process for verifying signed messages. A pair of signature keys thus consists of a "private signature key" and a "public verification key".

There are two types of digital signature schemes.

- — When the whole message or a part of the message may be recovered from the signature, the scheme is named a "digital signature scheme giving message recovery" (see ISO/IEC 9796).
- –– When the whole message has to be stored and transmitted along with the signature, the scheme is named a "digital signature scheme with appendix" (see ISO/IEC 14888).

NOTE — The two types of schemes are not mutually exclusive. Specifically, any scheme giving message recovery, e.g., ISO/IEC 9796:1991, can be used for provision of digital signatures with appendix. In this case, the signature is produced by applying the scheme to a hash-code computed from the message.

ISO/IEC 9796 specifies digital signature schemes giving either total or partial message recovery. One practical advantage of such schemes is that, depending upon the implementation environment, they may reduce data storage and transmission costs.

ISO/IEC 9796:1991 will be transformed into part 1. It specifies a digital signature scheme for messages of limited length, so that the message is completely recovered from the signature. The scheme does not involve a hash-function; it is designed so that, depending on the choice of parameters, the verification process can be extremely efficient, i.e., it can be implemented in environments with very limited processing capabilities.

This part of ISO/IEC 9796 uses a hash-function for hashing the entire message. If the message is short enough, then the verification process recovers the entire message and the hash-code from the signature. Otherwise, the verification process recovers only a part of the message along with the hash-code.

Information technology — Security techniques — Digital signature schemes giving message recovery —

Part 2: Mechanisms using a hash-function

1 Scope

This part of ISO/IEC 9796 specifies a digital signature scheme for messages of any length. The messages are arbitrary strings of bits; no assumption is made as to the possible presence of natural redundancy.

The scheme uses a hash-function and a public-key system. It includes

— a signature process using successively a hash-function, format mechanisms and a signature function for signing messages,

— a verification process using successively a verification function, recovery mechanisms and a hash-function for checking signed messages.

The input to the signature function includes both the hash-code and all or part of the message. The verification function yields as an output the values input to the signature function. By this means, the scheme provides either total or partial message recovery.

This part of ISO/IEC 9796 does not specify the hash-function to be used. ISO/IEC 10118 specifies hash-functions appropriate for digital signatures.

This part of ISO/IEC 9796 does not specify the public-key system used to construct the signature and verification functions. Annex A gives an example of a suitable public-key system, including a key production process, a signature function and a verification function; this public-key system is currently the only one known appropriate for this part of ISO/IEC 9796. Annex B provides illustrative examples related to the operations specified in annex A.

Some parameters affect the security of the scheme. This part of ISO/IEC 9796 does not specify the values to be used to reach a given level of security; however, it is specified in such a way as to minimize the required changes in its use if some of these parameters need to be modified.

2 Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO/IEC 9796. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO/IEC 9796 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references, the latest edition of the normative document referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

ISO/IEC 9796:1991, *Information technology — Security techniques — Digital signature scheme giving message recovery*.

ISO/IEC 9798-1:1997, *Information technology — Security techniques — Entity authentication — Part 1: General.*

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

ISO/IEC 14888 (all parts)[1], *Information technology — Security techniques — Digital signatures with appendix.*

[1] To be published.

3 Terms and definitions

For the purposes of this part of ISO/IEC 9796, the following terms and definitions apply.

3.1 keys

3.1.1
private signature key
private key which defines the private signature transformation

[ISO/IEC 9798-1:1997, definition 3.3.18]

3.1.2
public verification key
public key which defines the public verification transformation

[ISO/IEC 9798-1:1997, definition 3.3.23]

3.2
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.

[ISO/IEC 10118-1:1994, definition 2.4]

3.2.1
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.

[ISO/IEC 10118-1:1994, definition 2.1]

3.3 strings of bits

3.3.1
message
string of bits of any length, possibly empty

NOTE — Adapted from ISO/IEC 9796:1991, definition 2.1.

3.3.1.1
non-recoverable part
part of the message stored and transmitted along with the signature, empty when the recovery is total

3.3.1.2
recoverable part
part of the message conveyed in the signature, empty when the message itself is empty

3.3.2
intermediate string
string of bits from which the recoverable string is derived

3.3.2.1
header
leftmost field of the intermediate string (see table 1)

3.3.2.2
more-data bit
specific bit of the intermediate string, indicating whether the recovery is total or partial (see table 1)

3.3.2.3
padding field
field of the intermediate string, conveying the padding bits (see table 1)

3.3.2.3.1
border bit
rightmost bit of the padding field

3.3.2.4
data field
field of the intermediate string, conveying the recoverable part (see table 1)

3.3.2.5
hash field
field of the intermediate string, conveying the hash-code (see table 1)

3.3.2.6
trailer
rightmost field of the intermediate string (see table 1)

3.3.2.6.1
hash-function identifier
optional byte of the trailer, identifying a hash-function

3.3.3
nibble
block of four consecutive bits

3.3.3.1
border nibble
specific nibble of the intermediate string, containing the border bit

3.3.4
recoverable string
string of bits derived from the intermediate string, input to the signature function

3.3.5
signature
string of bits resulting from the signature process

[ISO/IEC 9796:1991, definition 2.2]

3.3.6 coding conventions

3.3.6.1
bit numbering
In all strings of x bits, the numbering starts from the rightmost bit b_1 (the last bit) to the leftmost bit b_x (the first bit).

3.3.6.2
bit significance
All integers are written with the most significant digit (or bit or nibble or byte) in the leftmost position.

NOTE — For example, for computing a byte value, b_1 is the least significant bit and b_8 is the most significant bit.

4 Symbols and abbreviations

For the purposes of this part of ISO/IEC 9796, the following symbols and abbreviations apply.

H hash-code computed from the message to be signed

H' hash-code recovered from the signature

H'' hash-code computed from the recovered message

k length in bits of Si, Sr, Σ, Sr' and Si', equal to the size of a public parameter of the public-key system

k_h length of H in bits

k_h' length of H' in bits

k_m length of M in bits ($k_m = k_r + 8x$), possibly null

k_r length of Mr in bits

M message to be signed during the signature process

M' message recovered during the verification process

Mn non-recoverable part of M

Mr recoverable part of M

Mr' part of M' recovered from the signature

Si intermediate string, built during the signature

Si' intermediate string recovered during the verification

Sr recoverable string, input to the signature function

Sr' recovered string, output of the verification function

Σ signature

t length of the trailer in bytes ($8t$ bits)

x length of Mn in bytes ($8x$ bits)

'XY' hexadecimal notation, equal to XY to the base 16, with the digits 0 to 9 and A to F

For the purposes of this part of ISO/IEC 9796, the following functions apply.

Hash hash-function which maps M to H

Sign signature function which maps Sr to Σ under the control of the private signature key

Verif verification function which maps Σ to Sr' under the control of the public verification key

5 Requirements

The message to be signed M is a binary string of any length, possibly empty.

— If M is sufficiently short (see 6.3.2.1), then the recovery will be total because M shall be entirely included in the signature.

— If M is too long (see 6.3.2.2), the recovery will be partial because M shall be divided into
- the recoverable part, a string of bits of limited length to be included in the signature,
- the non-recoverable part, a string of bytes of any length to be stored and transmitted along with the signature.

At the beginning of the signature process, the message to be signed M shall be hashed. At the end of the verification process, the recovered message M' shall be hashed.

— In the case of partial recovery, the hash-function shall be collision-resistant.

— In the case of total recovery, collision-resistance is not required.

When a digital signature mechanism uses a hash-function, there shall be a binding between the signature mechanism and the hash-function in use. Without such a binding, an adversary might claim the use of a weak hash-function (and not the actual one) and thereby forge a signature. There are various ways to accomplish this binding. The following options are listed in order of increasing risk.

1. — **Require a particular hash-function when using a particular signature mechanism** — The verification process shall exclusively use that particular hash-function. ISO/IEC 14888-3 gives an example of this option where the DSA mechanism requires the use of SHA-1.

2. — **Allow a set of hash-functions and explicitly indicate in every signature the hash-function in use by an identifier included as part of the signature calculation** — The hash-function identifier is an extension of the hash-code: it indicates how to derive the hash-code. The verification process shall exclusively use the hash-function indicated by the identifier in the signature. This part of ISO/IEC 9796 gives an example of this option; see 6.3.1.

3. — **Allow a set of hash-functions and explicitly indicate the hash-function in use in the certificate domain parameters** — Inside the certificate domain, the verification process shall exclusively use the hash-function indicated in the certificate. Outside the certificate domain, there is a risk due to unrigorous certification authorities. If other certificates may be created, then other signatures may be created. Then the attacked user would be in a dispute situation with the certification authority that produced the other certificate.

4. — **Allow a set of hash-functions and indicate the hash-function in use by some other method, e.g., an indication in the message or a bilateral agreement** — The verification process shall exclusively use the hash-function indicated by the other method. However,

there is a risk that an adversary may forge a signature using another hash-function.

The user of a digital signature mechanism should conduct a risk assessment considering the costs and benefits of the various alternatives. This assessment includes the cost associated with the possibility of a bogus signature being produced.

The signature function uses a private signature key, while the verification function uses the corresponding public verification key.

— Each signing entity shall use and keep secret its own private signature key corresponding to its public verification key.

— Each verifying entity should know the public verification key of the signing entity.

A signed message consists of either the signature alone in the case of total recovery, or the non-recoverable part of the message together with the signature in the case of partial recovery. A signature shall be accepted if and only if the verification process specified in clause 7 is successful.

NOTES

1 This part of ISO/IEC 9796 does not specify key management.

2 This part of ISO/IEC 9796 does not specify message structure. One example of structured message is the concatenation of a registered identification number, a certificate expiration date, a certificate serial number and a public modulus. Then the signature scheme produces a public-key certificate which consists of

- either the signature alone in the case of total recovery,
- or the signature together with the non-recoverable part of the message in the case of partial recovery.

3 The hash-function may limit the message length, e.g., to less than 2^{64} bits.

4 Typical values of k_h are 64 and 80 in the case of total recovery, 128 and 160 in the case of partial recovery.

6 Signature process

6.1 General overview

Figure 1 shows the signature process.

— A hash-code results from applying the hash-function to the message to be signed.

— The format mechanisms produce a trailer, recoverable and non-recoverable parts by dividing the message, and an intermediate string which includes the hash-code and the recoverable part; after minor modifications, the intermediate string becomes a recoverable string.

— A signature finally results from applying the signature function to the recoverable string.

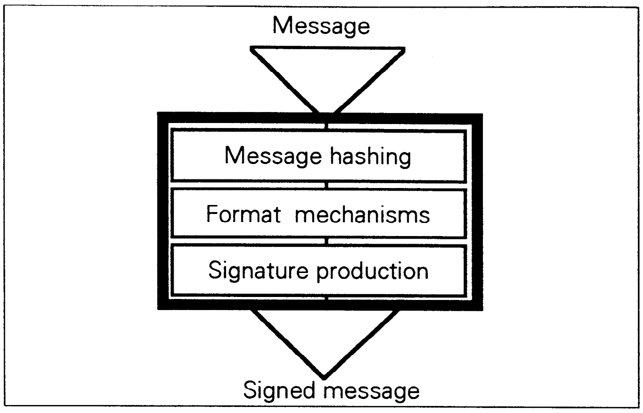

Figure 1 — Signature process

NOTE — Implementations of the signature process should physically protect the operations in such a way that there is no direct access to the signature function under the control of the private signature key. Moreover, every signature should be checked before being delivered to the outside world.

6.2 Message hashing

The k_h bits of the hash-code H shall be computed by applying the hash-function to the k_m bits of the message M.

$$H = \textbf{Hash}(M)$$

6.3 Format mechanisms

6.3.1 Trailer

The trailer shall consist of t consecutive bytes (one or two) where the rightmost nibble shall always be equal to 'C'.

— If the hash-function in use is known (either implicitly or by an identifier coded inside the message), then the trailer shall consist of a single byte (t=1); this byte shall be equal to 'BC'.

— If the hash-function has to be identified, then the trailer shall consist of two consecutive bytes (t=2): the rightmost byte shall be equal to 'CC' and the leftmost byte shall be the hash-function identifier.

The hash-function identifier indicates the hash-function in use.

— The range '00' to '7F' is reserved for ISO/IEC JTC 1; ISO/IEC 10118 fixes a unique identifier in that range for every standardized hash-function.

— The range '80' to 'FF' is reserved for proprietary use.

NOTE — ISO/IEC 10118-3 fixes the values '31', '32' and '33' to identify RIPEMD-160, RIPEMD-128 and SHA-1 respectively.

ISO/IEC JTC 1 reserves for future use the other formats of the trailer, i.e.,

— the values of the rightmost byte different from 'BC' and 'CC' in the range '0C' to 'FC',

— the other lengths and interpretations of the trailer.

6.3.2 Message allocation

6.3.2.1 Total recovery

If $k \geq k_h + k_m + 8t + 4$, then M shall be entirely allocated to the recoverable part Mr while the non-recoverable part Mn shall be kept empty, to give a total recovery.

$b_{k_m} \dots b_2\ b_1$ $\longleftarrow\ Mr\ \longrightarrow$ $k_r = k_m$ bits	Mn is empty

If M is empty, then Mr is also empty.

6.3.2.2 Partial recovery

If $k < k_h + k_m + 8t + 4$, then M shall be divided into two parts respectively allocated to the recoverable and non-recoverable parts, Mr and Mn, to give a partial recovery.

The division shall minimize the length in bytes of Mn. Specifically, x is the least integer greater than or equal to:

$$(k_h + k_m + 8t + 4 - k)\ /\ 8$$

Moreover: $k_r = k_m - 8x$

— The leftmost k_r bits of M shall be allocated to Mr.

— The rightmost x bytes of M shall be allocated to Mn.

$b_{k_m} \dots b_{8x+1}$ $\longleftarrow\ Mr\ \longrightarrow$ $k_r = k_m - 8x$ bits	$b_{8x} \dots b_{8x-7} \dots b_8 \dots b_1$ $\longleftarrow\ Mn\ \longrightarrow$ x bytes

NOTE — This part of ISO/IEC 9796 does not specify the case of no recovery. However, ISO/IEC 14888-3 specifies a scheme with no recovery, which is an extension of the present specifications.

6.3.3 Intermediate string

The intermediate string Si shall consist of the following k bits listed from left to right (see table 1):

— two bits equal to 01, i.e., the header,

— the more-data bit equal to
- 0 in the case of total recovery,
- 1 in the case of partial recovery,

— all the padding bits, i.e.,
- $k - k_h - k_r - 8t - 4$ bits equal to 0,
- and then, the border bit equal to 1,

— the k_r bits of Mr, i.e., the data field,

— the k_h bits of H, i.e., the hash field,

— the $8t$ bits of the trailer.

NOTE — In the case of partial recovery, Mn is kept as short as possible in bytes; therefore the number of padding bits equal to 0 lies in the range zero to seven, i.e., it is less than eight.

6.3.4 Recoverable string

The recoverable string Sr shall result from processing Si from left to right in blocks of four consecutive bits, i.e., in nibbles by starting at the left end.

— The leftmost nibble, where the rightmost bit is either the border bit (1) or not (0), shall remain unchanged.

— If the leftmost nibble is not the border nibble, then
- every subsequent nibble equal to '0', if any, shall be replaced by a nibble equal to 'B'; it is a part of the padding field.
- the first subsequent nibble different from '0' shall be exclusive-ored with 'B'; it is the border nibble.

— All the bits on the right of the border nibble shall remain unchanged.

6.4 Signature production

The recoverable string Sr shall be transformed into the signature Σ by applying the signature function under the control of the private signature key.

$$\Sigma = \textbf{Sign}(\,Sr\,)$$

Consequently, the signed message shall be

— either Σ alone in the case of total recovery,

— or Σ together with Mn in the case of partial recovery.

Table 1 — Structure of the intermediate string *Si* (*k* bits)

Header	More-data bit	Padding field	Data field	Hash field	Trailer
Two bits	One bit	One or more bits	k_r bits	k_h bits	$8t$ bits
= 01	= 0 if Mn empty = 1 otherwise	Zero, one or more bits equal to 0 followed by a single bit equal to 1	Recoverable part Mr	Hash-code H	Either = 'BC' Or = 'XYCC'

7 Verification process

7.1 General overview

Figure 2 shows the verification process.

— A recovered string results from applying the verification function to the signature.

— The recovery mechanisms successively recover an intermediate string, a trailer, a hash-code and a message.

— Another hash-code results from applying the hash-function to the recovered message for finally checking the signed message.

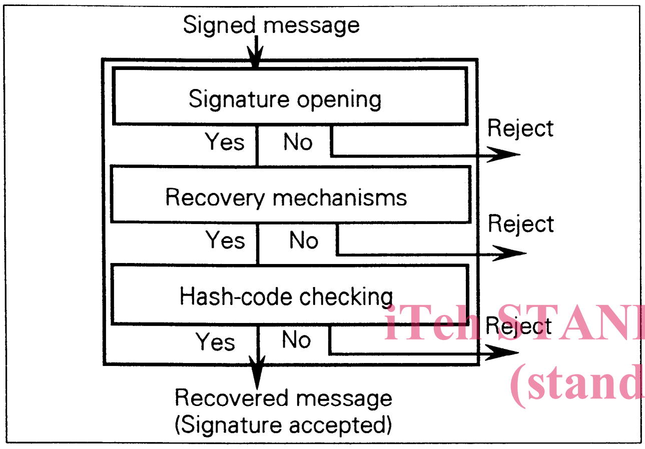

Figure 2 — Verification process

7.2 Signature opening

The signature Σ shall be transformed into the recovered string *Sr'* by applying the verification function under the control of the public verification key.

$$Sr' = \textbf{Verif}(\Sigma)$$

The signature Σ shall be rejected if *Sr'* is not a string of *k* bits where the leftmost two bits are equal to 01 and the rightmost four bits to 1100. Otherwise, the verification process shall continue.

The leftmost third bit of *Sr'* is the more-data bit. Its value indicates whether the recovery is total (0) or partial (1).

7.3 Recovery mechanisms

7.3.1 Intermediate string recovery

The recovered intermediate string *Si'* shall result from processing *Sr'* from left to right in blocks of four consecutive bits, i.e., in nibbles by starting at the left end.

— The leftmost nibble, where the rightmost bit is either the border bit (1) or not (0), shall remain unchanged.

— If the leftmost nibble is not the border nibble, then

- every subsequent nibble equal to 'B', if any, is a part of the padding field,
- the first subsequent nibble different from 'B' shall be exclusive-ored with 'B' to recover the initial value of the border nibble.

— All the bits on the right of the border nibble shall remain unchanged.

In the case of partial recovery, the signature Σ shall be rejected if eight or more padding bits are equal to 0. Otherwise, the verification process shall continue.

7.3.2 Trailer recovery

If the rightmost byte of *Si'* is equal to

— 'BC', then the trailer consists of that single byte; the hash-function in use is known (either implicitly or by an identifier coded inside the message);

— 'CC', then the trailer consists of the rightmost two bytes of *Si'* where the leftmost byte is the identifier of the hash-function in use.

The signature Σ shall be rejected if either the trailer or the hash-function identifier cannot be interpreted. Otherwise, the verification process shall continue.

7.3.3 Hash-code recovery

At this stage, the verifier knows the hash-function in use and therefore, the length k_h' of the hash-code to be recovered.

The header, the more-data bit and the padding field shall be removed on the left of *Si'*; the trailer shall be removed on the right of *Si'*. The remaining binary string shall be divided into two parts.

— The recovered hash-code *H'* shall consist of the rightmost k_h' bits.

— The recovered part *Mr'* shall consist of the remaining bits on the left.

7.3.4 Message recovery

The recovered message *M'* shall be

— either *Mr'* alone in the case of total recovery,

— or the concatenation of *Mr'* as the left part and *Mn* as the right part in the case of partial recovery.

In the case of partial recovery, the signature Σ shall be rejected if *Mn* is not a string of one or more bytes. Otherwise, the verification process shall continue.

7.4 Hash-code checking

Another hash-code *H"* shall be computed by applying the hash-function to the recovered message *M'*.

$$H'' = \textbf{Hash}(M')$$

The signature Σ shall be accepted if the two hash-codes *H'* and *H"* are identical. Then and only then the recovered message *M'* shall be assumed to be identical to the initial message *M*.

Annex A
(informative)

Example of a public-key system for digital signature

A.1 Terms and definitions

For the purposes of this annex, the following definitions apply.

A.1.1
modulus
integer constructed as the product of two primes

A.1.2
private signature key
signature exponent

A.1.3
public verification key
modulus and verification exponent

A.2 Symbols and abbreviations

For the purposes of this annex, the following symbols and abbreviations apply.

Ir	recoverable integer
Ir'	recovered integer
Is	resulting integer
J	representative integer
k	size of the modulus in bits
n	modulus
p, q	prime factors of the modulus
s	signature exponent
v	verification exponent
lcm(a, b)	least common multiple of integers a and b
mod n	arithmetic computation modulo n
(a I n)	Jacobi symbol of a with respect to n

NOTES

1 Let p be an odd prime and let a be a positive integer. The following formula defines the Legendre symbol of a with respect to p.

$$(a \mid p) = a^{(p-1)/2} \bmod p$$

The Legendre symbol of multiples of p with respect to p is 0. When a is not a multiple of p, then the Legendre symbol of a with respect to p is either +1 or −1 depending on whether a is or is not a square modulo p.

2 Let n be an odd positive integer and let a be a positive integer. The Jacobi symbol of a with respect to n is the product of the Legendre symbols of a with respect to the prime factors of n.

Therefore if $n = p\,q$,

then $(a \mid n) = (a \mid p)\,(a \mid q)$.

The Jacobi symbol of any integer a with respect to any integer n may be efficiently computed without the prime factors of n.