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

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 hand-written 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.

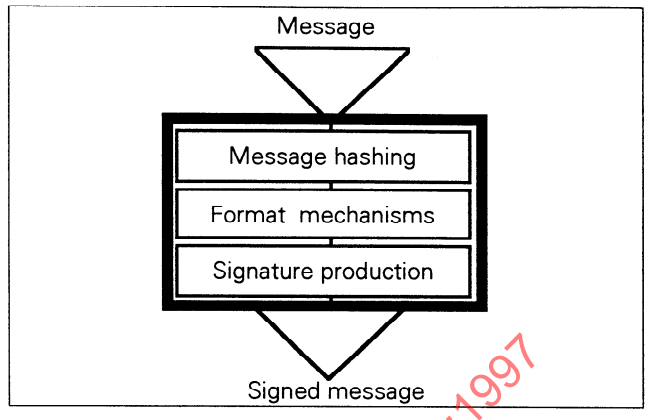

Figure 1 — Signature process

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.

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

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.

$b_{km} \ldots b_2\ b_1$ <—— Mr ——> $k_r = k_m$ 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.

$b_{km} \ldots b_{8x+1}$ <—— Mr ——> $k_r = k_m - 8x$ bits	$b_{8x} \ldots b_{8x-7} \ldots b_8 \ldots b_1$ <———— Mn ————> 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.

$$\Sigma = \mathbf{Sign}(Sr)$$

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.

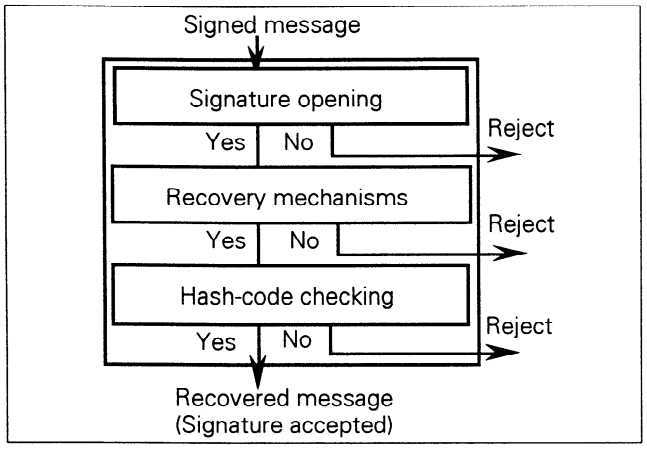

Figure 2 — Verification process

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* | *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*.

A.3 Key production

A.3.1 Public verification exponent

Each signing entity shall select a positive integer v as its public verification exponent. The public verification exponent may be standardized in specific applications.

NOTE — Values 2 and 3 may have some practical advantages.

A.3.2 Secret prime factors and public modulus

Each signing entity shall secretly and randomly select two distinct large primes p and q subject to the following conditions.

— If v is odd, then $p-1$ and $q-1$ shall be coprime to v.

— If v is even, then $(p-1)/2$ and $(q-1)/2$ shall be coprime to v. Moreover, p and q shall not be congruent to each other modulo 8.

The public modulus n is the product of the secret prime factors p and q. The public modulus n lies in the range 2^{k-1} to 2^k-1, i.e., its size is k bits.

$$n = p\,q$$

NOTES

1 Some additional conditions on the choice of primes may be taken into account in order to prevent the modulus factorization.

2 Some forms of the modulus simplify the modular reduction and need less table storage. These forms are

$F_{x,y,-}$ $n = 2^{64x} - c$ of length $k = 64x$ bits,

$F_{x,y,+}$ $n = 2^{64x} + c$ of length $k = 64x+1$ bits,

where $1 \le y \le 2x$ and $c < 2^{64x-8y} < 2c$.

In the moduli of negative form, the most significant $8y$ bits are equal to 1, where $8y$ is at most one quarter of the modulus length.

In the moduli of positive form, after the most significant bit equal to 1, the subsequent $8y$ bits are equal to 0, where $8y$ is at most one quarter of the modulus length.

A.3.3 Private signature exponent

The private signature exponent is any positive integer s such that $sv-1$ is a multiple of

— lcm$(p-1, q-1)$ if v is odd;

— lcm$(p-1, q-1)/2$ if v is even.

NOTE — Generally, s is the least possible value.

A.4 Signature function

The recoverable string Sr is a string of k bits. The leftmost two bits are equal to 01 and the rightmost four bits to 1100. Sr represents an unsigned positive integer which is known as the recoverable integer Ir.

The representative integer of Ir with respect to n and v is denoted as J.

— If v is odd, then J is Ir.

— If v is even and if $(Ir \mid n) = +1$, then J is Ir.

— If v is even and if $(Ir \mid n) = -1$, then J is $Ir/2$.

NOTE — If v is even, then the above operation ensures that the Jacobi symbol of J with respect to n is always +1.

The representative integer J shall be raised to the power s modulo n. Either the result or its complement to n is less than $n/2$: the value smaller than $n/2$ shall be selected.

$$\min \{ J^s \bmod n,\ n-(J^s \bmod n) \}$$

The signature Σ is the string of k bits representing the above value as an unsigned positive integer; therefore the leftmost bit of Σ is equal to 0. This defines the signature function.

$$\Sigma = \mathbf{Sign}(Sr)$$

A.5 Verification function

The signature Σ is a string of k bits where the leftmost bit is equal to 0; that string represents an unsigned positive integer less than $n/2$ which shall be raised to the power v modulo n for getting the resulting integer Is.

$$Is = \Sigma^v \bmod n$$

The recovered integer Ir' is then defined by the following decoding of the resulting integer Is.

— If v is odd and if Is is congruent to
- 12 modulo 16, then Ir' is Is,
- $n-12$ modulo 16, then Ir' is $n-Is$.

— If v is even and if Is is congruent to
- 1 modulo 8, then Ir' is $n-Is$,
- 4 modulo 8, then Ir' is Is,
- 6 modulo 8, then Ir' is $2Is$,
- 7 modulo 8, then Ir' is $2(n-Is)$.

The signature Σ shall be rejected in all the other cases; it shall also be rejected if Ir' is not congruent to 12 modulo 16 and if Ir' does not lie in the range 2^{k-2} to $2^{k-1}-1$.

The recovered string Sr' is the string of k bits representing the recovered integer Ir' as an unsigned positive integer. This defines the verification function.

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

Annex B
(informative)

Illustrative examples related to annex A

B.1 Examples with public exponent 3

B.1.1 Example of key production process

This clause illustrates a modulus of k = 640 bits with v = 3.

Because the public verification exponent v is equal to 3, the secret prime factors are both congruent to 2 modulo 3.

```
p = 1 3630EE72  7602F8A4  11C9A601  34DA37FE
      5A0FD8C7  C18A929D  3C6C87D3  378A8DA7
      D71EB28D  EE15A8AB

q =   D346B3FC  4ED1F278  4C85F1D2  6F76FFC9
      F29A1F2F  D9FD9BCE  7C88F7D4  7C43A617
      03666483  F459427F
```

The public modulus n is the product of the secret prime factors p and q. Its size is 640 bits. Its form is $2^{640}-c$, with $2c > 2^{608} > c$ (form $F_{x,y,-}$ with x = 10 and y = 4).

```
n =   FFFFFFFF  78F6C555  06C59785  E871211E
      E120B0B5  DD644AA7  96D82413  A47B2457
      3F1BE574  5B5CD995  0F6B389B  52350D4E
      01E90009  669A8720  BF265A28  65994190
      A661DEA3  C7828E2E  7CA1B196  51ADC2D5
```

The private signature exponent s is equal to $(n-p-q+3)/6$.

```
s =   2AAAAAAA  942920E3  8120EE96  5168302F
      D0301D73  A4E60C71  43CEB0AD  F0BF30B9
      352F50E8  B9E4CEED  D65343B2  179005B2
      F099915E  4B0C37E4  1314BB08  21AD8330
      D23CBA7F  589E0F12  9B04C46B  67DFCE9D
```

B.1.2 Example with total recovery

B.1.2.1 Example of signature process

The message to be signed is the following string of 56 ASCII-coded characters.

```
abcdbcdecdefdefgefghfghighij
hijkijkljklmklmnlmnomnopnopq
```

In hexadecimal, the message M is the following string of 56 bytes, i.e., 448 bits.

```
M =   61626364  62636465  63646566  64656667
      65666768  66676869  6768696A  68696A6B
      696A6B6C  6A6B6C6D  6B6C6D6E  6C6D6E6F
      6D6E6F70  6E6F7071
```

The 128 bits of the hash-code H are computed by applying RIPEMD-128 to the 448 bits of M (see ISO/IEC 10118-3: 1997, A.3.8).

```
H =   A1AA0689  D0FAFA2D  DC22E88B  49133A06
```

An identifier in the trailer indicates the hash-function in use; ISO/IEC 10118-3 sets the RIPEMD-128 identifier at the value '32'. Therefore the trailer consists of the following 16 bits.

```
Trailer = 32CC
```

The message is short enough for a total recovery. The 640 bits of the intermediate string Si result from concatenating the two bits of the header equal to 01, the more-data bit equal to 0, 44 (=640–448–128–16–4) padding bits equal to 0, the border bit equal to 1, the 448 bits of Mr (=M), the 128 bits of H and the 16 bits of the trailer.

```
Si =  40000000  00016162  63646263  64656364
      65666465  66676566  67686667  68696768
      696A6869  6A6B696A  6B6C6A6B  6C6D6B6C
      6D6E6C6D  6E6F6D6E  6F706E6F  7071A1AA
      0689D0FA  FA2DDC22  E88B4913  3A0632CC
```

The recoverable string *Sr* results from replacing the ten padding nibbles equal to '0' by ten nibbles equal to 'B' and also the border nibble equal to '1' by a nibble equal to 'A'.

```
Sr =  4BBBBBBB  BBBA6162  63646263  64656364
      65666465  66676566  67686667  68696768
      696A6869  6A6B696A  6B6C6A6B  6C6D6B6C
      6D6E6C6D  6E6F6D6E  6F706E6F  7071A1AA
      0689D0FA  FA2DDC22  E88B4913  3A0632CC
```

The recoverable integer *Ir* is the unsigned positive integer represented by *Sr*. *Ir* is raised to the power *s* modulo *n*. Being less than *n*/2, the result is kept. The binary string representing that integer as an unsigned positive integer is the signature Σ.

```
Σ =   374695B7  EE8B2739  25B4656C  C2E008D4
      14639965  34AA5AA5  AFE72A52  FFD84E11
      8085F855  8F366314  71D043AD  342DE268
      B94B080B  EE18A068  C10965F5  81E7F328
      99AD3788  35477064  ABED8EF3  BD530FCE
```

The signed message consists of the 80 bytes of the signature Σ alone because *Mn* is empty.

B.1.2.2 Example of verification process

The signature Σ is a binary string representing an unsigned positive integer less than *n*/2. That integer is raised to the power 3 modulo *n*, thus providing the resulting integer *Is*.

```
Is =  4BBBBBBB  BBBA6162  63646263  64656364
      65666465  66676566  67686667  68696768
      696A6869  6A6B696A  6B6C6A6B  6C6D6B6C
      6D6E6C6D  6E6F6D6E  6F706E6F  7071A1AA
      0689D0FA  FA2DDC22  E88B4913  3A0632CC
```

Because *Is* is here congruent to 12 modulo 16, the recovered integer is *Ir'* = *Is*.

Ir' is represented as an unsigned positive integer by the recovered string *Sr'*.

— The leftmost byte of *Sr'* is equal to '4B'; it consists of the header equal to 01, the more-data bit equal to 0 (total recovery), one padding bit equal to 0 and one padding nibble equal to 'B'; it is followed by 9 nibbles equal to 'B' and the border nibble equal to 'A'; those six bytes are removed on the left of *Sr'*.

— The rightmost byte of *Sr'* is equal to 'CC'; therefore the trailer is equal to '32CC'; those two bytes are also removed on the right of *Sr'*.

The hash-function identifier is equal to '32'; therefore according to ISO/IEC 10118-3, the hash-function in use is RIPEMD-128.

The remaining string of 576 bits is divided into two parts.

— *Mr'* consists of the leftmost 448 bits.

— *H'* consists of the rightmost 128 bits.

```
Mr' =  61626364  62636465  63646566  64656667
       65666768  66676869  6768696A  68696A6B
       696A6B6C  6A6B6C6D  6B6C6D6E  6C6D6E6F
       6D6E6F70  6E6F7071

H' =   A1AA0689  D0FAFA2D  DC22E88B  49133A06
```

The recovered message *M'* consists of *Mr'* alone because the recovery is total. Another hash-code *H"* is computed by applying RIPEMD-128 to *M'*.

```
H" =   A1AA0689  D0FAFA2D  DC22E88B  49133A06
```

Because the two hash-codes *H'* and *H"* are identical, the signature Σ is accepted. Therefore the recovered message *M'* is assumed to be identical to the initial message *M*.

B.1.3 Example with partial recovery

B.1.3.1 Example of signature process

This example illustrates the signature of a message of 896 bits, i.e., 112 bytes.

```
M =   fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
```

The 160 bits of the hash-code *H* are computed by applying RIPEMD-160 to the 896 bits of the message *M*.

```
H  =  71df2f2c  3fff1e35  df6920b4  5115fac5
      998ba88e
```

The hash-function in use is implicitly known. Therefore the trailer consists of a single byte.

```
Trailer = BC
```

The message is too long for being entirely recoverable by the verification process. Therefore, it is divided into two parts.

— *Mr* consists of the leftmost 464 bits.

— *Mn* consists of the remaining 432 bits, i.e., 54 bytes.

```
Mr =  fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210     fedc

Mn =                         ba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
      fedcba98  76543210  fedcba98  76543210
```

The 640 bits of the intermediate string *Si* result from concatenating the two bits of the header equal to 01, the more-data bit equal to 1, four (=640–464–160–8–4) padding bits equal to 0, the border bit equal to 1, the 464 bits of *Mr*, the 160 bits of *H* and the 8 bits of the trailer.

Si = 61fedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedc71 df2f2c3f
ff1e35df 6920b451 15fac599 8ba88ebc

The recoverable string Sr results from replacing the border nibble equal to '1' by a nibble equal to 'A'.

Sr = 6afedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedcba 98765432
10fedcba 98765432 10fedc71 df2f2c3f
ff1e35df 6920b451 15fac599 8ba88ebc

The recoverable integer Ir is the unsigned positive integer represented by Sr. Ir is raised to the power s modulo n. Being less than $n/2$, the result is kept. The binary string representing that integer as an unsigned positive integer is the signature Σ.

Σ = 5CF9A018 54DBACAE C83AAE8E FC563D74
538192E9 5466BABA CD361D7C 86000FE4
2DCB4581 E48E4FEB 862D0469 8DA9203B
1803B262 105104D5 10B365EE 9C660857
BA1C001A A57ABFD1 C8DE92E4 7C275CAE

The signed message consists of the 80 bytes of the signature Σ together with the 54 bytes of the non-recoverable part Mn, i.e., only 22 bytes more than the message M.

B.1.3.2 Example of verification process

The signature Σ is a binary string representing an unsigned positive integer less than $n/2$. That integer is raised to the power 3 modulo n, thus providing the resulting integer Is.

Is = 6AFEDCBA 98765432 10FEDCBA 98765432
10FEDCBA 98765432 10FEDCBA 98765432
10FEDCBA 98765432 10FEDCBA 98765432
10FEDCBA 98765432 10FEDC71 DF2F2C3F
FF1E35DF 6920B451 15FAC599 8BA88EBC

Because Is is here congruent to 12 modulo 16, the recovered integer is $Ir' = Is$.

Ir' is represented as an unsigned positive integer by the recovered string Sr'.

— The leftmost byte of Sr' is equal to '6A'; it consists of the header equal to 01, the more-data bit equal to 1 (partial recovery), one padding bit equal to 0 and the border nibble equal to 'A'; this byte is removed on the left of Sr'.

— The rightmost byte of Sr' is equal to 'BC'; this byte, i.e., the trailer, is also removed on the right of Sr'.

Because the trailer is equal to 'BC', the hash-function in use shall be implicitly known: RIPEMD-160 in this example.

The remaining string of 624 bits is divided into two parts.

— Mr' consists of the leftmost 464 bits.

— H' consists of the rightmost 160 bits.

Mr' = fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedc

H' = 71df2f2c 3fff1e35 df6920b4 5115fac5
998ba88e

Because the recovery is partial, the recovered message M' consists of the concatenation of Mr' and Mn, the recovered and non-recoverable parts respectively.

M' = fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210
fedcba98 76543210 fedcba98 76543210

Another hash-code H'' is computed by applying RIPEMD-160 to the recovered message M'.

H'' = 71df2f2c 3fff1e35 df6920b4 5115fac5
998ba88e

Because the two hash-codes H' and H'' are identical, the signature Σ is accepted and the recovered message M' is assumed to be identical to the initial message M.

B.2 Examples with public exponent 2

B.2.1 Example of key production process

This clause illustrates a modulus of k = 768 bits with v = 2. The signature process involves the Jacobi symbol.

Because v is even (here v = 2), one secret prime factor is congruent to 3 modulo 8 and the other one to 7 modulo 8.

p = 1 7F53C2E3 F109D600 F7F6145E 9C8DD582
026B8779 9C35C769 C4E5FDB3 735FB398
8D16E95B 7B9BCDA4 A1E60DD4 177F0AEB

q = AAF759FF 9CC86291 8007E811 6BAD330B
0E997FA6 7085E65C D26D72E7 2C276E5B
BE124BD5 7645F0C0 3EC2C5BA FC075427

The public modulus n is the product of the secret prime factors p and q. Because the factors p and q are congruent to 3 and 7 modulo 8, the value of the rightmost nibble of n is equal to either 5 or D.

In this example, the size of n is 768 bits. Its form is $2^{768}-c$, with 2 $c > 2^{736} > c$ (form $F_{x,y,-}$ with x = 12 and y = 4).

n = FFFFFFFF 45F1903E BB83D4D3 63F70DC6
47B839F2 A84E119B 8830B2DE C424A1CE
0C9FD667 966B8140 7E892782 83F27CA8
857D4097 9407FC6D A4CC8A20 ECB4B891
3B581333 2409BC1F 391A94C9 C328DFE4
6695DAF9 22259174 544E2BFB E45CC5CD

Because $(p-1)/2$ and $(q-1)/2$ are coprime, the private signature exponent s is equal to $(n-p-q+5)/8$.

```
s =   1FFFFFFF  E8BE3207  D7707A9A  6C7EE1B8
      C8F7073E  5509C233  7106165B  D8849439
      C193FACC  F2CD7028  0FD124F0  507E4F94
      CB664476  80C6B87B  6599D1B6  1C8F3600
      854A6182  62E9C1CB  1438E485  E47437BE
      036D94B9  06087A61  EE74AB0D  9A1ACCD8
```

B.2.2 Example with total recovery

B.2.2.1 Example of signature process

The message to be signed consists of the following string of 56 ASCII-coded characters.

```
abcdbcdecdefdefgefghfghighij
hijkijkljklmklmnlmnomnopnopq
```

In hexadecimal, the message M consists of the following string of 56 bytes, i.e., 448 bits.

```
M =   61626364  62636465  63646566  64656667
      65666768  66676869  6768696A  68696A6B
      696A6B6C  6A6B6C6D  6B6C6D6E  6C6D6E6F
      6D6E6F70  6E6F7071
```

The 160 bits of the hash-code H are computed by applying SHA-1 to the 448 bits of M (see ISO/IEC 10118-3: 1997, A.4.8).

```
H =   84983E44  1C3BD26E  BAAE4AA1  F95129E5
      E54670F1
```

An identifier in the trailer indicates the hash-function in use; ISO/IEC 10118-3 sets the SHA-1 identifier at the value '33'. Therefore the trailer consists of the following 16 bits.

```
Trailer = 33CC
```

The message is short enough for a total recovery. The 768 bits of the intermediate string Si result from concatenating the two bits of the header equal to 01, the more-data bit equal to 0, 140 (=768–448–160–16–4) padding bits equal to 0, the border bit equal to 1, the 448 bits of Mr (=M), the 160 bits of H and the 16 bits of the trailer.

```
Si =  40000000  00000000  00000000  00000000
      00016162  63646263  64656364  65666465
      66676566  67686667  68696768  696A6869
      6A6B696A  6B6C6A6B  6C6D6B6C  6D6E6C6D
      6E6F6D6E  6F706E6F  70718498  3E441C3B
      D26EBAAE  4AA1F951  29E5E546  70F133CC
```

The recoverable string Sr results from replacing the 34 padding nibbles equal to '0' by 34 nibbles equal to 'B' and the border nibble equal to '1' by a nibble equal to 'A'.

```
Sr =  4BBBBBBB  BBBBBBBB  BBBBBBBB  BBBBBBBB
      BBBA6162  63646263  64656364  65666465
      66676566  67686667  68696768  696A6869
      6A6B696A  6B6C6A6B  6C6D6B6C  6D6E6C6D
      6E6F6D6E  6F706E6F  70718498  3E441C3B
      D26EBAAE  4AA1F951  29E5E546  70F133CC
```

The recoverable integer Ir is the unsigned positive integer represented by Sr. Because the Jacobi symbol of Ir with respect to n is –1, the representative integer is $J = Ir/2$.

```
J =   25DDDDDD  DDDDDDDD  DDDDDDDD  DDDDDDDD
      DDDD30B1  31B23131  B232B1B2  32B33232
      B333B2B3  33B43333  B434B3B4  34B53434
      B535B4B5  35B63535  B636B5B6  36B73636
      B737B6B7  37B83737  B838C24C  1F220E1D
      E9375D57  2550FCA8  94F2F2A3  387899E6
```

J is raised to the power s modulo n.

```
      D279A2A7  D29BC5A0  E941F4BF  277D6240
      02179827  A6B68C46  382DC76E  0EAD9125
      AD99645B  52002C8B  A728DBAC  7514DDD9
      161A8231  45D83944  2B8C731C  F56243CE
      CA304DFE  AF00C180  CD2CD430  FC403C5E
      D89EBBD5  B55C40E5  150BF0AA  96F704AE
```

Because the above result is greater than $n/2$, it is replaced by its complement to n. The binary string representing that integer as an unsigned positive integer is the signature Σ.

```
Σ =   2D865D57  7355CA9D  D241E014  3C79AB86
      45A0A1CB  01978555  5002EB6F  B57710A8
      5F06720C  446B54B4  D7604BD6  0EDD9ECF
      6F62BE66  4E2FC329  79401703  F75274C2
      7127C534  7508FA9E  6BEDC098  C6E8A385
      8DF71F23  6CC9508F  3F423B51  4D65C11F
```

The signed message consists of the 96 bytes of the signature Σ alone because Mn is empty.

B.2.2.2 Example of verification process

The signature Σ is a binary string representing an unsigned positive integer which is less than $n/2$. That integer is squared modulo n, thus providing the resulting integer Is.

```
Is =  DA222221  6813B260  DDA5F6F5  86192FE8
      69DB0941  769BE069  D5FE012C  91716F9B
      596C23B4  62B74E0C  CA5473CE  4F3D4873
      D0478BE2  5E51C737  EE95D46A  B5FD825A
      84205C7B  EC5184E7  80E1D27D  A406D1C6
      7D5E7DA1  FCD494CB  BF5B3958  ABE42BE7
```

The verification process does not involve the Jacobi symbol. Because the least significant three bits of the resulting integer Is are equal to 111, $Ir' = 2(n-Is)$.

```
Ir' =  4BBBBBBB  BBBBBBBB  BBBBBBBB  BBBBBBBB
       BBBA6162  63646263  64656364  65666465
       66676566  67686667  68696768  696A6869
       6A6B696A  6B6C6A6B  6C6D6B6C  6D6E6C6D
       6E6F6D6E  6F706E6F  70718498  3E441C3B
       D26EBAAE  4AA1F951  29E5E546  70F133CC
```

Ir' is represented as an unsigned positive integer by the recovered string Sr'.

— The leftmost byte of Sr' is equal to '4B'; it consists of the header equal to 01, the more-data bit equal to 0 (total recovery), one padding bit equal to 0 and one padding nibble equal to 'B'; it is followed by 33 nibbles equal to 'B' and the border nibble equal to 'A'; those 18 bytes are removed on the left of Sr'.