DRAFT INTERNATIONAL STANDARD
ISO/IEC DIS 9797-2

ISO/IEC JTC **1**/SC **27** Secretariat: **DIN**

Voting begins on: **2020-09-01** Voting terminates on: **2020-11-24**

Information security — Message authentication codes (MACs) —

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

Partie 2: Mécanismes utilisant une fonction de hachage dédiée

ICS: 35.030

Reference number
ISO/IEC DIS 9797-2:2020(E)

Contents

Page

Foreword .... **v**

1 Scope **1**

2 Normative reference **1**

3 Terms and definitions **1**

4 Symbols and notation **3**

5 Requirements **5**

6 MAC Algorithm 1 .... **6**
- 6.1 General 6
- 6.2 Description of MAC Algorithm 1 7
  - 6.2.1 General 7
  - 6.2.2 Step 1 (key expansion) 7
  - 6.2.3 Step 2 (modification of the constants and the IV) 8
  - 6.2.4 Step 3 (hashing operation) 8
  - 6.2.5 Step 4 (output transformation) 8
  - 6.2.6 Step 5 (truncation) 8
- 6.3 Efficiency 8
- 6.4 Computation of the constants 9
  - 6.4.1 General 9
  - 6.4.2 Dedicated Hash-Function 1 (RIPEMD-160) 9
  - 6.4.3 Dedicated Hash-Function 2 (RIPEMD-128) 9
  - 6.4.4 Dedicated Hash-Function 3 (SHA-1) 10
  - 6.4.5 Dedicated Hash-Function 4 (SHA-256) 10
  - 6.4.6 Dedicated Hash-Function 5 (SHA-512) 11
  - 6.4.7 Dedicated Hash-Function 6 (SHA-384) 12
  - 6.4.8 Dedicated Hash-Function 8 (SHA-224) 12
  - 6.4.9 Dedicated Hash-Function 17 (SM3) 12

7 MAC Algorithm 2 .... **13**
- 7.1 General 13
- 7.2 Description of MAC Algorithm 2 13
  - 7.2.1 General 13
  - 7.2.2 Step 1 (key expansion) 13
  - 7.2.3 Step 2 (hashing operation) 14
  - 7.2.4 Step 3 (output transformation) 14
  - 7.2.5 Step 4 (truncation) 14
- 7.3 Efficiency 14

8 MAC Algorithm 3 .... **14**
- 8.1 General 14
- 8.2 Description of MAC Algorithm 3 15
  - 8.2.1 General 15
  - 8.2.2 Step 1 (key expansion) 15
  - 8.2.3 Step 2 (modification of the constants and the IV) 15
  - 8.2.4 Step 3 (padding) 15
  - 8.2.5 Step 4 (application of the round-function) 16
  - 8.2.6 Step 5 (truncation) 16
- 8.3 Efficiency 16

9 MAC Algorithm 4 .... **16**
- 9.1 General 16
- 9.2 Description of MAC Algorithm 4 17
- 9.3 Encoding and padding 17
  - 9.3.1 Integer to byte encoding 17
  - 9.3.2 String encoding 18

9.3.3 Padding ... 18
9.4 KMAC128 ... 19
9.4.1 General ... 19
9.4.2 Step 1 (Prepare ***newD***) ... 19
9.4.3 Step 2 (Prepare ***X***) ... 19
9.4.4 Step 3 (Generate MAC output) ... 19
9.5 KMAC256 ... 19
9.5.1 General ... 19
9.5.2 Step 1 (Prepare ***newD***) ... 19
9.5.3 Step 2 (Prepare X) ... 20
9.5.4 Step 3 (Generate MAC output) ... 20
9.6 KMACXOF128 ... 20
9.6.1 General ... 20
9.6.2 Step 1 (Prepare ***newD***) ... 20
9.6.3 Step 2 (Prepare ***X***) ... 20
9.6.4 Step 3 (Generate MAC output) ... 21
9.7 KMACXOF256 ... 21
9.7.1 General ... 21
9.7.2 Step 1 (Prepare ***newD***) ... 21
9.7.3 Step 2 (Prepare ***X***) ... 21
9.7.4 Step 3 (Generate MAC output) ... 21

Annex A (normative) **Object identifiers** ... **22**

Annex B (informative) **Numerical examples** ... **24**

Annex C (informative) **A Security Analysis of the MAC Algorithms** ... **50**

Bibliography ... **52**

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/or] 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*, SC 27, Information security, cybersecurity and privacy protection.

This third edition cancels and replaces the second edition (ISO/IEC 9797-2:2011).

The main changes include using Dedicated Hash-Functions 17 for MAC Algorithms 1 and 3, Dedicated Hash-Functions 11, 12, 13 — 16, and 17 for MAC Algorithm 2, and adding MAC Algorithm 4 based on Keccak, a primitive used in the definition of Dedicated Hash Function 13 –16. The Dedicated Hash-Functions are specified in ISO/IEC 10118-3:2018

Information security — Message authentication codes (MACs) —

Part 2:
Mechanisms using a dedicated hash-function

1 Scope

This document specifies MAC algorithms that use a secret key and a hash-function (or its round-function or sponge function) to calculate an m-bit MAC. These mechanisms can be used as data integrity mechanisms to verify that data has not been altered in an unauthorised manner.

NOTE A general framework for the provision of integrity services is specified in ISO/IEC 10181-6 [5].

2 Normative reference

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 10118-3:2018, *Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash-functions (fourth edition)*.

3 Terms and definitions

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

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

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

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

3.1
block
bit-string of length L_1, i.e., the length of the first input to the round-function

[SOURCE: ISO/IEC 10118-3:2018, 3.1]

3.2
entropy
total amount of information yielded by a set of bits, and it is representative of the work effort required for an adversary to be able to reproduce the same set of bits

[SOURCE: ISO/IEC 18031:2011, 3.11]

3.3
input data string
string of bits which is the input to a MAC algorithm

3.4
hash-code
string of bits which is the output of a hash-function

[SOURCE: ISO/IEC 10118-1:2016, 3.3]

3.5
hash-function
function which maps strings of bits 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 of ISO/IEC 10118-1:2016.

[SOURCE: ISO/IEC 10118-1:2016, 3.4]

3.6
initializing value
value used in defining the starting point of a hash-function

[SOURCE: ISO/IEC 10118-1:2016, 3.5]

3.7
MAC algorithm key
key that controls the operation of a MAC algorithm

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

3.8
Message Authentication Code
MAC
string of bits which is the output of a MAC algorithm NOTE 1 to entry: A MAC is sometimes called a cryptographic check value (see for example ISO 7498-2 [1]).

[SOURCE: ISO/IEC 9797-1:2011, 3.9]

3.9
Message Authentication Code algorithm
MAC algorithm
algorithm for computing a function which maps strings of bits and a secret key to fixed-length strings of bits, satisfying the following two properties:

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

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

Note 1 to entry: to entry: A MAC algorithm is sometimes called a cryptographic check function (see for example ISO 7498-2 [1]).

Note 2 to entry: to entry: Computational feasibility depends on the user's specific security requirements and environment.

[SOURCE: ISO/IEC 9797-1:2011, 3.10]

3.10
output transformation
function that is applied at the end of the MAC algorithm, before the truncation operation

[SOURCE: ISO/IEC 9797-1:2011, 3.12]

3.11
padding
appending extra bits to a data string

[SOURCE: ISO/IEC 10118-1:2016, 3.7]

3.12
round-function
function $\varnothing$ (..,..)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: 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.

[SOURCE: ISO/IEC 10118-1:2016, 3.8]

3.13
security strength
number associated with the amount of work required to break a cryptographic algorithm or system and specified in bits such that security strength *s* bits implies the required number of operations is 2^s

Note 1 to entry: to entry: Computationally infeasible in 3.2, 3.6, and 3.10 implies the security strength is at least 112 bits. Refer to Annex C of ISO/IEC 10118-1:2016.

3.14
word
string of 32 bits used in Dedicated Hash-Functions 1, 2, 3, 4, 8 and 17 or a string of 64 bits used in Dedicated Hash-Functions 5, 6, 9 and 10 of ISO/IEC 10118-3:2018

[SOURCE: ISO/IEC 10118-3:2018, 3.2, Modified with specific bit lengths.]

4 Symbols and notation

This document makes use of the following symbols and notation defined in ISO/IEC 9797-1 [3]:

D	the input data string, i.e. the data string to be input to the MAC algorithm.
$j \sim X$	the string obtained from the string X at least j bits in length by taking the leftmost j bits of X.
m	the length (in bits) of the MAC.
q	the number of blocks in the input data string D after the padding and splitting process.
$X \oplus Y$	bitwise exclusive-or of bit-strings X and Y.
$X \| Y$	concatenation of bit-strings X and Y (in that order).
: =	a symbol denoting the 'set equal to' operation used in the procedural specifications of MAC algorithms, where it indicates that the value of the string on the left side of the symbol shall be made equal to the value of the expression on the right side of the symbol.

For the purposes of this document, the following symbols and notation apply:

$\bar{D}$ padded data string.

h hash-function.

h' the hash-function h with modified constants and modified IV.

$\bar{h}$ simplified hash-function h without the padding and length appending, and without truncating the round-function output (L_2 bits) to its left-most L_H bits.

NOTE 1 $\bar{h}$ shall only be applied to input strings with a length that is a positive integer multiple of L_1.

NOTE 2 The output of $\bar{h}$ should be L_2 bits rather than L_H bits; in particular, in Dedicated Hash-Functions 6 and 8 defined in ISO/IEC 10118-3:2018, L_H is always smaller than L_2.

H', H'' strings of L_2 bits which are used in the MAC algorithm computation to store an intermediate result.

IV', IV_1, IV_2 initializing values.

k length (in bits) of the MAC algorithm key.

K MAC algorithm key.

K', K_0, K_1, K_2, $\bar{K}$, $\bar{K}_1$, $\bar{K}_2$ secret keys derived to be used for a MAC algorithm.

KT the first input string of the function ϕ' used in the output transformation step of MAC Algorithm 1.

$\tilde{L}$ the bit string encoding the message length in MAC Algorithm 3.

$OPAD$, $IPAD$ constant strings used in MAC Algorithm 2.

R, S_0, S_1, S_2 constant strings used in the computation of the constants for MAC Algorithm 1 and MAC Algorithm 3.

T_0, T_1, $T_{2,}$ U_0, U_1, U_2 constant strings used in the key derivation for MAC Algorithm 1 and MAC Algorithm 3.

ϕ'

$K_1[i]$

This document makes use of the following symbols and notation defined in ISO/IEC 10118-1:

H hash-code.

IV initializing value.

L_X length (in bits) of a bit-string X.

This document makes use of the following symbols and notation defined in ISO/IEC 10118-3:2018:

C_i, C'_i constant words used in the round-functions.

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

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

w the length (in bits) of a word; w is 32 when using Dedicated Hash-Functions 1, 2, 3, 4, 8 and 17 of ISO/IEC 10118-3:2018, and w is 64 when using Dedicated Hash-Functions 5, 6, 9 and 10 of ISO/IEC 10118-3:2018.

ϕ a round-function, i.e. if X and Y are bit-strings of lengths L_1 and L_2 respectively, then ϕ (X, Y) is the string obtained by applying ϕ to X and Y.

Ψ the modulo 2^w addition operation, where w is the number of bits in a word. i.e. if A and B are words, then $\mathrm{A}\Psi B$ is the word obtained by treating A and B as the binary representations of integers and computing their sum modulo 2^w, and the result is constrained to lie between 0 and $2^w - 1$ inclusive. The value of w is 32 in Dedicated Hash-Functions 1, 2, 3, 4, 8 and 17, and 64 in Dedicated Hash-Functions 5, 6, 9 and 10.

5 Requirements

Users who wish to employ a MAC algorithm from this document shall select:

— a dedicated hash-function from the functions specified in ISO/IEC 10118-3:2018 so that the hash function and its round function or its sponge function is implemented or suitable to use; a MAC algorithm amongst those specified in Clauses 6, 7, 8 and 9 which can use the selected hash function or its round function or sponge function; and

— the length (in bits) m of the MAC, where m is at least 32.

The use of Dedicated Hash-Functions 7 and 9-16 from ISO/IEC 10118-3:2018 with MAC Algorithms 1 and 3 is not specified in this document. The use of Dedicated Hash-Functions 9 and 10 from ISO/IEC 10118-3:2018 with MAC Algorithm 2 is also not specified in this document. MAC Algorithm 4 makes use of the Keccak function, a primitive (known as a sponge function) used in defining Dedicated Hash-Functions 13-16 from ISO/IEC 10118-3:2018. The permitted combinations of MAC algorithms and hash-functions are summarized in Table 1.

Table 1 — The MAC algorithm with different Dedicated Hash-Functions.

Dedicated Hash-Function in ISO/IEC 10118-3:2018	MAC Algorithm 1	MAC Algorithm 2	MAC Algorithm 3	MAC Algorithm 4
1 RIPEMD-160	√	√	√	
2 RIPEMD-128	√	√	√	
3 SHA-1	√	√	√	
4 SHA-256	√	√	√	
5 SHA-512	√	√	√	
6 SHA-384	√	√	√	
7 Whirlpool		√		
8 SHA-224	√	√	√	
9 SHA-512/224				
10 SHA-512/256				
11 STREEBOG 512		√		
12 STREEBOG 256		√		
13 SHA3–224		√		√

Table 1 *(continued)*

Dedicated Hash-Function in ISO/IEC 10118-3:2018	MAC Algorithm 1	MAC Algorithm 2	MAC Algorithm 3	MAC Algorithm 4
14 SHA3–256		√		√
15 SHA3–384		√		√
16 SHA3–512		√		√
17 SM3	√	√	√	

Agreement on these choices amongst the users is essential for use of the data integrity mechanism.

The key K used in a MAC algorithm shall have entropy that meets or exceeds the security strength to be provided by the MAC algorithm.

In every case, the MAC algorithm key K shall be chosen such that every possible key is approximately equally likely to be selected.

For MAC Algorithms 1 and 2, the length m of the MAC is a positive integer less than or equal to the length of the hash-code L_H. For MAC Algorithm 2, the length m of MAC value must be at least 32 bits. For MAC Algorithm 3, the length m of the MAC is a positive integer less than or equal to half the length of the hash-code, i.e. $m \leq L_H /2$. The length in bits of the input data string may be limited by the dedicated hash function or/and the MAC algorithm and will be discussed for each MAC algorithm. For MAC Algorithm 4, the length in bits of the input data string D shall be at most $2^{2040} - 1$. The selection of a specific MAC algorithm, dedicated hash-function as specified in Table 1, and value for m is beyond the scope of this document.

These choices affect the security level of the MAC algorithm. For a detailed discussion, see Annex C. The key used for calculating and verifying the MAC is the same. If the input data string is also being enciphered, the key used for the calculation of the MAC should be different from that used for encipherment, because it is considered as good cryptographic practice to have independent keys for confidentiality and for data integrity.

Annex A lists the object identifiers which shall be used to identify the mechanisms defined in this document.

Annex B provides numerical examples for the MAC algorithms specified in this document to be used for checking the correctness of implementations.

Annex C describes major attacks and the proofs of security for the MAC algorithms specified in this document.

6 MAC Algorithm 1

6.1 General

This clause contains a description of MDx-MAC [9] with Dedicated Hash-Functions 1 – 6, 8 and 17. Table 1 shows the commonly used names of MDx-MAC with individual dedicated hash-functions.

Table 2 — The MDx-MAC algorithm with different Dedicated Hash-Functions

Dedicated Hash-Function:	The MDx-MAC algorithm is also known as
Dedicated Hash-Function 1	RIPEMD-160-MAC
Dedicated Hash-Function 2	RIPEMD-128-MAC
Dedicated Hash-Function 3	SHA-1-MAC
Dedicated Hash-Function 4	SHA-256-MAC
Dedicated Hash-Function 5	SHA-512-MAC

Table 2 *(continued)*

Dedicated Hash-Function 6	SHA-384-MAC
Dedicated Hash-Function 8	SHA-224-MAC
Dedicated Hash-Function 17	SM3-MAC

The use of MAC Algorithm 1 with Dedicated Hash-Functions 7, 9 –16 of ISO/IEC 10118-3:2018 is not specified in this document.

MAC Algorithm 1 requires one application of the hash-function to compute a MAC value but requires that the constants in the corresponding round-function are modified.The hash-function shall be selected from Dedicated Hash-Functions 1 – 6, 8 and 17 from ISO/IEC 10118-3:2018.MAC Algorithm 1 can accommodate the maximum of 128 bit key K and therefore provide at most 128 bits security strength. For MAC Algorithm 1, the length in bits of the input data string D shall be at most 2^{64} - 1 when using Dedicated Hash-Functions 1, 2, 3, 4, 8 and 17, and at most 2^{128} – 1 when using Dedicated Hash-Functions 5 and 6.

6.2 Description of MAC Algorithm 1

6.2.1 General

MAC algorithm 1 involves the following five steps: key expansion, modification of the constants and the *IV*, hashing operation, output transformation, and truncation.

6.2.2 Step 1 (key expansion)

If K is shorter than 128 bits, concatenate K to itself $\lceil 128/k \rceil$ times and select the leftmost 128 bits of the result to form the 128-bit key K':

K': = 128 ~ (K || K || … || K). If the length (in bits) of K is larger than or equal to 128, K': = 128~K.

Compute the derived keys K_0, K_1, and K_2 as follows:

$K_0 := \bar{h}\ (K' \| U_0 \| K')$

$K_1 := 128 \sim \bar{h}\ (K' \| U_1 \| K')$, when using Dedicated Hash-Functions 1, 2 and 3

$K_1 := 256 \sim \bar{h}\ (K' \| U_1 \| K')$, when using Dedicated Hash-Functions 4, 5, 6, 8 and 17

$K_2 := 128 \sim \bar{h}\ (K' \| U_2 \| K')$.

Here $\bar{h}$ is a simplified hash function h selected from dedicated hash functions listed in Table 2 and U_0, U_1, and U_2 are 768-bit constants that are defined in 6.4.1.

Padding and length appending can be omitted because in this case the length of the input string is either L_1 bits or $2L_1$ bits.

When deriving K_0, truncation is omitted and the length of K_0 is always L_2 bits.

When using Dedicated Hash-Functions 1, 2, 3, 5 and 6, the derived key K_1 is split into four words denoted by $K_1[i]$ $(0 \le i \le 3)$, i.e.

$K_1 = K_1[0] \| K_1[1] \| K_1[2] \| K_1[3]$.

When using Dedicated Hash-Functions 4, 8, and 17 the derived key K_1 is split into eight words denoted by $K_1[i]$ $(0 \le i \le 7)$, i.e.

$K_1 = K_1[0] \| K_1[1] \| K_1[2] \| K_1[3] \| K_1[4] \| K_1[5] \| K_1[6] \| K_1[7]$.

For the conversion of a string into words, a byte ordering convention is required. The byte ordering convention for this conversion is that which is defined for the selected dedicated hash-function in ISO/IEC 10118-3:2018.

6.2.3 Step 2 (modification of the constants and the IV)

When using Dedicated Hash-Functions 1, 2, 3, 4, 5, 6, 8, and 17, the additive constants used in the round-function are modified by the modulo 2^w addition of a word of K_1, e.g.

$$C_0 := C_0 \, \Psi \, K_1[0].$$

In 6.4, it indicates which word of K_1 is added to each constant.

The initializing value *IV* of the hash function is replaced by $IV' := K_0$.

6.2.4 Step 3 (hashing operation)

The string which is input to the modified hash-function *h′* is equal to the input data string *D*, i.e.

$$H' := h'(D).$$

6.2.5 Step 4 (output transformation)

The modified round-function ϕ' is applied one additional time, with as first input the string *KT* (defined below) and as second input the string *H′* (the result of Step 3) i.e.

$$H'' := \phi'(KT, H').$$

For Dedicated Hash-Functions 1, 2, 3, 4, 8, and 17,

$$KT = K_2 \,||\, (K_2 \oplus T_0) \,||\, (K_2 \oplus T_1) \,||\, (K_2 \oplus T_2).$$

For Dedicated Hash-Functions 5 and 6,

$$KT = K_2 \,||\, (K_2 \oplus T_0) \,||\, (K_2 \oplus T_1) \,||\, (K_2 \oplus T_2) \,||\, K_2 \,||\, (K_2 \oplus T_0) \,||\, (K_2 \oplus T_1) \,||\, (K_2 \oplus T_2).$$

Here T_0, T_1, and T_2 are 128-bit strings defined in 6.4 for each dedicated hash-function.

The output transformation corresponds to processing an additional data block derived from K_2 *after* padding and appending of the length field.

6.2.6 Step 5 (truncation)

The MAC of *m* bits is derived by taking the leftmost *m* bits of the string *H″*, i.e.

$$\mathrm{MAC} := m \sim H''.$$

6.3 Efficiency

If the padded data string (where the padding algorithm depends on the selected hash-function) contains *q* blocks, then MAC Algorithm 1 requires $q + 7$ applications of the round-function, when Dedicated Hash-Functions 1, 2, 3, 4, 8, and 17 are selected and $q + 4$ applications of the round function, when Dedicated Hash-Functions 5 and 6 are selected. This can be reduced to $q + 1$ applications of the round-function by pre-computing the values K_0, K_1 and K_2, and by replacing the initializing value *IV* by *IV′* in the application of the hash-function.It is recommended to make this modification to the code of the hash-function together with the mandatory modification required for Step 2.For long input strings, MAC Algorithm 1 has a performance which is comparable to that of the hash-function used.

6.4 Computation of the constants

6.4.1 General

The constants described in 6.4 are used in MAC Algorithms 1 and 3. MAC Algorithm 3 is specified in Clause 8.The strings T_i and U_i are fixed elements in the description of the MAC algorithm. They are computed (only once) using the hash-function; they are different for each of the eight hash-functions. The 128-bit constants T_i and 768-bit constants U_i are defined as follows. The definition of T_i involves the 496-bit constant R = “ab...yzAB...YZ01...89” and 16-bit constants S_0, S_1, S_2, where S_i is the 16-bit string formed by repeating twice the 8-bit representation of i (e.g. the hexadecimal representation of S_1 is 3131). In both cases ASCII coding is used; this is equivalent to coding using ISO/IEC 646:1991.

for i: = 0 *to* 2

T_i:= 128 ~ $\overline{h}$ $(S_i \,||\, R)$ for Dedicated Hash-Functions 1, 2, 3, 4, 8 and 17;

T_i:= 128 ~ $\overline{h}$ $(S_i \,||\, R \,||\, 0^{512})$ for Dedicated Hash-Functions 5 and 6, where 0^{512} is 512 zero bits, and

for i: = 0 *to* 2 U_i: = $T_i \,||\, T_{i+1} \,||\, T_{i+2} \,||\, T_i \,||\, T_{i+1} \,||\, T_{i+2}$

where the subscripts in T_i are taken modulo 3.In Dedicated Hash-Functions 1, 2, 3, 4, 5, 6, 8, and 17, for all constants C_i, C'_i and all words $K_1[i]$ the most significant bit corresponds to the left-most bit. The constants C_i and C'_i are presented using a hexadecimal representation.

6.4.2 Dedicated Hash-Function 1 (RIPEMD-160)

The 128-bit constant strings T_i for Dedicated Hash-Function 1 are defined as follows (in hexadecimal representation):

T_0 = 1CC7086A046AFA22353AE88F3D3DACEB

T_1 = E3FA02710E491D851151CC34E4718D41

T_2 = 93987557C07B8102BA592949EB638F37

Two sequences of constant words C_0, C_1, ..., C_{79} and C'_0, C'_1, ..., C'_{79} are used in the round-function of Dedicated Hash-Function 1. They are defined as follows:

$C_i = K_1[0]$, $(0 \le i \le 15)$,

$C_i = K_1[1] \,\Psi$ 5A827999, $(16 \le i \le 31)$,

$C_i = K_1[2] \,\Psi$ 6ED9EBA1, $(32 \le i \le 47)$,

$C_i = K_1[3] \,\Psi$ 8F1BBCDC, $(48 \le i \le 63)$,

$C_i = K_1[0] \,\Psi$ A953FD4E, $(64 \le i \le 79)$,

$C'_i = K_1[1] \,\Psi$ 50A28BE6, $(0 \le i \le 15)$,

$C'_i = K_1[2] \,\Psi$ 5C4DD124, $(16 \le i \le 31)$,

$C'_i = K_1[3] \,\Psi$ 6D703EF3, $(32 \le i \le 47)$,

$C'_i = K_1[0] \,\Psi$ 7A6D76E9, $(48 \le i \le 63)$,

$C'_i = K_1[1]$, $(64 \le i \le 79)$.

6.4.3 Dedicated Hash-Function 2 (RIPEMD-128)

The 128-bit constant strings T_i for Dedicated Hash-Function 2 are defined as follows (in hexadecimal representation):

