INTERNATIONAL STANDARD

# ISO/IEC 10118-2

Third edition
2010-10-15

# Information technology — Security techniques — Hash-functions —

## Part 2: Hash-functions using an *n*-bit block cipher

*Technologies de l'information — Techniques de sécurité — Fonctions de brouillage —*

*Partie 2: Fonctions de brouillage utilisant un chiffrement par blocs de* n *bits*



Reference number
ISO/IEC 10118-2:2010(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

 

## Contents

Page





 

# Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. 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 10118-2 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

This third edition cancels and replaces the second edition (ISO/IEC 10118-2:2000), which has been technically revised. It also incorporates the Technical Corrigendum ISO/IEC 10118-2:2000/Cor.2:2007. The major change is that in the second edition the underlying block cipher used in the hash-functions was assumed to be Data Encryption Algorithm (DEA), whereas in the third edition it is assumed to be more secure block ciphers like Advanced Encryption Standard (AES) and other ciphers included in ISO/IEC 18033-3.

ISO/IEC 10118 consists of the following parts, under the general title *Information technology — Security techniques — Hash-functions*:

— *Part 1: General*

— *Part 2: Hash-functions using an n-bit block cipher*

— *Part 3: Dedicated hash-functions*

— *Part 4: Hash-functions using modular arithmetic*

## Introduction

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

The ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured the ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with the ISO and IEC. Information may be obtained from the ISO/IEC JTC 1 Patent database:

http://www.iso.org/patents

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

# Information technology — Security techniques — Hash-functions —

# Part 2: Hash-functions using an *n*-bit block cipher



## 1 Scope

This part of ISO/IEC 10118 specifies hash-functions which make use of an *n*-bit block cipher algorithm. They are therefore suitable for an environment in which such an algorithm is already implemented.

Four hash-functions are specified. The first provides hash-codes of length less than or equal to *n*, where *n* is the block-length of the underlying block cipher algorithm used. The second provides hash-codes of length less than or equal to 2*n*; the third provides hash-codes of length equal to 2*n*; and the fourth provides hash-codes of length 3*n*. All four of the hash-functions specified in this part of ISO/IEC 10118 conform to the general model specified in ISO/IEC 10118-1.

## 2 Normative references

The following referenced documents are indispensable for the application 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 10118-1:2000, *Information technology — Security techniques — Hash-functions — Part 1: General*

## 3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 10118-1 and the following apply.

**3.1**
**block**
string of bits of defined length

**3.2**
***n*-bit block cipher**
block cipher with the property that plaintext blocks and ciphertext blocks are *n* bits in length

[ISO/IEC 18033-3:2005]

**3.3**
**round function**
function $\phi(.,.)$ that transforms two binary strings of lengths $L_1$ and $L_2$ to a binary string of length $L_2$

NOTE The round function is used iteratively.

## 4 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviated terms given in ISO/IEC 10118-1 and the following apply.

| | |
|---|---|
| $B^L$ | When $n$ is even, the string composed of the $n/2$ leftmost bits of the block $B$. When $n$ is odd, the string composed of the $(n+1)/2$ leftmost bits of the block $B$ |
| $B^R$ | When $n$ is even, the string composed of the $n/2$ rightmost bits of the block $B$. When $n$ is odd, the string composed of the $(n-1)/2$ rightmost bits of the block $B$ |
| $B_x$ | When $B$ is a sequence of blocks and each block has $m$ bits, $B_x$ ($x \geq 0$) represents the $x$-th block of $B$. |
| $E_K(P)$ | $n$-bit block cipher algorithm taking the key $K$ and plaintext $P$ as input. It is recommended that the block cipher algorithms specified in ISO/IEC 18033-3 are used in the hash-functions. |
| $K$ | Key for the algorithm $E$ |
| $u$ or $u'$ | Function which takes as input an $n$-bit block and gives as output a key for the algorithm $E$. |

## 5 Use of the general model

The hash-functions specified in the next four clauses provide hash-codes $H$ of length $L_H$. The hash-functions conform to the general model specified in ISO/IEC 10118-1. For each of the four hash-functions that follow, it is therefore only necessary to specify

— the parameters $L_1$, $L_2$, $L_H$,

— the padding method,

— the initializing value $IV$,

— the round function $\phi$,

— the output transformation $T$.



## 6 Hash-function 1

### 6.1 General

The hash-function specified in this clause provides hash-codes of length $L_1$ and $L_2$ where $L_1$ and $L_2$ are equal to $n$. Some specific definitions that are required to specify hash-function 1 follow.

NOTE This hash-function is described in [5].

### 6.2 Parameter selection

The parameters $L_1$, $L_2$ and $L_H$ for the hash-function specified in this clause shall satisfy $L_1 = L_2 = n$, and $L_H$ is less than or equal to $n$.

## 6.3 Padding method

The selection of the padding method for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. As minimum requirements, the padding method shall output a set of $q$ blocks $D_1$, $D_2$, ..., $D_q$ where each block $D_j$ is of length $n$ and shall be such that each possible input produces distinct outputs. Examples of padding methods are presented in ISO/IEC 10118-1:2000, Annex A.

## 6.4 Initializing value

The selection of the *IV* for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. The *IV* shall be a bit-string of length $n$ and the value of the *IV* shall be agreed upon and fixed by users of the hash-function.

## 6.5 Round function

**Transformation *u*:**

Define a mapping $u$ from the ciphertext space.

The round function $\phi$ combines a padded data block $D_j$ (of $L_1 = n$-bits) with $H_{j-1}$, the previous output of the round function (of $L_2 = n$ bits), to yield $H_j$. As part of the round function it is necessary to choose a function $u$, which transforms an $n$-bit block into a key for use with the block cipher algorithm $E$. The selection of the function $u$ for use with this hash-function is outside the scope of this part of ISO/IEC 10118.

The round function itself is defined as follows:

Set $H_0$ equal to *IV*

$$\phi(D_j, H_{j-1}) = E_{K_j}(D_j) \oplus D_j$$

where $K_j = u(H_{j-1})$. The round function is shown in Figure 1.

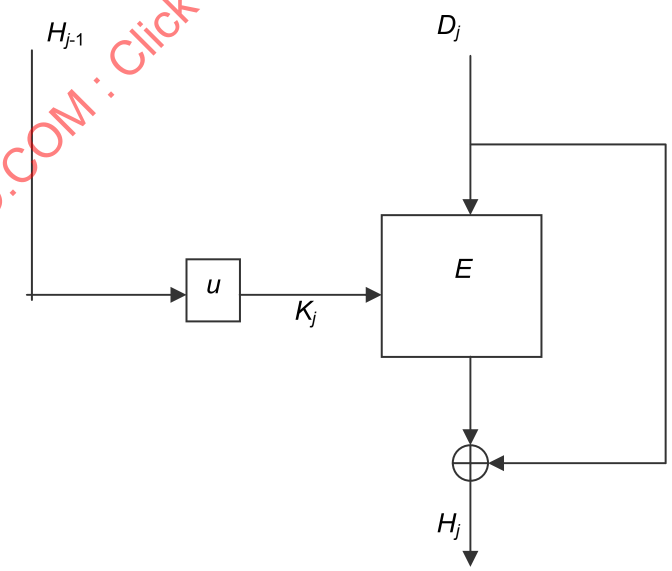


**Figure 1 — Round function of hash-function 1**

### 6.6 Output transformation

The output transformation $T$ is simply truncation, i.e., the hash-code $H$ is derived by taking the leftmost $L_H$ bits of the final output block $H_q$.

## 7 Hash-function 2

### 7.1 General

The hash-function specified in this clause provides hash-codes of length $L_1$ and $L_2$ where $L_1$ is equal to $n$ and $L_2$ is equal to $2n$. Some specific definitions that are required to specify hash-function 2 follow.

NOTE 1 This hash-function is described in [4].

NOTE 2 In [6], theoretical attacks on hash-function 2 have been reported: a collision attack, with $n = 128$, which has complexity $2^{124.5}$, and a preimage attack requiring complexity and space about $2^n$.

The only reason to keep hash-function 2 in this part of ISO/IEC 10118 is for compatibility with the existing applications.

### 7.2 Parameter selection

The parameters $L_1$, $L_2$ and $L_H$ for the hash-function specified in this clause shall satisfy $L_1 = n$, $L_2 = 2n$, and $L_H$ is less than or equal to $2n$.

### 7.3 Padding method

The selection of the padding method for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. As minimum requirements, the padding method shall output a set of $q$ blocks $D_1$, $D_2$, ..., $D_q$ where each block $D_j$ is of length $n$ and shall be such that each possible input produces distinct outputs. Examples of padding methods are presented in ISO/IEC 10118-1:2000, Annex A.

### 7.4 Initializing value

The selection of the *IV* (of length $2n$) for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. The *IV* shall be a bit-string of length $2n$ and the value of the *IV* shall be agreed upon and fixed by users of the hash-function. However, the *IV* shall be selected such that $u(IV^L)$ and $u'(IV^R)$ are different.

### 7.5 Round function

The round function $\phi$ combines a padded data block $D_j$ (of $L_1 = n$ bits) with $H_{j\text{-}1}$, the previous output of the round function (of $L_2 = 2n$ bits), to yield $H_j$. As part of the round function it is necessary to choose two transformations $u$ and $u'$. These transformations are used to transform an output block into two suitable $L_K$ bit keys for the algorithm $E$. The specification of $u$ and $u'$ is beyond the scope of this part of ISO/IEC 10118. However, it should be taken into consideration that the selection of $u$ and $u'$ is important for the security of the hash-function.

Set $H_0^L$ and $H_0^R$ equal to $IV^L$ and $IV^R$ respectively. The round function is defined in the following way, for $j = 1$ to $q$:

$$H_j = \phi(D_j, H_{j\text{-}1})$$

$$X = u(H_{j\text{-}1}^L) \text{ and } Y = u'(H_{j\text{-}1}^R)$$

$$B_j = E_X(D_j) \oplus D_j, \text{ and } B'_j = E_Y(D_j) \oplus D_j$$

$H_j^L = B_j^L \,||\, B'^R_j$ and $H_j^R = B'^L_j \,||\, B_j^R$

The round function is shown in Figure 2 where *X* and *Y* are replaced with $K^L_j$ and $K^R_j$ respectively.

## 7.6 Output transformation

If $L_H$ is even, the hash-code is the concatenation of the $L_H/2$ leftmost bits of $H_q^L$ and the $L_H/2$ leftmost bits of $H_q^R$. If $L_H$ is odd, the hash-code is the concatenation of the $(L_H+1)/2$ leftmost bits of $H_q^L$ and the $(L_H -1)/2$ leftmost bits of $H_q^R$.

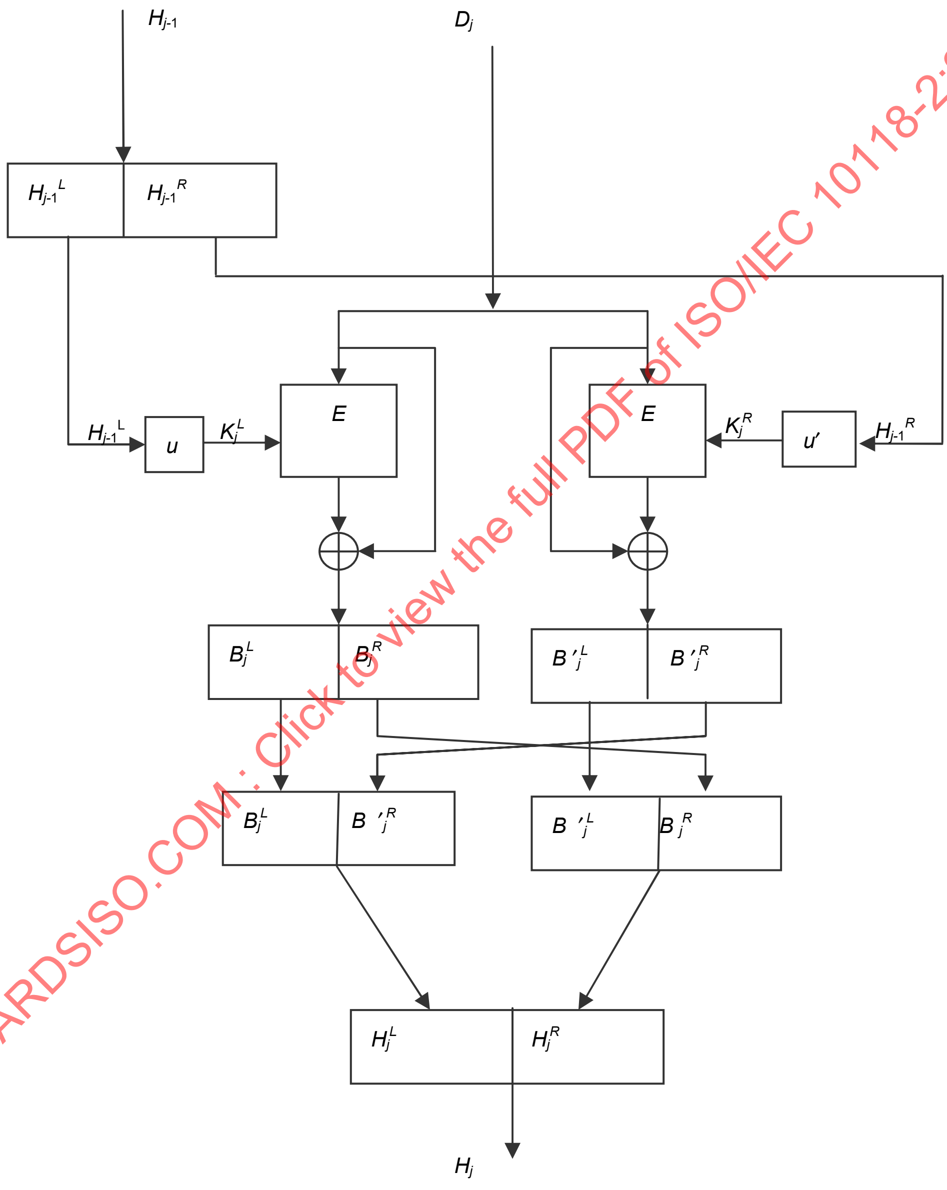


**Figure 2 — Round function of hash-function 2**



 

# 8 Hash-function 3

## 8.1 General

The hash-function specified in this clause provides hash-codes of length $L_H$, where $L_H$ is equal to $2n$ for even values of $n$. Some specific definitions that are required to specify hash-function 3 follow.

NOTE This hash-function is described in [1].

## 8.2 Parameter selection

The parameters $L_1$, $L_2$ and $L_H$ for the hash-function specified in this clause shall satisfy $L_1 = 4n$, $L_2 = 8n$, and $L_H = 2n$.

## 8.3 Padding method

The padding method for use with this hash-function shall be that specified in ISO/IEC 10118-1:2000, A.3, such that $r = n$.

## 8.4 Initializing value

The selection of the *IV* for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. The *IV* shall be a bit-string of length $8n$ and the value of the *IV* shall be agreed upon and fixed by users of the hash-function.

## 8.5 Round function

**Transformation *u*:**

Define eight mappings $u_1, u_2, \ldots, u_8$ from the ciphertext space to the key space, such that:

$u_i(C) \neq u_j(C)$, for all $i$, $j$ from the set $\{1,2,\ldots, 8\}$, $j \neq i$, and for all values of $C$

This can be achieved by fixing specific key bits: e.g., One can fix three key bits to the values 000, 001, ..., 111. Additional conditions might be imposed upon the mappings $u_i$, for example, to avoid the problems related to weak keys or complementation properties of the block cipher. Let $u_{j,i} = u_j(X_{j,i})$.

**Function $f_i$:**

Define the eight functions $f_i$ as follows:

$f_i(X,Y) = E_{ui(X)}(Y) \oplus Y$, $1 \leq i \leq 8$.

**Linear mapping $\beta$:**

Define the linear mapping $\beta$ that takes as input a $2n$-bit string $X = x_0||x_1||x_2||x_3$ and maps it to a $2n$-bit string $Y = y_0||y_1||y_2||y_3$ as follows:

$y_0 := x_0 \oplus x_3$

$y_1 := x_0 \oplus x_1 \oplus x_3$

$y_2 := x_1 \oplus x_2$

$y_3 := x_2 \oplus x_3$

Here $x_i$ and $y_j$ are $n/2$-bit strings.



 

The round function $\phi$ has eight parallel encryptions, and eight $n$-bit chaining variables $H_{j,1}$, $H_{j,2}$,…, $H_{j,8}$.

In every iteration, four $n$-bit data blocks, $D_{j,1}$, $D_{j,2}$, $D_{j,3}$, $D_{j,4}$ (of length $L_1 = 4n$ bits) are combined from the previous output of the round function, $H_{j\text{-}1,1}$, $H_{j\text{-}1,2}$,…, $H_{j\text{-}1,8}$ (of length $L_2 = 8n$ bits) to yield $H_{j,1}$, $H_{j,2}$,…, $H_{j,8}$ (of length $L_2 = 8n$ bits).

The round function is based on a linear mapping $\gamma_1$ , that takes as input 12 $n$-bit strings $I_1$, $I_2$,…, $I_{12}$ and maps them to eight $n$-bit strings $X_1$, $X_2$,…, $X_8$ and to eight $n$-bit strings $Y_1$, $Y_2$,…, $Y_8$. The mapping uses eight $2n$-bit auxiliary strings $R_0$, $R_1$, $M_0$, $M_1$,..., $M_5$. The mapping $\gamma_1$ is defined by the following steps:

i) Set $H_{0,1}$,..., $H_{0,8}$ in the way that $H_{0,1}||...||H_{0,8}$ is equal to $IV$.

ii) for $i$ = 0 to 5 do { $M_i^L := I_{2i+1}$ ; $M_i^R := I_{2i+2}$ ;}

$R_0 := 0$ ; $R_1 := 0$ ;

iii) for $i$ = 0 to 5 do {

$B := R_1 \oplus M_i$;

$R_1 := R_0 \oplus \beta(B)$ ;

$R_0 := B$ ; }

iv) for $i$ = 1 to 8 do { $X_i := I_i$ ;}

$Y_1 := R_0^L$ ;

$Y_2 := R_0^R$ ;

$Y_3 := R_1^L$ ;

$Y_4 := R_1^R$ ;

for $i$ = 1 to 4 do { $Y_{4+i} := I_{8+i}$ ;}

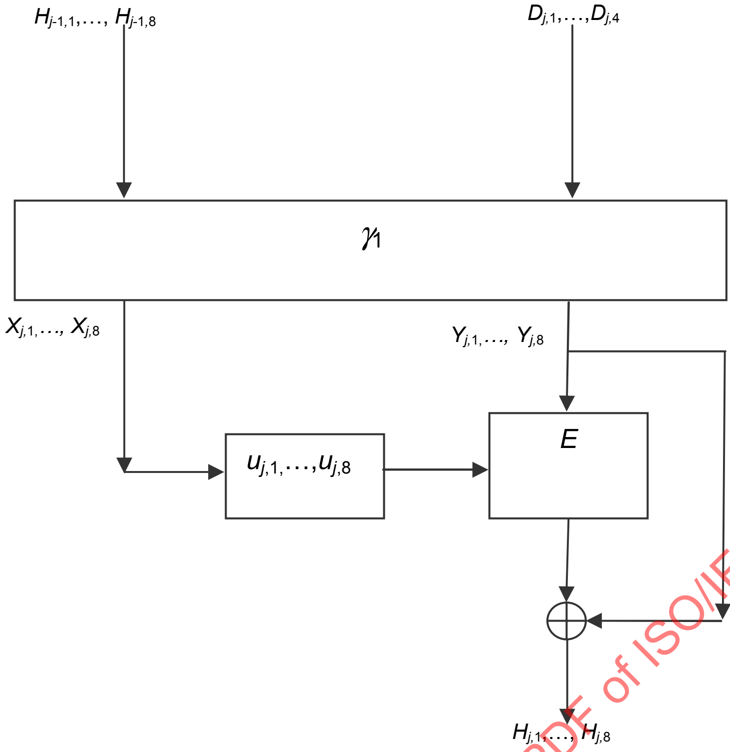


**Figure 3 — Round function of hash-function 3**



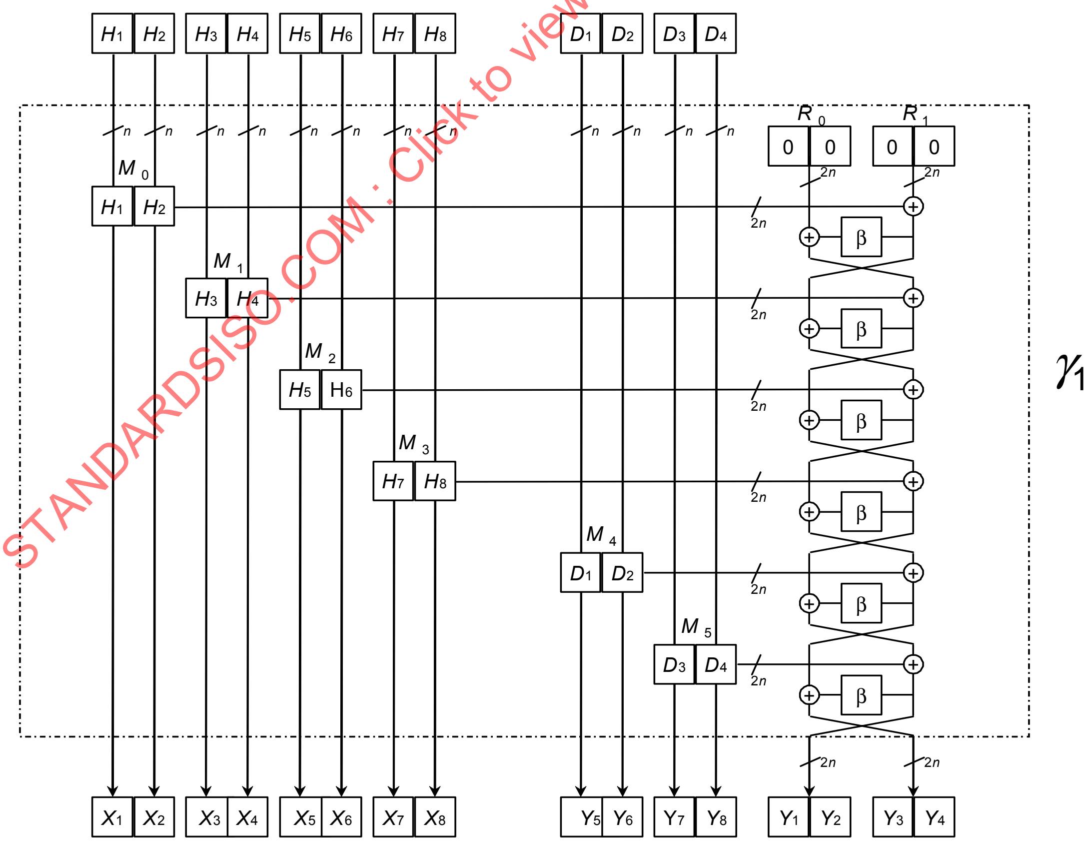


**Figure 4 — Linear mapping $\gamma_1$ of hash-function 3**

The round function has the following form ($1 \le j \le q$).

$(X_{j,1}, X_{j,2},\ldots, X_{j,8}, Y_{j,1}, Y_{j,2},\ldots, Y_{j,8}) := \gamma_1(H_{j-1,1}, H_{j-1,2},\ldots, H_{j-1,8}, D_{j,1}, D_{j,2}, D_{j,3}, D_{j,4})$;

for $i$ = 1 to 8 do { $H_{j,i} := f_i (X_{j,i}, Y_{j,i})$ ;}

The round function is illustrated in Figure 3 and the linear mapping $\gamma_1$ in Figure 4.

### 8.6 Output transformation

After processing of the padded message, the chaining variables have the values, $H_{q,1}, H_{q,2},\ldots, H_{q,8}$. Perform four additional iterations of the round function with the data inputs

$D_{q+1,i} = H_{q,i},\ 1 \le i \le 4$

$D_{q+2,i} = H_{q,i+4},\ 1 \le i \le 4$

$D_{q+3,i} = H_{q,i},\ 1 \le i \le 4$

$D_{q+4,i} = H_{q,i+4},\ 1 \le i \le 4$.

The output $L_H$ of the hash-function then consists of $H_{q+4,1} \,||\, H_{q+4,2}$. The output transformation requires 26 encryptions (in the last iteration only two encryptions need to be performed).

## 9 Hash-function 4

### 9.1 General

The hash-function specified in this clause provides hash-codes of length $L_H$, where $L_H$ is equal to 3$n$ for even values of $n$.

NOTE This hash-function is described in [2].

### 9.2 Parameter selection

The parameters $L_1$ and $L_2$ and $L_H$ for the hash-function specified in this clause shall satisfy $L_1 = 3n$, $L_2 = 9n$, and $L_H = 3n$.

### 9.3 Padding method

The padding method for use with this hash-function shall be that specified in ISO/IEC 10118-1:2000, A.3, such that $r = n$.

### 9.4 Initializing value

The selection of the *IV* for use with this hash-function is beyond the scope of this part of ISO/IEC 10118. The *IV* shall be a bit-string of length 9$n$ and the value of the *IV* shall be agreed upon and fixed by users of the hash-function.

### 9.5 Round function

**Transformation *u*:**

Define nine mappings $u_1, u_2, \ldots, u_9$ from the ciphertext space to the key space, such that, For all $i$, $j$ from the set {1,2,…,9}, $j \neq i$, $u_i(C) \neq u_j(C)$ for all values of $C$

This can be achieved by fixing specific key bits: e.g., One can fix four key bits to the values 0000, 0001, ..., 1000. Additional conditions might be imposed upon the mappings $u_i$, for example, to avoid the problems related to weak keys or complementation properties of the block cipher.

**Function $f_i$:**

Define the nine functions $f_i$ as follows:

$$f_i(X,Y) = E_{ui(X)}(Y) \oplus Y, \ 1 \leq i \leq 9.$$

**Linear mapping $\beta$:**

See 8.1 for specific definitions relevant to this hash-function.

The round function $\phi$ has nine parallel encryptions, and nine $n$-bit chaining variables, $H_{j,1}$, $H_{j,2}$,…, $H_{j,9}$.

In every iteration, three $n$-bit data blocks, $D_{j,1}$, $D_{j,2}$, $D_{j,3}$ (of length $L_1 = 3n$ bits) are combined from the previous output of the round function, $H_{j\text{-}1,1}$, $H_{j\text{-}1,2}$,…, $H_{j\text{-}1,9}$ (of length $L_2 = 9n$ bits) to yield $H_{j,1}$, $H_{j,2}$,…, $H_{j,9}$ (of length $L_2 = 9n$ bits).

The round function is based on a linear mapping $\gamma_2$, that takes as input 12 $n$-bit strings $I_1$, $I_2$,…, $I_{12}$ and maps them to nine $n$-bit strings $X_1$, $X_2$,…, $X_9$ and to nine $n$-bit strings $Y_1$, $Y_2$,…, $Y_9$. The mapping uses nine $2n$-bit auxiliary strings $R_0$, $R_1$, $R_2$, $M_0$, $M_1$,..., $M_5$. The mapping $\gamma_2$ is defined by the following steps:

i) Set $H_{0,1}$,..., $H_{0,9}$ in the way that $H_{0,1}||...||H_{0,9}$ is equal to $IV$

ii) for $i$ = 0 to 5 do { $M_i^L := I_{2i+1}$ ; $M_i^R := I_{2i+2}$ ;}

$R_0 := 0$ ; $R_1 := 0$ ;$R_2 := 0$ ;

iii) for $i$ = 0 to 5 do {

$B := R_2 \oplus M_i$ ;

$U := \beta(B)$ ;

$R_2 := R_1 \oplus U$ ;

$R_1 := R_0 \oplus U$ ;

$R_0 := B$; }

iv) for $i$ = 1 to 9 do { $X_i := I_i$ ;}

$Y_1 := R_0^L$ ;

$Y_2 := R_0^R$ ;

$Y_3 := R_1^L$ ;

$Y_4 := R_1^R$ ;

$Y_5 := R_2^L$ ;

$Y_6 := R_2^R$ ;

for $i$ = 1 to 3 do { $Y_{6+i} := I_{9+i}$ ;}



 

The round function has the following form ($1 \le j \le q$).

$$(X_{j,1}, X_{j,2},\ldots, X_{j,9}, Y_{j,1}, Y_{j,2},\ldots, Y_{j,9}) := \gamma_2(H_{j-1,1}, H_{j-1,2},\ldots, H_{j-1,9}, D_{j,1}, D_{j,2}, D_{j,3});$$

for $i$ = 1 to 9 do { $H_{j,i} := f_i\,(X_{j,i}, Y_{j,i})$ ;}

The round function is illustrated in Figure 5 and the linear mapping $\gamma_2$ in Figure 6.

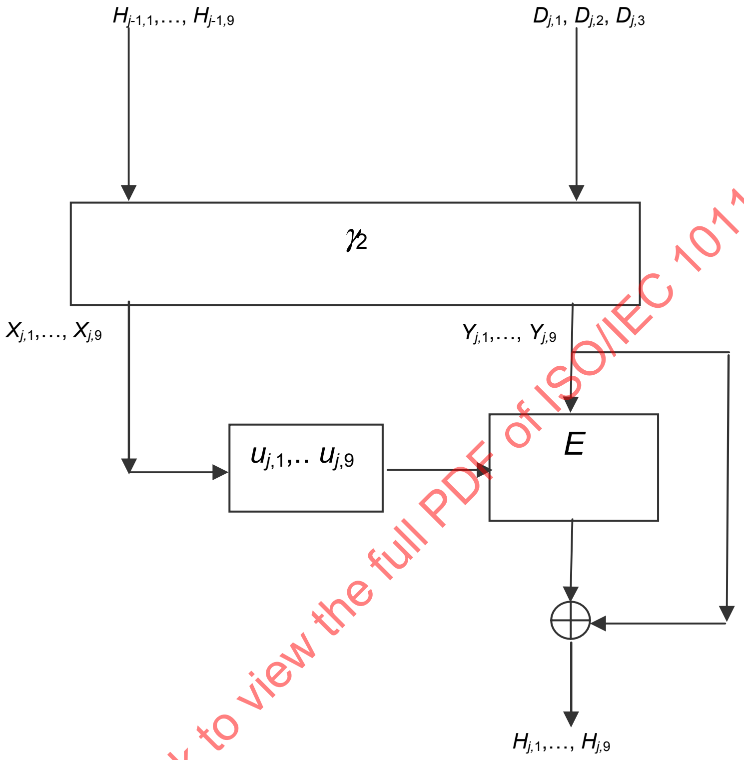


**Figure 5 — Round function of hash-function 4**

## 9.6 Output transformation

After processing of the padded message, the chaining variables have the values $H_{q,1}$, $H_{q,2},\ldots, H_{q,9}$. Perform four additional iterations of the round function with the message inputs

$D_{q+1,i} = H_{q,i}$, $1 \le i \le 3$

$D_{q+2,i} = H_{q,i+3}$, $1 \le i \le 3$

$D_{q+3,i} = H_{q,i+6}$, $1 \le i \le 3$

$D_{q+4,i} = H_{q,i}$, $1 \le i \le 3$.

The output of the hash-function then consists of $H_{q+4,1} \| H_{q+4,2} \| H_{q+4,3}$ The output transformation requires 30 encryptions (in the last iteration only three encryptions need to be performed).

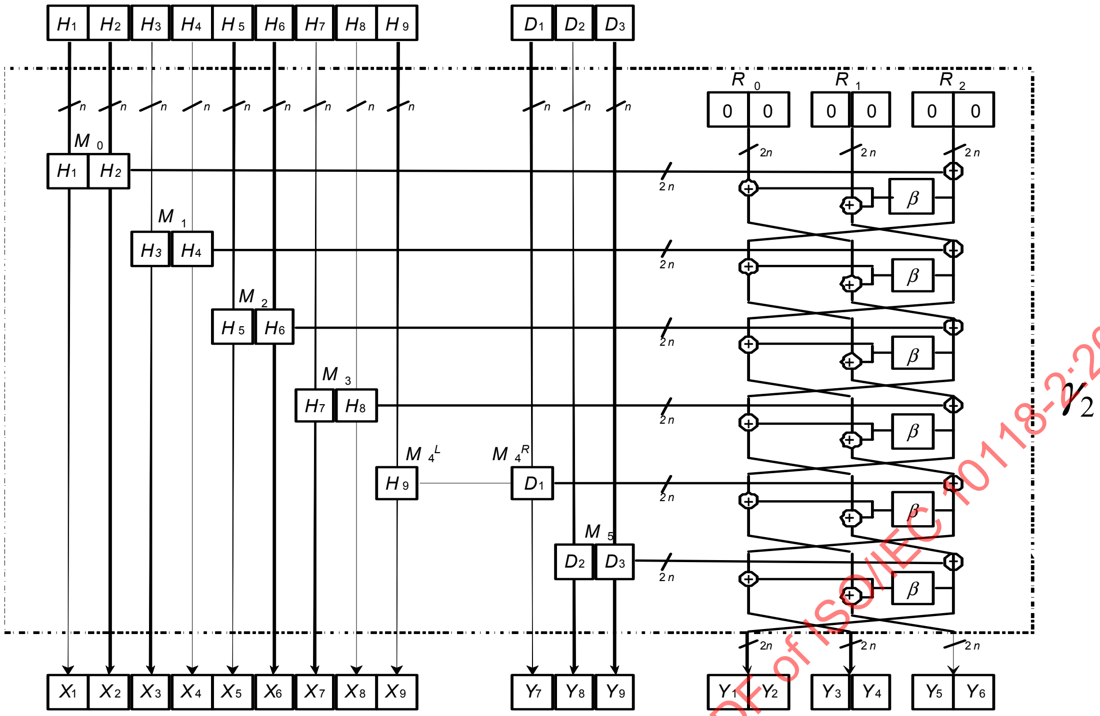


**Figure 6 — Linear mapping $\gamma_2$ of hash-function 4**

# Annex A
(informative)

# Use of AES

## A.1 General

This annex presents a way of using the AES (ISO/IEC 18033-3) in conjunction with the hashing operations specified in this part of ISO/IEC 10118. The parameter for AES is $n$ = 128. And the length of $K$ is 128 bits.

## A.2 Hash-function 1

$IV$ should be equal to ‘52525252525252525252525252525252’ (in hexadecimal notation).

The transformation $u$ should be chosen as follows. Let $X$ be the binary decomposition of a 128-bit string. Then $Y = u(X) = X$.

NOTE It is believed that finding collisions for the round function and for the hash-function requires $2^{64}$ AES encryptions.

## A.3 Hash-function 2

$IV^L$ should be equal to ‘52525252525252525252525252525252’ (in hexadecimal notation).

$IV^R$ should be equal to ‘25252525252525252525252525252525’ (in hexadecimal notation).

The transformation $u$ should be chosen as follows. Let $X = x_1x_2\ldots x_{128}$ be the binary decomposition of a 128-bit string $X$. Then $Y = u(X)$ is the string obtained after forcing the bit $x_1$ to the value ‘0’. The result is: $Y = 0x_2x_3\ldots x_{127}x_{128}$. The transformation $u'$ should be chosen as follows. Then $Y = u'(X)$ is the string obtained after forcing the bit $x_1$ to the value ‘1’. The result is: $Y = 1x_2x_3\ldots x_{127}x_{128}$.

## A.4 Hash-function 3

$IV_1, IV_2,\ldots, IV_8$ shall be equal to `52525252525252525252525252525252' (in hexadecimal notation).

The transformation $u_1, u_2, \ldots u_8$ shall be chosen as follows. Let $X = x_1x_2\ldots x_{128}$ be the binary decomposition of a 128-bit string $X$. Then $Y = u_i(X)$ is the string obtained after forcing the bits $x_1$, $x_2$, $x_3$ to the values given in Table 1.

**Table A.1 — Hash-function 3: Values of key bits no. 1, 2, 3 in the eight subfunctions**

| Subfunction *i* | Subfunction *i* |
|---|---|
| 1 | 000 |
| 2 | 001 |
| 3 | 010 |
| 4 | 011 |
| 5 | 100 |
| 6 | 101 |
| 7 | 110 |
| 8 | 111 |



## A.5 Hash-function 4

$IV_1$, $IV_2$, ... $IV_9$ shall be equal to `52525252525252525252525252525252' (in hexadecimal notation).

The transformation $u_1$, $u_2$, …$u_9$ shall be chosen as follows. Let $X = x_1x_2…x_{128}$ be the binary decomposition of a 128-bit string $X$. Then $Y = u_i(X)$ is the string obtained after forcing the bits $x_1$, $x_2$, $x_3$, $x_4$ to the values given in Table 2.

**Table A.2 — Hash-function 4: Values of key bits no. 1, 2, 3, and 4 in the nine subfunctions**

| Subfunction *i* | Subfunction *i* |
|---|---|
| 1 | 0000 |
| 2 | 0001 |
| 3 | 0010 |
| 4 | 0011 |
| 5 | 0100 |
| 6 | 0101 |
| 7 | 0110 |
| 8 | 0111 |
| 9 | 1000 |

# Annex B
(informative)

# Examples

## B.1 General

This annex gives examples for the computation of a hash-code for all the hash-functions specified in Clauses 6-9 of this part of ISO/IEC 10118, the block cipher specified in Annex A of this part of ISO/IEC 10118 and selected padding methods specified in Annex A of ISO/IEC 10118-1:2000.

The data string is the 7-bit ASCII code as described in [3] (no parity) for "Now_is_the_time_for_all_", where "_" denotes a blank in hexadecimal notation:

'4e6f772069732074686520746 96d6520666f7220616c6c20'

## B.2 Hash-function 1

See A.2.

Padding method 1

| $J$ | $D_j$ | $H_{j-1}$ | $H_j$ |
|---|---|---|---|
| 1 | 4e6f772069732074 | 5252525252525252 | 113fff9a8dfe98c1 |
| | 68652074696d6520 | 5252525252525252 | 6ed8932aff2dfd9e |
| 2 | 666f7220616c6c20 | 113fff9a8dfe98c1 | 08851dc2ef0dd720 |
| | 0000000000000000 | 6ed8932aff2dfd9e | b76972c33761b988 |

Padding method 2

| $J$ | $D_j$ | $H_{j-1}$ | $H_j$ |
|---|---|---|---|
| 1 | 4e6f772069732074 | 5252525252525252 | 113fff9a8dfe98c1 |
| | 68652074696d6520 | 5252525252525252 | 6ed8932aff2dfd9e |
| 2 | 666f7220616c6c20 | 113fff9a8dfe98c1 | 2bf0f0e63c36e020 |
| | 8000000000000000 | 6ed8932aff2dfd9e | 780d4835b98590ea |

## B.3 Hash-function 2

Padding method 1

| $j$ | $D_j$ | $H^L_{j-1}$ | $H^R_{j-1}$ |
|---|---|---|---|
| 1 | 4e6f772069732074 | 5252525252525252 | 2525252525252525 |
| | 68652074696d6520 | 5252525252525252 | 2525252525252525 |
| 2 | 666f7220616c6c20 | c84daaccf3ea34a4 | 5f66afab4d7e2f20 |
| | 0000000000000000 | 234c789d2e61f2e3 | b4df8be09cdcd69b |

| $J$ | $D_j$ | $H^L_j$ | $H^R_j$ |
|---|---|---|---|
| 1 | | c84daaccf3ea34a4 | 5f66afab4d7e2f20 |
| | | 234c789d2e61f2e3 | b4df8be09cdcd69b |
| 2 | | 4a56ed816a52ca1f | e88d9cdbcc55850c |
| | | 6d89483b781ec276 | e2ced29925a6f64b |

Padding method 2

| $j$ | $D_j$ | $H^L_{j-1}$ | $H^R_{j-1}$ |
|---|---|---|---|
| 1 | 4e6f772069732074 | 5252525252525252 | 2525252525252525 |
| | 68652074696d6520 | 5252525252525252 | 2525252525252525 |
| 2 | 666f7220616c6c20 | c84daaccf3ea34a4 | 5f66afab4d7e2f20 |
| | 8000000000000000 | 234c789d2e61f2e3 | b4df8be09cdcd69b |

| $j$ | $D_j$ | $H^L_j$ | $H^R_j$ |
|---|---|---|---|
| 1 | | c84daaccf3ea34a4 | 5f66afab4d7e2f20 |
| | | 234c789d2e61f2e3 | b4df8be09cdcd69b |
| 2 | | ca3eafd2bf937bfe | fff352b5d02670c6 |
| | | 8c11b00d4543a1cd | d2c0d86822aaeed5 |

## B.4 Hash-function 3

Padding method 3.

| $D_{1,1}, D_{1,2}, D_{1,3}, D_{1,4}$ | $H_{0,1}, H_{0,2}, H_{0,3}, H_{0,4}, H_{0,5}, H_{0,6}, H_{0,7}, H_{0,8}$ | $H_{1,1}, H_{1,2}, H_{1,3}, H_{1,4}, H_{1,5}, H_{1,6}, H_{1,7}, H_{1,8}$ |
|---|---|---|
| 4e6f772069732074 | 5252525252525252 | 218f923b370be9a8 |
| 68652074696d6520 | 5252525252525252 | 920562614859df7e |
| 666f7220616c6c20 | 5252525252525252 | 8a26575e97be292b |
| 8000000000000000 | 5252525252525252 | 4aa47e1e8206a2f7 |
| 0000000000000000 | 5252525252525252 | 1230f84cffde57fd |
| 0000000000000000 | 5252525252525252 | 988b6063b3b2d3cf |
| 0000000000000000 | 5252525252525252 | ed5d056582182065 |
| 00000000000000c0 | 5252525252525252 | c4fb4f2966b27058 |
| | 5252525252525252 | 6eb4beb7c1b2141f |
| | 5252525252525252 | 268ba3326336413b |
| | 5252525252525252 | c90a43026a380748 |
| | 5252525252525252 | dcc2521dd2cf3e0d |
| | 5252525252525252 | c9851c64fef13ad7 |
| | 5252525252525252 | 11d1a801e2ac052d |
| | 5252525252525252 | 1e79a495366b8cd9 |
| | 5252525252525252 | ca1eca9844dc09e5 |

| $D_{2,1}, D_{2,2}, D_{2,3}, D_{2,4}$ | $H_{1,1}, H_{1,2}, H_{1,3}, H_{1,4}, H_{1,5}, H_{1,6}, H_{1,7}, H_{1,8}$ | $H_{2,1}, H_{2,2}, H_{2,3}, H_{2,4}, H_{2,5}, H_{2,6}, H_{2,7}, H_{2,8}$ |
|---|---|---|
| 218f923b370be9a8 | 218f923b370be9a8 | e05e2407707fa017 |
| 920562614859df7e | 920562614859df7e | 44e1156f9ba14704 |
| 8a26575e97be292b | 8a26575e97be292b | 2d6d30d47c1736d0 |
| 4aa47e1e8206a2f7 | 4aa47e1e8206a2f7 | 597e1750720f4247 |
| 1230f84cffde57fd | 1230f84cffde57fd | 01af4028b2023819 |
| 988b6063b3b2d3cf | 988b6063b3b2d3cf | 40db2f9056889610 |
| ed5d056582182065 | ed5d056582182065 | 450cebc815285244 |
| c4fb4f2966b27058 | c4fb4f2966b27058 | 343f87f2aba57fe8 |
| | 6eb4beb7c1b2141f | ccc71fdf4a500dbe |
| | 268ba3326336413b | 6fc9f91932ec9cdd |
| | c90a43026a380748 | 7332ba30f8c7fab0 |
| | dcc2521dd2cf3e0d | 55f859f7c74d4589 |
| | c9851c64fef13ad7 | 9c8e431285712ab2 |
| | 11d1a801e2ac052d | 675dc2734f1bac40 |
| | 1e79a495366b8cd9 | 96c578ed26e38a77 |
| | ca1eca9844dc09e5 | d62f10e896523889 |

| $D_{3,1}, D_{3,2}, D_{3,3}, D_{3,4}$ | $H_{2,1}, H_{2,2}, H_{2,3}, H_{2,4}, H_{2,5}, H_{2,6}, H_{2,7}, H_{2,8}$ | $H_{3,1}, H_{3,2}, H_{3,3}, H_{3,4}, H_{3,5}, H_{3,6}, H_{3,7}, H_{3,8}$ |
|---|---|---|
| 6eb4beb7c1b2141f | e05e2407707fa017 | f9c0dfe1c95010b2 |
| 268ba3326336413b | 44e1156f9ba14704 | 8f8bcb23eef6daa2 |
| c90a43026a380748 | 2d6d30d47c1736d0 | ea0ad33cc80231dc |
| dcc2521dd2cf3e0d | 597e1750720f4247 | 9790b34d5ec03c0e |
| c9851c64fef13ad7 | 01af4028b2023819 | 861bafcee007b4cd |
| 11d1a801e2ac052d | 40db2f9056889610 | 6dbf787a2654dcf7 |
| 1e79a495366b8cd9 | 450cebc815285244 | 977028407cb93345 |
| ca1eca9844dc09e5 | 343f87f2aba57fe8 | b163d9e3a005ff7f |
| | ccc71fdf4a500dbe | 5688331e36f098bc |
| | 6fc9f91932ec9cdd | 75d83967830d4086 |
| | 7332ba30f8c7fab0 | 6196b975ab6fee13 |
| | 55f859f7c74d4589 | fff012673153fd87 |
| | 9c8e431285712ab2 | 7f021bdfc73f846f |
| | 675dc2734f1bac40 | 8e485a4fe0fa1644 |
| | 96c578ed26e38a77 | 6662de8b03a6b64d |
| | d62f10e896523889 | 5fb159f1adf26d5d |

| $D_{4,1}, D_{4,2}, D_{4,3}, D_{4,4}$ | $H_{3,1}, H_{3,2}, H_{3,3}, H_{3,4},$ $H_{3,5}, H_{3,6}, H_{3,7}, H_{3,8}$ | $H_{4,1}, H_{4,2}, H_{4,3}, H_{4,4},$ $H_{4,5}, H_{4,6}, H_{4,7}, H_{4,8}$ |
|---|---|---|
| 218f923b370be9a8 | f9c0dfe1c95010b2 | 5a3824dd343c1c91 |
| 920562614859df7e | 8f8bcb23eef6daa2 | cd5ddb98d4c0da49 |
| 8a26575e97be292b | ea0ad33cc80231dc | f929439b08ccf36b |
| 4aa47e1e8206a2f7 | 9790b34d5ec03c0e | 14ae2fce0d7e2c76 |
| 1230f84cffde57fd | 861bafcee007b4cd | ff001505ccb8b3a6 |
| 988b6063b3b2d3cf | 6dbf787a2654dcf7 | 0a3f4674496b91f1 |
| ed5d056582182065 | 977028407cb93345 | baa3b2b7746c548e |
| c4fb4f2966b27058 | b163d9e3a005ff7f | 0676aff6595c6e11 |
| | 5688331e36f098bc | 3be7c5a7d47b7bbb |
| | 75d83967830d4086 | f3f8df583e5633d1 |
| | 6196b975ab6fee13 | 4a87df6f5892eece |
| | fff012673153fd87 | 73bf2cd832bfc181 |
| | 7f021bdfc73f846f | fead044cd64757ed |
| | 8e485a4fe0fa1644 | 74477d02b1ecfff2 |
| | 6662de8b03a6b64d | a836d76f2117e1f1 |
| | 5fb159f1adf26d5d | faa55af85c67f5b2 |

| $D_{5,1}, D_{5,2}, D_{5,3}, D_{5,4}$ | $H_{4,1}, H_{4,2}, H_{4,3}, H_{4,4}, H_{4,5}, H_{4,6}, H_{4,7}, H_{4,8}$ | $H_{5,1}, H_{5,2}, H_{5,3}, H_{5,4}, H_{5,5}, H_{5,6}, H_{5,7}, H_{5,8}$ |
|---|---|---|
| 218f923b370be9a8 | 5a3824dd343c1c91 | 35af124f4845eb47 |
| 920562614859df7e | cd5ddb98d4c0da49 | 256a959eb84554e0 |
| 8a26575e97be292b | f929439b08ccf36b | 3b78dd0c4a1d9bf3 |
| 4aa47e1e8206a2f7 | 14ae2fce0d7e2c76 | 6c4a4010aa41d8c5 |
| 1230f84cffde57fd | ff001505ccb8b3a6 | 2cd3c769464dc946 |
| 988b6063b3b2d3cf | 0a3f4674496b91f1 | 6beb79285da9e383 |
| ed5d056582182065 | baa3b2b7746c548e | 0c0afc2e1fba5338 |
| c4fb4f2966b27058 | 0676aff6595c6e11 | d1ae7bff8f000138 |
| | 3be7c5a7d47b7bbb | 08b7bf8d2761947e |
| | f3f8df583e5633d1 | fb950243c0980b87 |
| | 4a87df6f5892eece | 683447121ef47b19 |
| | 73bf2cd832bfc181 | b043076cf44d931b |
| | fead044cd64757ed | af4f446c2e2cf09d |
| | 74477d02b1ecfff2 | c73cd1a4383d1f26 |
| | a836d76f2117e1f1 | 6dfaa1bfc27b6606 |
| | faa55af85c67f5b2 | 7c88bc330ec798f5 |

Hash-code:

'35af124f4845eb47256a959eb84554e03b78dd0c4a1d9bf36c4a4010aa41d8c5'

## B.5 Hash-function 4

Padding method 3

| $D_{1,1}$, $D_{1,2}$, $D_{1,3}$ | $H_{0,1}$, $H_{0,2}$, $H_{0,3}$, $H_{0,4}$, $H_{0,5}$, $H_{0,6}$, $H_{0,7}$, $H_{0,8}$, $H_{0,9}$ | $H_{1,1}$, $H_{1,2}$, $H_{1,3}$, $H_{1,4}$, $H_{1,5}$, $H_{1,6}$, $H_{1,7}$, $H_{1,8}$, $H_{1,9}$ |
|---|---|---|
| 4e6f772069732074 | 5252525252525252 | 35373c5888be113e |
| 68652074696d6520 | 5252525252525252 | 685b8c0a1c87af82 |
| 666f7220616c6c20 | 5252525252525252 | 10322300513de264 |
| 8000000000000000 | 5252525252525252 | f47883512306b378 |
| 0000000000000000 | 5252525252525252 | 3ccf820b5a6395f1 |
| 00000000000000c0 | 5252525252525252 | 6af97874f3ced2e5 |
| | 5252525252525252 | 64dd22a5fc7673d9 |
| | 5252525252525252 | 0deeed557012a0a0 |
| | 5252525252525252 | 546cff0e61ff9597 |
| | 5252525252525252 | 388dbe3bdc3ad0aa |
| | 5252525252525252 | 276b38ca16da0733 |
| | 5252525252525252 | 1efc14b2188b4510 |
| | 5252525252525252 | 9943f2aa62125370 |
| | 5252525252525252 | c7ee32c7e95ed829 |
| | 5252525252525252 | cec2c97e170a75ec |
| | 5252525252525252 | 2604a9fda4811e4b |
| | 5252525252525252 | 70f5c66e35c89830 |
| | 5252525252525252 | 3143d2449a614041 |