BRITISH STANDARD

BS ISO/IEC 10118-3:2004 +A1:2006

Incorporating corrigendum December 2011

Information technology — Security techniques — Hash-functions —

Part 3: Dedicated hash-functions

ICS 35.040

NO COPYING WITHOUT BSI PERMISSION EXCEPT AS PERMITTED BY COPYRIGHT LAW

BSI Document Sharing Hub British Standards

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

National foreword

This British Standard is the UK implementation of ISO/IEC 10118-3:2004+A1:2006, incorporating corrigendum December 2011. It supersedes BS ISO/IEC 10118-3:2003, which is withdrawn.

The start and finish of text introduced or altered by amendment is indicated in the text by tags. Tags indicating changes to ISO/IEC text carry the number of the amendment. For example, text altered by ISO/IEC amendment 1 is indicated in the text by [A1] ⟨A1⟩.

Technical Corrigendum December 2011 altered text in Annex A. It replaced all occurrences of "Y_0, Y_1, …, Y_7" with "X_0, X_1, …, X_7" in Clauses A.4.3, A.4.8, A.5.3, A.5.10, A.6.3 and A.6.10. It also altered the text of Clause A.6.7.

The UK participation in its preparation was entrusted to Technical Committee IST/33, IT – Security techniques.

A list of organizations represented on this committee can be obtained on request to its secretary.

This publication does not purport to include all the necessary provisions of a contract. Users are responsible for its correct application.

Compliance with a British Standard cannot confer immunity from legal obligations.

Amendments/corrigenda issued since publication

Amd. No.	Date	Comments
16227	28 April 2006	Implementation of amendment 1:2006
	31 October 2010	Implementation of national corrigendum. Removal of incorrect amendment 1:2006
	31 May 2012	Implementation of ISO/IEC amendment 1:2006 and ISO/IEC corrigendum December 2011

This British Standard was published under the authority of the Standards Policy and Strategy Committeeon 3 July 2003

© The British Standards Institution 2012. Published by BSI Standards Limited 2012.

ISBN 978 0 580 73586 8

Documents
Document Sharing Hub

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

INTERNATIONAL STANDARD

ISO/IEC 10118-3

Third edition
2004-03-01

Information technology — Security techniques — Hash-functions —

Part 3: Dedicated hash-functions

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

Partie 3: Fonctions de brouillage dédiées

Reference number
ISO/IEC 10118-3:2004(E)

Documents
Document Sharing Hub
© ISO/IEC 2004

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

PDF disclaimer

This PDF file may contain embedded typefaces. In accordance with Adobe's licensing policy, this file may be printed or viewed but shall not be edited unless the typefaces which are embedded are licensed to and installed on the computer performing the editing. In downloading this file, parties accept therein the responsibility of not infringing Adobe's licensing policy. The ISO Central Secretariat accepts no liability in this area.

Adobe is a trademark of Adobe Systems Incorporated.

Details of the software products used to create this PDF file can be found in the General Info relative to the file; the PDF-creation parameters were optimized for printing. Every care has been taken to ensure that the file is suitable for use by ISO member bodies. In the unlikely event that a problem relating to it is found, please inform the Central Secretariat at the address given below.

© ISO/IEC 2004

All rights reserved. Unless otherwise specified, no part of this publication may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying and microfilm, without permission in writing from either ISO at the address below or ISO's member body in the country of the requester.

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

Documents
Document Sharing Hub
© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

Contents

Page

Foreword ... iv

Introduction ... v

1 **Scope** ... 1

2 **Normative references** ... 1

3 **Terms and definitions** ... 1

4 **Symbols (and abbreviated terms)** ... 1
4.1 **Symbols specified in ISO/IEC 10118-1** ... 1
4.2 **Symbols specific to this part** ... 2

5 **Requirements** ... 3

6 **Model for dedicated hash-functions** ... 4

7 **Dedicated Hash-Function 1 (RIPEMD-160)** ... 4
7.1 **Parameters, functions and constants** ... 4
7.2 **Padding method** ... 7
7.3 **Description of the round-function** ... 7

8 **Dedicated Hash-Function 2 (RIPEMD-128)** ... 8
8.1 **Parameters, functions and constants** ... 8
8.2 **Padding method** ... 9
8.3 **Description of the round-function** ... 9

9 **Dedicated Hash-Function 3 (SHA-1)** ... 10
9.1 **Parameters, functions and constants** ... 10
9.2 **Padding method** ... 11
9.3 **Description of the round-function** ... 12

10 **Dedicated Hash-Function 4 (SHA-256)** ... 13
10.1 **Parameters, functions and constants** ... 13
10.2 **Padding method** ... 14
10.3 **Description of the round-function** ... 14

11 **Dedicated Hash-Function 5 (SHA-512)** ... 15
11.1 **Parameters, functions and constants** ... 15
11.2 **Padding method** ... 17
11.3 **Description of the round-function** ... 17

12 **Dedicated Hash-Function 6 (SHA-384)** ... 18
12.1 **Parameters, functions and constants** ... 18
12.2 **Padding method** ... 19
12.3 **Description of the round-function** ... 19

13 **Dedicated Hash-Function 7 (WHIRLPOOL)** ... 19
13.1 **Parameters, functions and constants** ... 19
13.2 **Padding method** ... 21
13.3 **Description of the round-function** ... 22

14 **Dedicated Hash-Function 8 (SHA-224)** ... 23
14.1 **Parameters, functions and constants** ... 23
14.2 **Padding method** ... 24
14.3 **Description of the round-function** ... 24

Annex A (informative) **Examples** ... 25

Annex B (informative) **Formal specifications** ... 88

Annex C (normative) **ASN.1 Module** ... 101

Bibliography ... 104

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

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 10118-3 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 10118-3:2003), which has been technically revised.

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*

© ISO 2012
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

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 International Standard may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured ISO and IEC that he is willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with the ISO and IEC. Information may be obtained from:

ISO/IEC JTC 1/SC 27 Standing Document 8 (SD8) "Patent Information"

Standing Document 8 (SD8) is publicly available at: http://www.ni.din.de/sc27

Attention is drawn to the possibility that some of the elements of this International Standard 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.

© ISO 2012

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

Documents
Document Sharing Hub
© ISO 2012

Information technology — Security techniques — Hash-functions —

Part 3: Dedicated hash-functions

1 Scope

This part of ISO/IEC 10118 specifies dedicated hash-functions, i.e. specially designed hash-functions. The hash-functions in this part of ISO/IEC 10118 are based on the iterative use of a round-function. Seven distinct round-functions are specified, giving rise to distinct dedicated hash-functions.

The first and third dedicated hash-functions in Clauses 7 and 9 respectively provide hash-codes of lengths up to 160 bits; the second in Clause 8 provides hash-codes of lengths up to 128 bits; the fourth in Clause 10 provides hash-codes of lengths up to 256 bits; the sixth in Clause 12 provides hash-codes of a fixed length, 384 bits; and the fifth and seventh in Clauses 11 and 13 respectively provide hash-codes of lengths up to 512 bits.

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 part of ISO/IEC 10118, the definitions given in ISO/IEC 10118-1 and the following apply.

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

3.2
word
a string of 32 bits used in dedicated hash-functions 1, 2, 3 and 4 of Clauses 7, 8, 9 and 10 respectively, or a string of 64 bits used in dedicated hash-functions 5 and 6 of Clauses 11 and 12 respectively

3.3
matrix
an 8 by 8 matrix in which each entry is a string of 8 bits used in dedicated hash-function 7 of Clause 13

4 Symbols (and abbreviated terms)

4.1 Symbols specified in ISO/IEC 10118-1

This part of ISO/IEC 10118 makes use of the following symbols and notations defined in ISO/IEC 10118-1.

B_i A byte.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

D Data.

H Hash-code.

IV Initializing value.

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 the IV.

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

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

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

4.2 Symbols specific to this part

For the purpose of this part of ISO/IEC 10118, the following symbols and notations apply:

a_i, a'_i Sequences of indices used in specifying a round-function.

A^i A sequence of constant matrices used in specifying the round-function defined in Clause 13.

c_0 Function taking a string of 64 elements of GF(2^8) as input, and giving an 8 by 8 matrix with entries from GF(2^8) as output, used in specifying the round-function defined in Clause 13.

c_1, c_2, c_3 Functions taking an 8 by 8 matrix of elements of GF(2^8) as input, and giving an 8 by 8 matrix with entries from GF(2^8) as output, used in specifying the round-function defined in Clause 13.

c_4 Function taking two 8 by 8 matrices of elements of GF(2^8) as input, and giving an 8 by 8 matrix with entries from GF(2^8) as output, used in specifying the round-function defined in Clause 13.

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

C'' An 8 by 8 circulant matrix with entries chosen from GF(2^8) used in specifying the round-function in Clause 13.

D_i A block derived from the data-string after the padding process.

d_i, e_i, f_i, g_i Functions taking either one or three words as input and producing a single word as output, used in specifying round-functions.

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

GF(2^8) A field defined as GF(2)[x] / $p_8(x)$ where $p_8(x) = x^8 + x^4 + x^3 + x^2 + 1$. The elements of the field are 8-bit strings.

M An 8 by 8 matrix whose entries are chosen from GF(2^8).

q The number of blocks in the data string after the padding and splitting processes.

$R^n()$ The operation of right shift by n bits, i.e. if A is a word and n is a non-negative integer then $R^n(A)$ denotes the word obtained by right-shifting the contents of A by n places.

s A nonlinear substitution box, which replaces an element $x \in$ GF(2^8) with another element $s[x] \in$ GF(2^8);

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

$S^n()$ The operation of 'circular left shift' by n bit positions, i.e. if A is a word and n is a non-negative integer then $S^n(A)$ denotes the word obtained by left-shifting the contents of A by n places in a cyclic fashion.

$S'^n()$ The operation of 'circular right shift' by n bit positions, i.e. if A is a word and n is a non-negative integer then $S'^n(A)$ denotes the word obtained by right-shifting the contents of A by n places in a cyclic fashion.

t_i, t'_i Shift-values used in specifying a round-function.

W, X_i, X'_i, Y_i, Z_i Words used to store the results of intermediate computations.

W', X'', K_i, Y', Z' Matrices with entries chosen from GF(2^8) used to store the results of intermediate computations.

Λ The bit-wise logical AND operation on bit-strings, i.e. if A, B are words then $A \Lambda B$ is the word equal to the bit-wise logical AND of A and B.

V The bit-wise logical OR operation on bit-strings, i.e. if A, B are words then $A \mathrm{V} B$ is the word equal to the bit-wise logical OR of A and B.

¬ The bit-wise logical NOT operation on a bit-string, i.e., if A is a word then $\neg A$ is the word equal to the bit-wise logical NOT of A.

$\boxplus$ 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 $A \boxplus B$ is the word obtained by treating A and B as the binary representations of integers and computing their sum modulo 2^w, where the result is constrained to lie between 0 and 2^w-1 inclusive. The value of w is 32 for dedicated hash-functions 1-4, defined in Clauses 7-10, and 64 for dedicated hash-functions 5 and 6, defined in Clauses 11 and 12.

• The multiplication operation of 8 by 8 matrices with entries chosen from GF(2^8). I.e. if A and B are such matrices, then $A \bullet B$ is the matrix obtained by multiplying A and B in the following way: treat each entry of either A or B as the binary polynomial representation of an integer (for example, the binary polynomial representation of integer 89 (hexadecimal) is x^7+x^3+1); treat a multiplication of two of the entries as the remainder when a multiplication of the two polynomials is divided by a polynomial $p_8(x)$, where $p_8(x) = x^8 + x^4 + x^3 + x^2 + 1$; and treat a sum operation as the operation $\oplus$.

:= A symbol denoting the 'set equal to' operation used in procedural specifications of round-functions, where it indicates that the word (or the matrix in Clause 13) on the left side of the symbol shall be made equal to the value of the expression on the right side of the symbol.

5 Requirements

Users who wish to employ a hash-function from this part of ISO/IEC 10118 shall select:

— one of the dedicated hash-functions specified below; and
— the length L_H of the hash-code H.

NOTE The first and second dedicated hash-functions are defined so as to facilitate software implementations for 'little-endian' computers, i.e., where the lowest-addressed byte in a word is interpreted as the least significant; conversely, the third, fourth, fifth and sixth dedicated hash-functions are defined so as to facilitate software implementations for 'big-endian' computers, i.e., where the lowest-addressed byte in a word is interpreted as the most significant. However, by adjusting the definition appropriately, any of these six round-functions can be implemented on a 'big-endian' or a 'little-endian' computer. The seventh dedicated hash-function is defined to be 'endian-neutral', in the sense that it uses no endian-sensitive arithmetical operation (such as integer addition). If sequences of elements from GF(2^8) (i.e., bytes) are mapped to computer words to parallelize such operations as exclusive-or, the byte disposition within a word is irrelevant, as long as the inverse mapping is consistent. All the hash-functions defined in this part of ISO/IEC 10118 take a bit-string as input and give a bit-string as output; this is independent of the internal byte-ordering convention used within each hash-function.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

NOTE The choice of L_H affects the security of the hash-function. All of the hash-functions specified in this part of ISO/IEC 10118 are believed to be collision-resistant hash-functions in environments where performing $2^{L_H/2}$ hash-code computations is deemed to be computationally infeasible.

6 Model for dedicated hash-functions

The hash-functions specified in this part of ISO/IEC 10118 are based on the general model for hash-functions given in part 1 of this standard, i.e., ISO/IEC 10118-1:2000.

In the specifications of the hash-functions in this part of ISO/IEC 10118, it is assumed that the padded data-string input to the hash-function is in the form of a sequence of bytes. If the padded data-string is in the form of a sequence of $8n$ bits, $x_0, x_1, \ldots, x_{8n-1}$, then it shall be interpreted as a sequence of n bytes, $B_0, B_1, \ldots, B_{n-1}$, in the following way. Each group of eight consecutive bits is considered as a byte, the first bit of a group being the most significant bit of that byte. Hence

$$B_i = 2^7 x_{8i} + 2^6 x_{8i+1} + \cdots + x_{8i+7}$$

for every i ($0 \le i < n$).

The output transformation for the hash-functions specified in this part of ISO/IEC 10118 is that the hash-code H is derived by taking the leftmost L_H bits of the final L_2-bit output string H_q.

Identifiers are defined for each of the seven dedicated hash-functions specified in this standard. The hash-function identifiers for the dedicated hash-functions specified in Clauses 7, 8, 9, 10, 11, 12 and 13 are equal to 31, 32, 33, 34, 35, 36 and 37 (hexadecimal) respectively. The range of values from 38 to 3F (hexadecimal) are reserved for future use as hash-function identifiers by this part of ISO/IEC 10118. The hash-function identifiers are also used in the OSI object identifiers assigned in Annex C.

7 Dedicated Hash-Function 1 (RIPEMD-160)

In this Clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 1. This dedicated hash-function can be applied to all data strings D containing at most 2^{64}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 1 is equal to 31 (hexadecimal).

NOTE Dedicated Hash-Function 1 defined in this clause is commonly called RIPEMD-160, [3].

7.1 Parameters, functions and constants

7.1.1 Parameters

For this hash-function $L_1 = 512$, $L_2 = 160$ and L_H is up to 160.

7.1.2 Byte ordering convention

In the specification of the round-function of this clause it is assumed that the block input to the round-function is in the form of a sequence of 32-bit words, each 512-bit block being made up of 16 such words. A sequence of 64 bytes, $B_0, B_1, \ldots, B_{63}$, shall be interpreted as a sequence of 16 words, $Z_0, Z_1, \ldots, Z_{15}$, in the following way. Each group of four consecutive bytes is considered as a word, the first byte of a word being the least significant byte of that word. Hence

$$Z_i = 2^{24} B_{4i+3} + 2^{16} B_{4i+2} + 2^8 B_{4i+1} + B_{4i}, \qquad (0 \le i \le 15).$$

To convert the hash-code from a sequence of words to a byte-sequence, the inverse process shall be followed.

NOTE The byte-ordering specified here is different from that of subclause 9.1.2.

© ISO 2012
Documents
Document Sharing Hub
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

7.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on 32-bit words. A sequence of functions $g_0, g_1, \ldots, g_{79}$ is used in this round-function, where each function g_i, $0 \le i \le 79$, takes three words X_0, X_1 and X_2 as input and produces a single word as output.

The functions g_i are defined as follows:

$$g_i(X_0,X_1,X_2) = X_0 \oplus X_1 \oplus X_2, \quad (0 \le i \le 15),$$
$$g_i(X_0,X_1,X_2) = (X_0 \wedge X_1) \vee (\neg X_0 \wedge X_2), \quad (16 \le i \le 31),$$
$$g_i(X_0,X_1,X_2) = (X_0 \vee \neg X_1) \oplus X_2, \quad (32 \le i \le 47),$$
$$g_i(X_0,X_1,X_2) = (X_0 \wedge X_2) \vee (X_1 \wedge \neg X_2), \quad (48 \le i \le 63),$$
$$g_i(X_0,X_1,X_2) = X_0 \oplus (X_1 \vee \neg X_2), \quad (64 \le i \le 79).$$

7.1.4 Constants

Two sequences of constant words $C_0, C_1, \ldots, C_{79}$ and $C'_0, C'_1, \ldots, C'_{79}$ are used in this round-function. In a hexadecimal representation (where the most significant bit corresponds to the left-most bit) these are defined as follows:

C_i = 00000000, ($0 \le i \le 15$),
C_i = 5A827999, ($16 \le i \le 31$),
C_i = 6ED9EBA1, ($32 \le i \le 47$),
C_i = 8F1BBCDC, ($48 \le i \le 63$),
C_i = A953FD4E, ($64 \le i \le 79$),

C'_i = 50A28BE6, ($0 \le i \le 15$),
C'_i = 5C4DD124, ($16 \le i \le 31$),
C'_i = 6D703EF3, ($32 \le i \le 47$),
C'_i = 7A6D76E9, ($48 \le i \le 63$),
C'_i = 00000000, ($64 \le i \le 79$).

Two sequences of 80 shift-values are used in this round-function, where each shift-value is between 5 and 15. We denote these sequences by $(t_0, t_1, \ldots, t_{79})$ and $(t'_0, t'_1, \ldots, t'_{79})$. A further two sequences of 80 indices are used in this round-function, where each value in the sequence is between 0 and 15. We denote these sequences as $(a_0, a_1, \ldots, a_{79})$, and $(a'_0, a'_1, \ldots, a'_{79})$. All four sequences are defined in Table 1 below.

Table 1

i	0	1	2	3	4	5	6	7
t_i	11	14	15	12	5	8	7	9
t'_i	8	9	9	11	13	15	15	5
a_i	0	1	2	3	4	5	6	7
a'_i	5	14	7	0	9	2	11	4

i	8	9	10	11	12	13	14	15
t_i	11	13	14	15	6	7	9	8
t'_i	7	7	8	11	14	14	12	6
a_i	8	9	10	11	12	13	14	15
a'_i	13	6	15	8	1	10	3	12

i	16	17	18	19	20	21	22	23
t_i	7	6	8	13	11	9	7	15
t'_i	9	13	15	7	12	8	9	11
a_i	7	4	13	1	10	6	15	3
a'_i	6	11	3	7	0	13	5	10

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

i	24	25	26	27	28	29	30	31
t_i	7	12	15	9	11	7	13	12
t'_i	7	7	12	7	6	15	13	11
a_i	12	0	9	5	2	14	11	8
a'_i	14	15	8	12	4	9	1	2

i	32	33	34	35	36	37	38	39
t_i	11	13	6	7	14	9	13	15
t'_i	9	7	15	11	8	6	6	14
a_i	3	10	14	4	9	15	8	1
a'_i	15	5	1	3	7	14	6	9

i	40	41	42	43	44	45	46	47
t_i	14	8	13	6	5	12	7	5
t'_i	12	13	5	14	13	13	7	5
a_i	2	7	0	6	13	11	5	12
a'_i	11	8	12	2	10	0	4	13

i	48	49	50	51	52	53	54	55
t_i	11	12	14	15	14	15	9	8
t'_i	15	5	8	11	14	14	6	14
a_i	1	9	11	10	0	8	12	4
a'_i	8	6	4	1	3	11	15	0

i	56	57	58	59	60	61	62	63
t_i	9	14	5	6	8	6	5	12
t'_i	6	9	12	9	12	5	15	8
a_i	13	3	7	15	14	5	6	2
a'_i	5	12	2	13	9	7	10	14

i	64	65	66	67	68	69	70	71
t_i	9	15	5	11	6	8	13	12
t'_i	8	5	12	9	12	5	14	6
a_i	4	0	5	9	7	12	2	10
a'_i	12	15	10	4	1	5	8	7

i	72	73	74	75	76	77	78	79
t_i	5	12	13	14	11	8	5	6
t'_i	8	13	6	5	15	13	11	11
a_i	14	1	3	8	11	6	15	13
a'_i	6	2	13	14	0	3	9	11

7.1.5 Initializing value

For this round-function the initializing value, *IV*, shall always be the following 160-bit string, represented here as a sequence of five words Y_0, Y_1, Y_2, Y_3, Y_4 in a hexadecimal representation, where Y_0 represents the left-most 32 of the 160 bits:

Y_0 = 67452301,
Y_1 = EFCDAB89,
Y_2 = 98BADCFE,
Y_3 = 10325476,
Y_4 = C3D2E1F0.

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

7.2 Padding method

The data string D needs to be padded to make it contain a number of bits which is an integer multiple of 512. The padding procedure operates as follows:

1. D is concatenated with a single '1' bit.

2. The result of the previous step is concatenated with between zero and 511 '0' bits such that the length (in bits) of the resultant string is congruent to 448 modulo 512. More explicitly, if the original length of D is L_D, and letting r be the remainder when L_D is divided by 512, then the number of concatenated zeros is equal to either 447-r (if $r \leq 447$) or 959-r (if $r > 447$). The result will be a bit string whose length will be 64 bits short of an integer multiple of 512 bits.

3. Divide the 64-bit binary representation of L_D into two 32-bit strings, one representing the 'most significant half' of L_D and the other the 'least significant half'. Now concatenate the string resulting from the previous step with these two 32-bit strings, with the 'least significant half' preceding the 'most significant half'.

In the description of the round-function which follows, each 512-bit data block D_i, $1 \leq i \leq q$, is treated as a sequence of 16 words, $Z_0, Z_1, \ldots, Z_{15}$, where Z_0 corresponds to the left-most 32 bits of D_i.

NOTE The concatenation of the two 32-bit strings of L_D in step 3 is such that these two 32-bit strings are used directly as the words Z_{14} and Z_{15} of the last data block; based on the byte ordering convention in Clause 7.1.2, the least significant octet of L_D is the leftmost octet, and the most significant octet of L_D is the rightmost octet.

7.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W, X_0, X_1, X_2, X_3, X_4, X'_0, X'_1, X'_2, X'_3, X'_4 to denote eleven distinct words which contain values required in the computations.

1. Suppose the 512-bit (first) input to Φ is contained in $Z_0, Z_1, \ldots, Z_{15}$, where Z_0 contains the left-most 32 of the 512 bits. Suppose also that the 160-bit (second) input to Φ is contained in five words, Y_0, Y_1, Y_2, Y_3, Y_4.

2. Let $X_0 := Y_0$, $X_1 := Y_1$, $X_2 := Y_2$, $X_3 := Y_3$ and $X_4 := Y_4$.

3. Let $X'_0 := Y_0$, $X'_1 := Y_1$, $X'_2 := Y_2$, $X'_3 := Y_3$ and $X'_4 := Y_4$.

4. For i := 0 to 79 do the following four steps in the order specified:

   (a) $W := S^{t_i}(X_0 \uplus g_i(X_1, X_2, X_3) \uplus Z_{a_i} \uplus C_i) \uplus X_4$;

   (b) $X_0 := X_4$; $X_4 := X_3$; $X_3 := S^{10}(X_2)$; $X_2 := X_1$; $X_1 := W$;

   (c) $W := S^{t'_i}(X'_0 \uplus g_{79-i}(X'_1, X'_2, X'_3) \uplus Z_{a'_i} \uplus C'_i) \uplus X'_4$;

   (d) $X'_0 := X'_4$; $X'_4 := X'_3$; $X'_3 := S^{10}(X'_2)$; $X'_2 := X'_1$; $X'_1 := W$.

5. Let

$$W := Y_0,$$
$$Y_0 := Y_1 \uplus X_2 \uplus X'_3,$$
$$Y_1 := Y_2 \uplus X_3 \uplus X'_4,$$
$$Y_2 := Y_3 \uplus X_4 \uplus X'_0,$$
$$Y_3 := Y_4 \uplus X_0 \uplus X'_1,$$
$$Y_4 := W \uplus X_1 \uplus X'_2.$$

6. The five words Y_0, Y_1, Y_2, Y_3, Y_4 represent the output of the round-function Φ. After the final iteration of the round-function, the five words Y_0, Y_1, Y_2, Y_3, Y_4 shall be converted to a sequence of 20 bytes using the inverse of the procedure specified in 7.1.2, and where Y_0 shall yield the first four bytes, Y_1 the next four bytes, and so on. Thus the first (left-most) byte will correspond to the least significant byte of Y_0, and the

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

20th (right-most) byte will correspond to the most significant byte of Y_4. The 20 bytes shall be converted to a string of 160 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 160th (right-most) bit will correspond to the least significant bit of the 20th (right-most) byte.

Figure 1 below shows steps a and b of item 4 of the round function Φ in Dedicated Hash-Function 1 (RIPEMD-160) (the other half, i.e., steps c and d is similar). In the round function Φ, steps a to c of item 4 are used for 80 times (i = 0, …, 79).

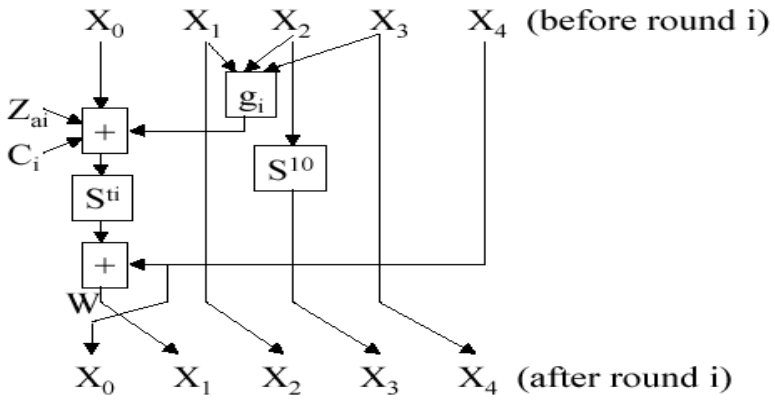

Figure 1 — Part of the round function in Dedicated Hash-Function 1

8 Dedicated Hash-Function 2 (RIPEMD-128)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 2. This dedicated hash-function can be applied to all data strings D containing at most 2^{64}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 2 is equal to 32 (hexadecimal).

NOTE Dedicated Hash-Function 2 defined in this clause is commonly called RIPEMD-128, [3]. This hash-function should only be used in applications where a hash-code containing 128 bits or less is considered adequately secure.

8.1 Parameters, functions and constants

8.1.1 Parameters

For this hash-function L_1=512, L_2=128 and L_H is up to 128.

8.1.2 Byte ordering convention

The byte ordering convention for this hash-function is the same as that for the hash-function of clause 7.

8.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on 32-bit words. A sequence of functions g_0, g_1, …, g_{63} is used in this round-function, where each function g_i, $0 \le i \le 63$, takes three words X_0, X_1 and X_2 as input and produces a single word as output.

The functions g_i are defined to be the same as the first 64 of the functions defined in subclause 7.1.3.

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

8.1.4 Constants

Two sequences of constant words C_0, C_1, …, C_{63} and C'_0, C'_1, …, C'_{63} are used in this round-function. In a hexadecimal representation (where the most significant bit corresponds to the left-most bit) these are defined as follows:

C_i = 00000000, ($0 \le i \le 15$),
C_i = 5A827999, ($16 \le i \le 31$),
C_i = 6ED9EBA1, ($32 \le i \le 47$),
C_i = 8F1BBCDC, ($48 \le i \le 63$),

C'_i = 50A28BE6, ($0 \le i \le 15$),
C'_i = 5C4DD124, ($16 \le i \le 31$),
C'_i = 6D703EF3, ($32 \le i \le 47$),
C'_i = 00000000, ($48 \le i \le 63$).

Two sequences of 64 shift-values are also used in this round-function, where each shift-value is between 5 and 15. We denote these sequences by (t_0, t_1, …, t_{63}) and (t'_0,t'_1, …, t'_{63}), and they are defined to be equal to the first 64 values of the corresponding sequences defined in subclause 7.1.4.

Finally, two further sequences of 64 indices are used in this round-function, where each value in the sequence is between 0 and 15. We denote these sequences by (a_0, a_1, …, a_{63}), and (a'_0, a'_1, …, a'_{63}), and they are defined to be equal to the first 64 values of the corresponding sequences defined in subclause 7.1.4.

8.1.5 Initializing value

For this hash-function the initializing value, *IV*, shall always be the following 128-bit string, represented here as a sequence of four words Y_0, Y_1, Y_2, Y_3 in a hexadecimal representation, where Y_0 represents the left-most 32 of the 128 bits:

Y_0 = 67452301,
Y_1 = EFCDAB89,
Y_2 = 98BADCFE,
Y_3 = 10325476.

8.2 Padding method

The padding method to be used with this hash-function shall be the same as the padding method defined in subclause 7.2.

8.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W, X_0, X_1, X_2, X_3, X'_0, X'_1, X'_2, X'_3 to denote nine distinct words which contain values required in the computations.

1. Suppose the 512-bit (first) input to Φ is contained in Z_0, Z_1, …, Z_{15}, where Z_0 contains the left-most 32 of the 512 bits. Suppose also that the 128-bit (second) input to Φ is contained in four words, Y_0, Y_1, Y_2, Y_3.

2. Let $X_0 := Y_0$, $X_1 := Y_1$, $X_2 := Y_2$ and $X_3 := Y_3$.

3. Let $X'_0 := Y_0$, $X'_1 := Y_1$, $X'_2 := Y_2$ and $X'_3 := Y_3$.

4. For $i := 0$ to 63 do the following four steps in the order specified:

   (a) $W := S^{t_i}(X_0 \boxplus g_i(X_1, X_2, X_3) \boxplus Z_{a_i} \boxplus C_i)$;

   (b) $X_0 := X_3$; $X_3 := X_2$; $X_2 := X_1$; $X_1 := W$;

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

(c) $W := S^{t'_i}(X'_0 \uplus g_{63-i}(X'_1, X'_2, X'_3) \uplus Z_{a'_i} \uplus C'_i)$;

(d) $X'_0 := X'_3$; $X'_3 := X'_2$; $X'_2 := X'_1$; $X'_1 := W$.

5. Let

$W := Y_0$,
$Y_0 := Y_1 \uplus X_2 \uplus X'_3$,
$Y_1 := Y_2 \uplus X_3 \uplus X'_0$,
$Y_2 := Y_3 \uplus X_0 \uplus X'_1$,
$Y_3 := W \uplus X_1 \uplus X'_2$.

6. The four words Y_0, Y_1, Y_2, Y_3 represent the output of the round-function Φ. After the final iteration of the round-function, the four words Y_0, Y_1, Y_2, Y_3 shall be converted to a sequence of 16 bytes using the inverse of the procedure specified in 7.1.2, and where Y_0 shall yield the first four bytes, Y_1 the next four bytes, and so on. Thus the first (left-most) byte will correspond to the least significant byte of Y_0, and the 16th (right-most) byte will correspond to the most significant byte of Y_3. The 16 bytes shall be converted to a string of 128 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 128th (right-most) bit will correspond to the least significant bit of the 16th (right-most) byte.

Figure 2 below shows steps a and b of item 4 of the round function Φ in Dedicated Hash-Function 2 (RIPEMD-128) (the other half, i.e., steps c and d is similar). In the round function Φ, steps a to c of item 4 are used for 64 times (i = 0, …, 63).

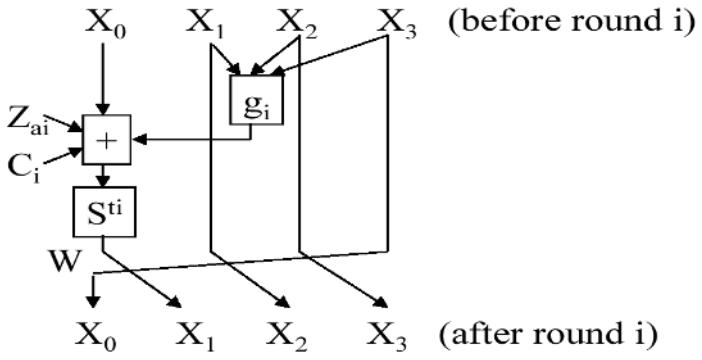

Figure 2 — Part of the round function in Dedicated Hash-Function 2

9 Dedicated Hash-Function 3 (SHA-1)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 3. This dedicated hash-function can be applied to all data strings D containing at most 2^{64}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 3 is equal to 33 (hexadecimal).

NOTE Dedicated Hash-Function 3 defined in this clause is commonly called SHA-1, [2].

9.1 Parameters, functions and constants

9.1.1 Parameters

For this hash-function L_1 = 512, L_2 = 160 and L_H is up to 160.

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

9.1.2 Byte ordering convention

In the specification of the round-function of this clause it is assumed that the block input to the round-function is in the form of a sequence of 32-bit words, each 512-bit block being made up of 16 such words. A sequence of 64 bytes, B_0, B_1, …, B_{63}, shall be interpreted as a sequence of 16 words, Z_0, Z_1, …, Z_{15}, in the following way. Each group of four consecutive bytes is considered as a word, the first byte of a word being the most significant byte of that word. Hence

$$Z_i = 2^{24}B_{4i} + 2^{16}B_{4i+1} + 2^{8}B_{4i+2} + B_{4i+3}, \qquad (0 \le i \le 15).$$

To convert the hash-code from a sequence of words to a sequence of bytes, the inverse process shall be followed.

NOTE The byte-ordering specified here is different from that of subclause 7.1.2.

9.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on 32-bit words. A sequence of functions f_0, f_1, …, f_{79} is used in this round-function, where each function f_i, $0 \le i \le 79$, takes three words X_0, X_1 and X_2 as input and produces a single word as output.

The functions f_i are defined as follows:

$$\begin{aligned}
f_i(X_0, X_1, X_2) &= (X_0 \wedge X_1) \vee (\neg X_0 \wedge X_2), && (0 \le i \le 19),\\
f_i(X_0, X_1, X_2) &= X_0 \oplus X_1 \oplus X_2, && (20 \le i \le 39),\\
f_i(X_0, X_1, X_2) &= (X_0 \wedge X_1) \vee (X_0 \wedge X_2) \vee (X_1 \wedge X_2), && (40 \le i \le 59),\\
f_i(X_0, X_1, X_2) &= X_0 \oplus X_1 \oplus X_2, && (60 \le i \le 79).
\end{aligned}$$

9.1.4 Constants

A sequence of constant words C_0, C_1, …, C_{79} is used in this round-function. In a hexadecimal representation (where the most significant bit corresponds to the left-most bit) these are defined as follows:

$$\begin{aligned}
C_i &= \text{5A827999}, && (0 \le i \le 19),\\
C_i &= \text{6ED9EBA1}, && (20 \le i \le 39),\\
C_i &= \text{8F1BBCDC}, && (40 \le i \le 59),\\
C_i &= \text{CA62C1D6}, && (60 \le i \le 79).
\end{aligned}$$

9.1.5 Initializing value

For this round-function the initializing value, *IV*, shall always be the following 160-bit string, represented here as a sequence of five words Y_0, Y_1, Y_2, Y_3, Y_4 in a hexadecimal representation, where Y_0 represents the left-most 32 of the 160 bits:

$$\begin{aligned}
Y_0 &= \text{67452301},\\
Y_1 &= \text{EFCDAB89},\\
Y_2 &= \text{98BADCFE},\\
Y_3 &= \text{10325476},\\
Y_4 &= \text{C3D2E1F0}.
\end{aligned}$$

9.2 Padding method

The data string *D* needs to be padded to make it contain a number of bits which is an integer multiple of 512. The padding procedure operates as follows:

1. *D* is concatenated with a single ‘1’ bit.

2. The result of the previous step is concatenated with between zero and 511 ‘0’ bits such that the length (in bits) of the resultant string is congruent to 448 modulo 512. More explicitly, if the original length of *D* is L_D,

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

and letting r be the remainder when L_D is divided by 512, then the number of concatenated zeros is equal to either 447-r (if $r \leq 447$) or 959-r (if $r > 447$). The result will be a bit string whose length will be 64 bits short of an integer multiple of 512 bits.

3. Concatenate the string resulting from the previous step with the 64-bit binary representation of L_D, most significant bit first.

In the description of the round-function which follows, each 512-bit data block D_i, $1 \leq i \leq q$, is treated as a sequence of 16 words, Z_0, Z_1, …, Z_{15}, where Z_0 corresponds to the left-most 32 bits of D_i.

NOTE The concatenation of the 64-bit string of L_D in step 3 is such that the most significant 32-bit string and the least significant 32-bit string of L_D are used respectively as the words Z_{14} and Z_{15} of the last data block; based on the byte ordering convention in Clause 9.1.2, the most significant byte of L_D is the leftmost byte and the least significant byte of L_D is the rightmost byte.

9.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W, X_0, X_1, X_2, X_3, X_4, Z_0, Z_1, …, Z_{79} to denote 86 distinct words which contain values required in the computations.

1. Suppose the 512-bit (first) input to Φ is contained in Z_0, Z_1, …, Z_{15}, where Z_0 contains the left-most 32 of the 512 bits. Suppose also that the 160-bit (second) input to Φ is contained in five words, Y_0, Y_1, Y_2, Y_3, Y_4.

2. For i = 16 to 79 let

   $Z_i := S^1(Z_{i-3} \oplus Z_{i-8} \oplus Z_{i-14} \oplus Z_{i-16})$.

3. Let $X_0 := Y_0$, $X_1 := Y_1$, $X_2 := Y_2$, $X_3 := Y_3$ and $X_4 := Y_4$.

4. For i = 0 to 79 do the following two steps

   (a) $W := S^5(X_0) \uplus f_i(X_1, X_2, X_3) \uplus X_4 \uplus Z_i \uplus C_i$;

   (b) $X_4 := X_3$; $X_3 := X_2$; $X_2 := S^{30}(X_1)$; $X_1 := X_0$; $X_0 := W$.

5. Let $Y_0 := Y_0 \uplus X_0$, $Y_1 := Y_1 \uplus X_1$, $Y_2 := Y_2 \uplus X_2$, $Y_3 := Y_3 \uplus X_3$ and $Y_4 := Y_4 \uplus X_4$.

6. The five words Y_0, Y_1, Y_2, Y_3, Y_4 represent the output of the round-function Φ. After the final iteration of the round-function, the five words Y_0, Y_1, Y_2, Y_3, Y_4 shall be converted to a sequence of 20 bytes using the inverse of the procedure specified in 9.1.2, and where Y_0 shall yield the first four bytes, Y_1 the next four bytes, and so on. Thus the first (left-most) byte will correspond to the most significant byte of Y_0, and the 20th (right-most) byte will correspond to the least significant byte of Y_4. The 20 bytes shall be converted to a string of 160 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 160th (right-most) bit will correspond to the least significant bit of the 20th (right-most) byte.

Figure 3 below shows steps a and b of item 4 of the round function Φ in Dedicated Hash-Function 3 (SHA-1). In the round function Φ, steps a and b of item 4 are used for 80 times (i = 0, …, 79).

Documents
Document Sharing Hub
© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

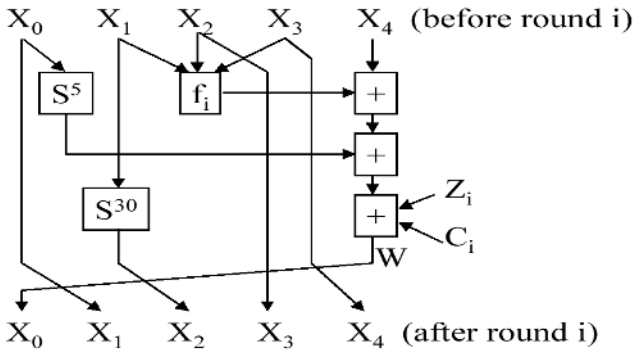

Figure 3 — Part of the round function in Dedicated Hash-Function 3

10 Dedicated Hash-Function 4 (SHA-256)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 4. This dedicated hash-function can be applied to all data strings D containing at most 2^{64}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 4 is equal to 34 (hexadecimal).

NOTE Dedicated Hash-Function 4 defined in this clause is commonly called SHA-256, [2].

10.1 Parameters, functions and constants

10.1.1 Parameters

For this hash-function $L_1 = 512$, $L_2 = 256$ and L_H is up to 256.

10.1.2 Byte ordering convention

The byte ordering convention to be used with this hash-function shall be the same as the byte ordering convention defined in subclause 9.1.2.

10.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on 32-bit words. A sequence of functions e_0, e_1, e_2, e_3, e_4, e_5 is used in this round-function, where e_0 and e_1 each takes three words X_0, X_1 and X_2 as input, e_2, e_3, e_4 and e_5 each takes one word X_0 as input, and each of these six functions produces a single 32-bit word as output.

The functions e_0, e_1, e_2, e_3, e_4 and e_5 are defined as follows:

$$e_0(X_0, X_1, X_2) = (X_0 \wedge X_1) \oplus (\neg X_0 \wedge X_2),$$
$$e_1(X_0, X_1, X_2) = (X_0 \wedge X_1) \oplus (X_0 \wedge X_2) \oplus (X_1 \wedge X_2),$$
$$e_2(X_0) = S'^{2}(X_0) \oplus S'^{13}(X_0) \oplus S'^{22}(X_0),$$
$$e_3(X_0) = S'^{6}(X_0) \oplus S'^{11}(X_0) \oplus S'^{25}(X_0),$$
$$e_4(X_0) = S'^{7}(X_0) \oplus S'^{18}(X_0) \oplus R^{3}(X_0),$$
$$e_5(X_0) = S'^{17}(X_0) \oplus S'^{19}(X_0) \oplus R^{10}(X_0).$$

© ISO 2012

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

10.1.4 Constants

A sequence of constant words C_0, C_1, …, C_{63} is used in this round-function. In a hexadecimal representation (where the most significant bit corresponds to the left-most bit) these are defined as follows, where the words are listed in the order C_0, C_1, …, C_{63}.

```
428a2f98 71374491 b5c0fbcf e9b5dba5 3956c25b 59f111f1 923f82a4 ab1c5ed5
d807aa98 12835b01 243185be 550c7dc3 72be5d74 80deb1fe 9bdc06a7 c19bf174
e49b69c1 efbe4786 0fc19dc6 240ca1cc 2de92c6f 4a7484aa 5cb0a9dc 76f988da
983e5152 a831c66d b00327c8 bf597fc7 c6e00bf3 d5a79147 06ca6351 14292967
27b70a85 2e1b2138 4d2c6dfc 53380d13 650a7354 766a0abb 81c2c92e 92722c85
a2bfe8a1 a81a664b c24b8b70 c76c51a3 d192e819 d6990624 f40e3585 106aa070
19a4c116 1e376c08 2748774c 34b0bcb5 391c0cb3 4ed8aa4a 5b9cca4f 682e6ff3
748f82ee 78a5636f 84c87814 8cc70208 90befffa a4506ceb bef9a3f7 c67178f2
```

NOTE These values are the first thirty-two bits of the fractional parts of the cube roots of the first sixty-four primes.

10.1.5 Initializing value

For this round-function the initializing value, *IV*, shall always be the following 256-bit string, represented here as a sequence of eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 in a hexadecimal representation, where Y_0 represents the left-most 32 of the 256 bits:

Y_0 = 6a09e667,
Y_1 = bb67ae85,
Y_2 = 3c6ef372,
Y_3 = a54ff53a,
Y_4 = 510e527f,
Y_5 = 9b05688c,
Y_6 = 1f83d9ab,
Y_7 = 5be0cd19.

NOTE These values are obtained by taking the fractional parts of the square roots of the first eight primes.

10.2 Padding method

The padding method to be used with this hash-function shall be the same as the padding method defined in subclause 9.2.

10.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W_1, W_2, X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, Z_0, Z_1, …, Z_{63} to denote 74 distinct words which contain values required in the computations.

1. Suppose the 512-bit (first) input to Φ is contained in Z_0, Z_1, …, Z_{15}, where Z_0 contains the left-most 32 of the 512 bits. Suppose also that the 256-bit (second) input to Φ is contained in eight words, Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7.

2. For i =16 to *63* let

   $Z_i := e_5(Z_{i-2}) \boxplus Z_{i-7} \boxplus e_4(Z_{i-15}) \boxplus Z_{i-16}$.

3. Let $X_0 := Y_0$, $X_1 := Y_1$, $X_2 := Y_2$, $X_3 := Y_3$, $X_4 := Y_4$, $X_5 := Y_5$, $X_6 := Y_6$ and $X_7 := Y_7$.

4. For i = 0 to 63 do the following three steps

   (a) $W_1 := X_7 \boxplus e_3(X_4) \boxplus e_0(X_4, X_5, X_6) \boxplus C_i \boxplus Z_i$;

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

(b) $W_2 := e_2(X_0) \uplus e_1(X_0, X_1, X_2)$;

(c) $X_7 := X_6$; $X_6 := X_5$; $X_5 := X_4$; $X_4 := X_3 \uplus W_1$; $X_3 := X_2$; $X_2 := X_1$; $X_1 := X_0$; $X_0 := W_1 \uplus W_2$.

5. Let $Y_0 := Y_0 \uplus X_0$, $Y_1 := Y_1 \uplus X_1$, $Y_2 := Y_2 \uplus X_2$, $Y_3 := Y_3 \uplus X_3$, $Y_4 := Y_4 \uplus X_4$, $Y_5 := Y_5 \uplus X_5$, $Y_6 := Y_6 \uplus X_6$ and $Y_7 := Y_7 \uplus X_7$.

6. The eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 represent the output of the round-function Φ. After the final iteration of the round-function, the eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 shall be converted to a sequence of 32 bytes using the inverse of the procedure specified in 10.1.2, and where Y_0 shall yield the first four bytes, Y_1 the next four bytes, and so on. Thus the first (left-most) byte will correspond to the most significant byte of Y_0, and the 32nd (right-most) byte will correspond to the least significant byte of Y_7. The 32 bytes shall be converted to a string of 256 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 256th (right-most) bit will correspond to the least significant bit of the 32nd (right-most) byte.

Figure 4 below shows steps a, b and c of item 4 of the round function Φ in Dedicated Hash-Function 4 (SHA-256). In the round function Φ, steps a and b and c of item 4 are used for 64 times (i = 0, …, 63).

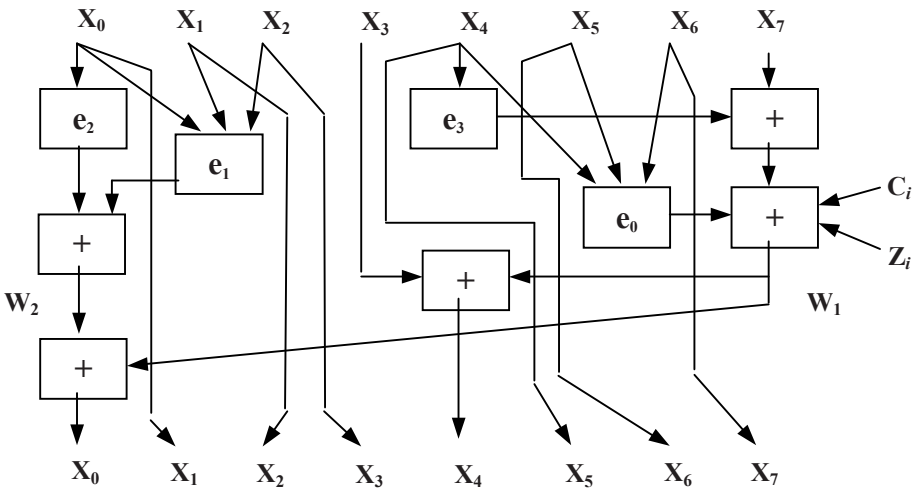

Figure 4 — Part of the round function in Dedicated Hash-Function 4

11 Dedicated Hash-Function 5 (SHA-512)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 5. This dedicated hash-function can be applied to all data strings D containing at most 2^{128}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 5 is equal to 35 (hexadecimal).

NOTE Dedicated Hash-Function 5 defined in this clause is commonly called SHA-512, [2].

11.1 Parameters, functions and constants

11.1.1 Parameters

For this hash-function L_1 = 1024, L_2 = 512 and L_H is up to 512.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

11.1.2 Byte ordering convention

In the specification of the round-function of this clause it is assumed that the block input to the round-function is in the form of a sequence of 64-bit words, each 1024-bit block being made up of 16 such words. A sequence of 128 bytes, B_0, B_1, …, B_{127}, shall be interpreted as a sequence of 16 words, Z_0, Z_1, …, Z_{15}, in the following way. Each group of eight consecutive bytes is considered as a word, the first byte of a word being the most significant byte of that word. Hence

$$Z_i = 2^{56}B_{8i} + 2^{48}B_{8i+1} + 2^{40}B_{8i+2} + 2^{32}B_{8i+3} + 2^{24}B_{8i+4} + 2^{16}B_{8i+5} + 2^{8}B_{8i+6} + B_{8i+7}, \qquad (0 \le i \le 15).$$

To convert the hash-code from a sequence of words to a sequence of bytes, the inverse process shall be followed.

11.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on 64-bit words. A sequence of functions d_0, d_1, d_2, d_3, d_4, d_5 is used in this round-function, where d_0 and d_1 each takes three 64-bit words X_0, X_1 and X_2 as input, d_2, d_3, d_4 and d_5 each takes one 64-bit word X_0 as input, and each of these six functions produces a single 64-bit word as output.

The functions d_0, d_1, d_2, d_3, d_4 and d_5 are defined as follows:

$$d_0(X_0, X_1, X_2) = (X_0 \wedge X_1) \oplus (\neg X_0 \wedge X_2),$$
$$d_1(X_0, X_1, X_2) = (X_0 \wedge X_1) \oplus (X_0 \wedge X_2) \oplus (X_1 \wedge X_2),$$
$$d_2(X_0) = S'^{28}(X_0) \oplus S'^{34}(X_0) \oplus S'^{39}(X_0),$$
$$d_3(X_0) = S'^{14}(X_0) \oplus S'^{18}(X_0) \oplus S'^{41}(X_0),$$
$$d_4(X_0) = S'^{1}(X_0) \oplus S'^{8}(X_0) \oplus R^{7}(X_0),$$
$$d_5(X_0) = S'^{19}(X_0) \oplus S'^{61}(X_0) \oplus R^{6}(X_0).$$

11.1.4 Constants

A sequence of constant words C_0, C_1, …, C_{79} is used in this round-function. In a hexadecimal representation (where the most significant bit corresponds to the left-most bit) these are defined as follows, where the words are listed in the order C_0, C_1, …, C_{79}.

```
428a2f98d728ae22 7137449123ef65cd b5c0fbcfec4d3b2f e9b5dba58189dbbc
3956c25bf348b538 59f111f1b605d019 923f82a4af194f9b ab1c5ed5da6d8118
d807aa98a3030242 12835b0145706fbe 243185be4ee4b28c 550c7dc3d5ffb4e2
72be5d74f27b896f 80deb1fe3b1696b1 9bdc06a725c71235 c19bf174cf692694
e49b69c19ef14ad2 efbe4786384f25e3 0fc19dc68b8cd5b5 240ca1cc77ac9c65
2de92c6f592b0275 4a7484aa6ea6e483 5cb0a9dcbd41fbd4 76f988da831153b5
983e5152ee66dfab a831c66d2db43210 b00327c898fb213f bf597fc7beef0ee4
c6e00bf33da88fc2 d5a79147930aa725 06ca6351e003826f 142929670a0e6e70
27b70a8546d22ffc 2e1b21385c26c926 4d2c6dfc5ac42aed 53380d139d95b3df
650a73548baf63de 766a0abb3c77b2a8 81c2c92e47edaee6 92722c851482353b
a2bfe8a14cf10364 a81a664bbc423001 c24b8b70d0f89791 c76c51a30654be30
d192e819d6ef5218 d69906245565a910 f40e35855771202a 106aa07032bbd1b8
19a4c116b8d2d0c8 1e376c085141ab53 2748774cdf8eeb99 34b0bcb5e19b48a8
391c0cb3c5c95a63 4ed8aa4ae3418acb 5b9cca4f7763e373 682e6ff3d6b2b8a3
748f82ee5defb2fc 78a5636f43172f60 84c87814a1f0ab72 8cc702081a6439ec
90befffa23631e28 a4506cebde82bde9 bef9a3f7b2c67915 c67178f2e372532b
ca273eceea26619c d186b8c721c0c207 eada7dd6cde0eb1e f57d4f7fee6ed178
06f067aa72176fba 0a637dc5a2c898a6 113f9804bef90dae 1b710b35131c471b
28db77f523047d84 32caab7b40c72493 3c9ebe0a15c9bebc 431d67c49c100d4c
4cc5d4becb3e42b6 597f299cfc657e2a 5fcb6fab3ad6faec 6c44198c4a475817
```

NOTE These values are the first sixty-four bits of the fractional parts of the cube roots of the first eighty primes.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

11.1.5 Initializing value

For this round-function the initializing value, *IV*, shall always be the following 512-bit string, represented here as a sequence of eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 in a hexadecimal representation, where Y_0 represents the left-most 64 of the 512 bits:

Y_0 = 6a09e667f3bcc908,
Y_1 = bb67ae8584caa73b,
Y_2 = 3c6ef372fe94f82b,
Y_3 = a54ff53a5f1d36f1,
Y_4 = 510e527fade682d1,
Y_5 = 9b05688c2b3e6c1f,
Y_6 = 1f83d9abfb41bd6b,
Y_7 = 5be0cd19137e2179.

NOTE These values are obtained by taking the fractional parts of the square roots of the first eight primes.

11.2 Padding method

The data string *D* needs to be padded to make it contain a number of bits which is an integer multiple of 1024. The padding procedure operates as follows:

1. *D* is concatenated with a single '1' bit.

2. The result of the previous step is concatenated with between zero and 1023 '0' bits such that the length (in bits) of the resultant string is congruent to 896 modulo 1024. More explicitly, if the original length of *D* is L_D, and letting *r* be the remainder when L_D is divided by 1024, then the number of concatenated zeros is equal to either 895-*r* (if $r \leq 895$) or 1919-*r* (if $r > 895$). The result will be a bit string whose length will be 128 bits short of an integer multiple of 1024 bits.

3. Concatenate the string resulting from the previous step with the 128-bit binary representation of L_D, most significant bit first.

In the description of the round-function which follows, each 1024-bit data block D_i, $1 \leq i \leq q$, is treated as a sequence of 16 words, Z_0, Z_1, …, Z_{15}, where Z_0 corresponds to the left-most 64 bits of D_i.

NOTE The concatenation of the 128-bit string of L_D in step 3 is such that the most significant 64-bit string and the least significant 64-bit string of L_D are used respectively as the words Z_{14} and Z_{15} of the last data block; based on the byte ordering convention in Clause 11.1.2, the most significant byte of L_D is the leftmost byte and the least significant byte of L_D is the rightmost byte.

11.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W_1, W_2, X_0, X_1, X_2, X_3, X_4, X_5, X_6, X_7, Z_0, Z_1, …, Z_{79} to denote 90 distinct words which contain values required in the computations.

1. Suppose the 1024-bit (first) input to Φ is contained in Z_0, Z_1, …, Z_{15}, where Z_0 contains the left-most 64 of the 1024 bits. Suppose also that the 512-bit (second) input to Φ is contained in eight words, Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7.

2. For i = 16 to 79 let

$$Z_i := d_5(Z_{i-2}) \boxplus Z_{i-7} \boxplus d_4(Z_{i-15}) \boxplus Z_{i-16}.$$

3. Let $X_0 := Y_0$, $X_1 := Y_1$, $X_2 := Y_2$, $X_3 := Y_3$, $X_4 := Y_4$, $X_5 := Y_5$, $X_6 := Y_6$ and $X_7 := Y_7$.

4. For i = 0 to 79 do the following three steps

(a) $W_1 := X_7 \boxplus d_3(X_4) \boxplus d_0(X_4, X_5, X_6) \boxplus C_i \boxplus Z_i$;

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

(b) $W_2 := d_2(X_0) \uplus d_1(X_0, X_1, X_2)$;

(c) $X_7 := X_6$; $X_6 := X_5$; $X_5 := X_4$; $X_4 := X_3 \uplus W_1$; $X_3 := X_2$; $X_2 := X_1$; $X_1 := X_0$; $X_0 := W_1 \uplus W_2$.

5. Let $Y_0 := Y_0 \uplus X_0$, $Y_1 := Y_1 \uplus X_1$, $Y_2 := Y_2 \uplus X_2$, $Y_3 := Y_3 \uplus X_3$, $Y_4 := Y_4 \uplus X_4$, $Y_5 := Y_5 \uplus X_5$, $Y_6 := Y_6 \uplus X_6$ and $Y_7 := Y_7 \uplus X_7$.

6. The eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 represent the output of the round-function Φ. After the final iteration of the round-function, the eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 shall be converted to a sequence of 64 bytes using the inverse of the procedure specified in 11.1.2, and where Y_0 shall yield the first eight bytes, Y_1 the next eight bytes, and so on. Thus the first (left-most) byte will correspond to the most significant byte of Y_0, and the 64th (right-most) byte will correspond to the least significant byte of Y_7. The 64 bytes shall be converted to a string of 512 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 512th (right-most) bit will correspond to the least significant bit of the 64th (right-most) byte.

Figure 5 below shows steps a, b and c of item 4 of the round function Φ in Dedicated Hash-Function 5 (SHA-512). In the round function Φ, steps a and b and c of item 4 are used for 80 times (i = 0, …, 79).

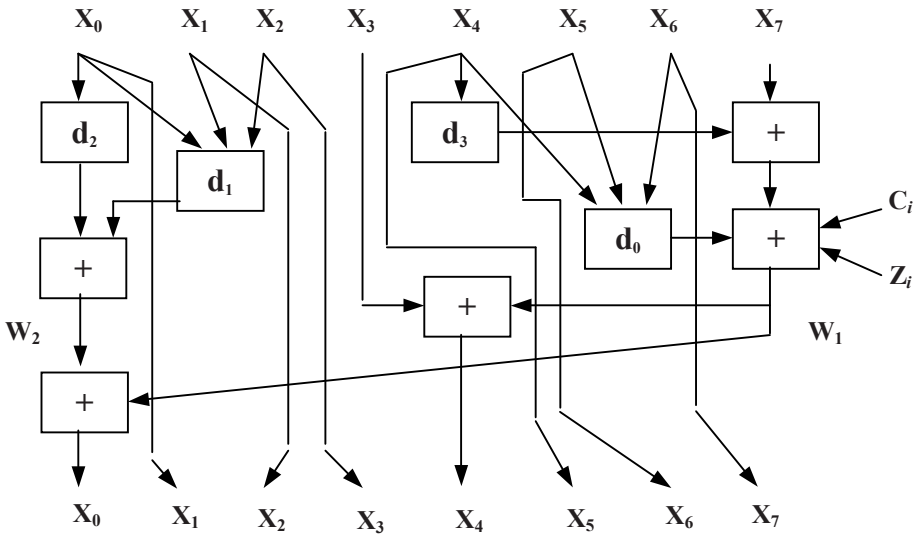

Figure 5 — Part of the round function in Dedicated Hash-Function 5

12 Dedicated Hash-Function 6 (SHA-384)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 6. This dedicated hash-function can be applied to all data strings D containing at most 2^{128}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 6 is equal to 36 (hexadecimal).

NOTE Dedicated Hash-Function 6 defined in this clause is commonly called SHA-384, [2].

12.1 Parameters, functions and constants

12.1.1 Parameters

For this hash-function L_1 = 1024, L_2 = 512 and L_H = 384.

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

12.1.2 Byte ordering convention

The byte ordering convention for this hash-function is the same as that for the hash-function of clause 11.

12.1.3 Functions

The functions for this hash-function are the same as that for the hash-function of clause 11.

12.1.4 Constants

The constants for this hash-function are the same as that for the hash-function of clause 11.

12.1.5 Initializing value

For this round-function the initializing value, *IV*, shall always be the following 512-bit string, represented here as a sequence of eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 in a hexadecimal representation, where Y_0 represents the left-most 64 of the 512 bits:

Y_0 = cbbb9d5dc1059ed8,
Y_1 = 629a292a367cd507,
Y_2 = 9159015a3070dd17,
Y_3 = 152fecd8f70e5939,
Y_4 = 67332667ffc00b31,
Y_5 = 8eb44a8768581511,
Y_6 = db0c2e0d64f98fa7,
Y_7 = 47b5481dbefa4fa4.

NOTE These values are obtained by taking the fractional parts of the square roots of the ninth to the sixteenth primes.

12.2 Padding method

The padding method to be used with this hash-function shall be the same as the padding method defined in subclause 11.2.

12.3 Description of the round-function

The round-function to be used with this hash-function shall be the same as the round-function defined in subclause 11.3.

The final 384-bit hash is obtained by truncating the SHA-512-based hash output to its left-most 384 bits.

13 Dedicated Hash-Function 7 (WHIRLPOOL)

In this clause we specify a padding method, an initializing value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initializing value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 7. This dedicated hash-function can be applied to all data strings *D* containing at most 2^{256}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 7 is equal to 37 (hexadecimal).

NOTE Dedicated Hash-Function 7 defined in this clause is commonly called WHIRLPOOL, [4].

13.1 Parameters, functions and constants

13.1.1 Parameters

For this hash-function L_1 = 512, L_2 = 512 and L_H is up to 512.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

13.1.2 Byte ordering convention

In the specification of the round-function of this clause it is assumed that the block input to the round-function is in the form of a matrix M (where all matrices here are 8 by 8 matrices with entries chosen from GF(2^8)), each 512-bit block being made up of such a matrix. A sequence of 64 bytes, $B = (B_0, B_1, \ldots, B_{63})$, shall be interpreted as a matrix M in the following way. The entry in the first row and the first column of the matrix shall be the left-most byte (where the left-most byte corresponds to the most significant byte) of the sequence B (i.e., B_0), the entry in the first row and the second column of the matrix shall be the second left-most byte of B (i.e., B_1), …, and the entry in the eighth row and the eighth column of the matrix shall be the right-most byte of B (i.e., B_{63}). This is performed using function c_0 specified in subclause 13.1.3.

To convert the hash-code from such a matrix to a sequence of bytes, the inverse process of the function c_0 shall be followed.

13.1.3 Functions

To facilitate software implementation, the round-function Φ is described in terms of operations on a matrix M. A sequence of functions c_0, c_1, c_2, c_3, c_4 is used in this round-function. They are defined as follows.

Function c_0 takes a 64-byte sequence, $B = (B_0, B_1, \ldots, B_{63})$ as input, and produces a matrix $Z' = (z'_{ij})$ as output where

$$z'_{ij} = B_{8i+j} \qquad (0 \le i, j \le 7).$$

This means that $Z' = c_0(B)$ if and only if $z'_{ij} = B_{8i+j}$ $(0 \le i, j \le 7)$.

Function c_1 takes a matrix $X'' = (x''_{ij})$ as input and produces another matrix $W' = (w'_{ij})$ as output where

$$w'_{ij} = s[x''_{ij}], \qquad (0 \le i, j \le 7),$$

and where s is a non-linear substitution box. This means $W' = c_1(X'')$ if and only if $w'_{ij} = s[x''_{ij}]$ $(0 \le i, j \le 7)$.

The s box replaces an element $x \in$ GF(2^8) with another element $s[x] \in$ GF(2^8); as specified in Table 2 (the elements in the first column are the 'most significant half' of x, and the elements in the first row are the 'least significant half' of x; for instance, if x = 01010110 = 56 (hexadecimal), $s[x]$ = 49 (hexadecimal) = 01001001.

Table 2 — The s-box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	18	23	C6	E8	87	B8	01	4F	36	A6	D2	F5	79	6F	91	52
1	60	BC	9B	8E	A3	0C	7B	35	1D	E0	D7	C2	2E	4B	FE	57
2	15	77	37	E5	9F	F0	4A	DA	58	C9	29	0A	B1	A0	6B	85
3	BD	5D	10	F4	CB	3E	05	67	E4	27	41	8B	A7	7D	95	D8
4	FB	EE	7C	66	DD	17	47	9E	CA	2D	BF	07	AD	5A	83	33
5	63	02	AA	71	C8	19	49	D9	F2	E3	5B	88	9A	26	32	B0
6	E9	0F	D5	80	BE	CD	34	48	FF	7A	90	5F	20	68	1A	AE
7	B4	54	93	22	64	F1	73	12	40	08	C3	EC	DB	A1	8D	3D
8	97	00	CF	2B	76	82	D6	1B	B5	AF	6A	50	45	F3	30	EF
9	3F	55	A2	EA	65	BA	2F	C0	DE	1C	FD	4D	92	75	06	8A
A	B2	E6	0E	1F	62	D4	A8	96	F9	C5	25	59	84	72	39	4C
B	5E	78	38	8C	D1	A5	E2	61	B3	21	9C	1E	43	C7	FC	04
C	51	99	6d	0D	FA	DF	7E	24	3B	AB	CE	11	8F	4E	B7	EB
D	3C	81	94	F7	B9	13	2C	D3	E7	6E	C4	03	56	44	7F	A9
E	2A	BB	C1	53	DC	0B	9D	6C	31	74	F6	46	AC	89	14	E1
F	16	3A	69	09	70	B6	D0	ED	CC	42	98	A4	28	5C	F8	86

Function c_2 takes a matrix $X'' = (x''_{ij})$ as input and produces another matrix $W' = (w'_{ij})$ as output where

$$w'_{ij} = x''_{(i-j) \bmod 8, j}, \qquad (0 \le i, j \le 7).$$

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

This means that $W' = c_2(X'')$ if and only if $w'_{ij} = x''_{(i-j) \bmod 8, j}$ $(0 \le i, j \le 7)$.

Function c_3 takes a matrix X'' as input and produces another matrix W' as output where

$$W' = X'' \bullet C'',$$

and where C'' is an 8 by 8 circulant matrix with entries chosen from GF(2^8), as specified below:

$$C'' = \begin{bmatrix} 01 & 01 & 04 & 01 & 08 & 05 & 02 & 09 \\ 09 & 01 & 01 & 04 & 01 & 08 & 05 & 02 \\ 02 & 09 & 01 & 01 & 04 & 01 & 08 & 05 \\ 05 & 02 & 09 & 01 & 01 & 04 & 01 & 08 \\ 08 & 05 & 02 & 09 & 01 & 01 & 04 & 01 \\ 01 & 08 & 05 & 02 & 09 & 01 & 01 & 04 \\ 04 & 01 & 08 & 05 & 02 & 09 & 01 & 01 \\ 01 & 04 & 01 & 08 & 05 & 02 & 09 & 01 \end{bmatrix}.$$

This means that $W' = c_3(X'')$ if and only if $W' = X'' \bullet C''$.

Function c_4 takes two matrices $X'' = (x''_{ij})$ and $Y' = (y'_{ij})$ as input and produces a single matrix $W' = (w'_{ij})$ as output where

$$w'_{ij} = x''_{ij} \oplus y'_{ij}, \qquad (0 \le i, j \le 7).$$

This means that $W' = c_4(X'')$ if and only if $w'_{ij} = x''_{ij} \oplus y'_{ij}$ $(0 \le i, j \le 7)$.

13.1.4 Constants

A sequence of constant matrices $A^r = (A^r_{ij})$ $(0 < r \le 10)$ is used in this round-function. The round constant for the r-th round is a matrix, defined as:

$$A^r_{0j} = s[8(r - 1) + j], \qquad (0 \le j \le 7),$$
$$A^r_{ij} = 0, \qquad (1 \le i \le 7, 0 \le j \le 7).$$

13.1.5 Initializing value

The initializing value IV is a string of 512 '0' bits.

13.2 Padding method

The data string D needs to be padded to make it contain a number of bits which is an integer multiple of 512. The padding procedure operates as follows:

1. D is concatenated with a single '1' bit.

2. The result of the previous step is concatenated with between zero and 511 '0' bits such that the length (in bits) of the resultant string is an odd multiple of 256.

3. If the original length of D is L_D, concatenate the string resulting from the previous step with the 256-bit binary representation of L_D, most significant bit first.

In the description of the round-function which follows, each 512-bit data block D_i, $1 \le i \le q$, is treated as a matrix $Z' = (z'_{ij})$ $(0 \le i, j \le 7)$, as specified in subclause 13.1.2, where z'_{00} corresponds to the left-most 8 bits of D_i, and z'_{77} corresponds to the right-most 8 bits of D_i.

NOTE The concatenation of the 256-bit string of L_D in step 3 is such that the 256-bit string is used directly as the second half of the last data matrix; based on the byte ordering convention in Clause 13.1.2, the most significant byte of L_D is the fifth row and the first column and the least significant byte of L_D is the eighth row and the eighth column.

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

13.3 Description of the round-function

The round-function Φ operates as follows. Note that, in this description, we use the symbols W', X'', K_0, K_1, …, K_{10} to denote 13 distinct matrices, each with entries chosen from GF(2^8), which contain values required in the computations.

1. Suppose the 512-bit (first) input to Φ is contained in a matrix Z' with entries chosen from GF(2^8) which is formed by using the byte ordering convention specified in subclause 13.1.2. Suppose also that the 512-bit (second) input to Φ is contained in a matrix Y' with entries chosen from GF(2^8).

2. Let $K_0 := Y'$; and for i = 1 to 10 let

   $K_i := c_4(c_3(c_2(c_1(K_{i-1}))), A^i)$.

   NOTE This step expands the matrix Y' onto a sequence of round keys K_0 , …, K_{10}.

3. Let $X'' := c_4(Z', K_0)$; and for j = 1 to 10 do the following two steps

   (a) $W' := c_4(c_3(c_2(c_1(X''))), K_j)$;

   (b) $X'' := W'$.

4. Let $Y' := W' \oplus K_0 \oplus Z'$.

5. The matrix Y' represents the output of the round-function Φ. After the final iteration of the round-function, the matrix Y' shall be converted to a sequence of 64 bytes using the inverse of the procedure specified in 13.1.2, and where the entry in the first row and the first column of the matrix shall yield the first byte, the entry in the first row and the second column of the matrix the next byte, …, the entry in the eighth row and the eighth column of the matrix the last byte. The 64 bytes shall be converted to a string of 512 bits using the inverse of the procedure specified in clause 6, i.e., the first (left-most) bit will correspond to the most significant bit of the first (left-most) byte, and the 512th (right-most) bit will correspond to the least significant bit of the 64th (right-most) byte.

Figure 6 below shows steps a) and b) of item 3 of the round function Φ in Dedicated Hash-Function 7 (WHIRLPOOL). In the round function Φ, the steps shown in Figure 6 are used 10 times (j = 1, …, 10).

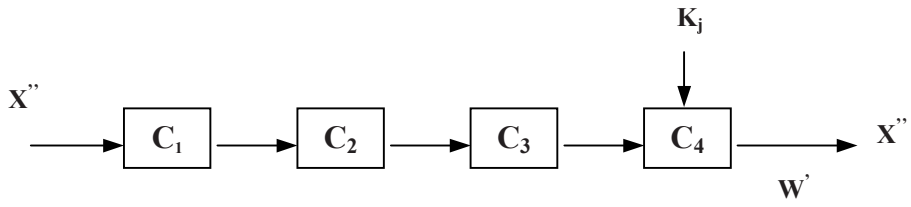

Figure 6 — Part of the round function in Dedicated Hash-Function 7

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

14 Dedicated Hash-Function 8 (SHA-224)

In this clause we specify a padding method, an initialising value, and a round-function for use in the general model for hash-functions described in ISO/IEC 10118-1:2000. The padding method, initialising value and round-function specified here, when used in the above general model, together define Dedicated Hash-Function 8. This dedicated hash-function can be applied to all data strings D containing at most 2^{64}-1 bits.

The ISO/IEC hash-function identifier for Dedicated Hash-Function 8 is equal to 38 (hexadecimal).

NOTE Dedicated Hash-Function 8 defined in this clause is commonly called SHA-224, [2].

14.1 Parameters, functions and constants

14.1.1 Parameters

For this hash-function $L_1 = 512$, $L_2 = 256$ and $L_H = 224$.

14.1.2 Byte ordering convention

The byte ordering convention for this hash-function is the same as that for the hash-function of clause 10.

14.1.3 Functions

The functions for this hash-function are the same as those for the hash-function of clause 10.

14.1.4 Constants

The constants for this hash-function are the same as those for the hash-function of clause 10.

14.1.5 Initialising value

For this round-function the initialising value, IV, shall always be the following 256-bit string, represented here as a sequence of eight words Y_0, Y_1, Y_2, Y_3, Y_4, Y_5, Y_6, Y_7 in a hexadecimal representation, where Y_0 represents the left-most 32 of the 256 bits:

Y_0 = c1059ed8
Y_1 = 367cd507
Y_2 = 3070dd17
Y_3 = f70e5939
Y_4 = ffc00b31
Y_5 = 68581511
Y_6 = 64f98fa7
Y_7 = befa4fa4

© ISO 2012

Documents
Document Sharing Hub
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

NOTE These values are the low order 32-bits of the values specified in 12.1.5.

14.2 Padding method

The padding method to be used with this hash-function shall be the same as the padding method defined in 10.2.

14.3 Description of the round-function

The round-function to be used with this hash-function shall be the same as the round-function defined in 10.3.

The final 224-bit hash is obtained by truncating the SHA-256-based hash output to its left-most 224 bits.

Documents
Document Sharing Hub
© ISO 2012
Full document details: BS ISO/IEC 10118-3:2004+A1:2006

Annex A
(informative)

Examples

A1 This annex gives examples for the computation of Dedicated Hash-Functions 1-8. For each of the hash-functions, intermediate values derived during the hash-function's operation are given for some examples. A1

Throughout this annex we refer to ASCII coding of data strings; this is equivalent to coding using ISO 646.

A.1 Dedicated Hash-Function 1

NOTE — Reference [3] contains a pseudocode description of Dedicated Hash-Function 1.

A.1.1 Example 1

In this example the data-string is the empty string, i.e., the string of length zero.

The hash-code is the following 160-bit string.

```
9C 11 85 A5 C5 E9 FC 54 61 28 08 97 7E E8 F5 48 B2 25 8D 31
```

A.1.2 Example 2

In this example the data-string consists of a single byte, namely the ASCII-coded version of the letter ‘a’.

The hash-code is the following 160-bit string.

```
0B DC 9D 2D 25 6B 3E E9 DA AE 34 7B E6 F4 DC 83 5A 46 7F FE
```

A.1.3 Example 3

In this example the data-string is the three-byte string consisting of the ASCII-coded version of ‘abc’. This is equivalent to the bit-string: ‘01100001 01100010 01100011’.

After the padding process, the single 16-word block derived from the data-string is as follows.

```
80636261  00000000  00000000  00000000  00000000  00000000  00000000  00000000
00000000  00000000  00000000  00000000  00000000  00000000  00000018  00000000
```

The following are (hexadecimal representations of) the successive values of the variables X_0, X_1, X_2, X_3, X_4, X'_0, X'_1, X'_2, X'_3, X'_4.

```
67452301, EFCDAB89, 98BADCFE, 10325476, C3D2E1F0, 67452301, EFCDAB89, 98BADCFE, 10325476, C3D2E1F0
C3D2E1F0, 3115FC67, EFCDAB89, EB73FA62, 10325476, C3D2E1F0, DDD63FB8, EFCDAB89, EB73FA62, 10325476
10325476, B41192D5, 3115FC67, 36AE27BF, EB73FA62, 10325476, 322E7AE3, DDD63FB8, 36AE27BF, EB73FA62
EB73FA62, 3A35DC50, B41192D5, 57F19CC4, 36AE27BF, EB73FA62, 883EE903, 322E7AE3, 58FEE377, 36AE27BF
36AE27BF, D3786413, 3A35DC50, 464B56D0, 57F19CC4, 36AE27BF, 92B2B79B, 883EE903, B9EB8CC8, 58FEE377
57F19CC4, 0E946720, D3786413, D77140E8, 464B56D0, 58FEE377, F9091FF2, 92B2B79B, FBA40E20, B9EB8CC8
464B56D0, D52BF632, 0E946720, E1904F4D, D77140E8, B9EB8CC8, E5B09992, F9091FF2, CADE6E4A, FBA40E20
D77140E8, 150BD8A8, D52BF632, 519C803A, E1904F4D, FBA40E20, 8B2D9FB3, E5B09992, 247FCBE4, CADE6E4A
E1904F4D, 3D6F601F, 150BD8A8, AFD8CB54, 519C803A, CADE6E4A, E755F422, 8B2D9FB3, C2664B96, 247FCBE4
519C803A, B7B60384, 3D6F601F, 2F62A054, AFD8CB54, 247FCBE4, 5922D09E, E755F422, B67ECE2C, C2664B96
AFD8CB54, B85A0A3F, B7B60384, BD807CF5, 2F62A054, C2664B96, CF24E72C, 5922D09E, 57D08B9D, B67ECE2C
2F62A054, 7F8B38E5, B85A0A3F, D80E12DE, BD807CF5, B67ECE2C, CA6A1C75, CF24E72C, 8B427964, 57D08B9D
BD807CF5, 9DACA495, 7F8B38E5, 6828FEE1, D80E12DE, 57D08B9D, 227F6D84, CA6A1C75, 939CB33C, 8B427964
D80E12DE, BC05F46F, 9DACA495, 2CE395FE, 6828FEE1, 8B427964, 5D801685, 227F6D84, A871D729, 939CB33C
```

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

```
6828FEE1, 1494F053, BC05F46F, B2925676, 2CE395FE, 939CB33C, B3C3F4D5, 5D801685, FDB61089, A871D729
2CE395FE, 85861D02, 1494F053, 17D1BEF0, B2925676, A871D729, 3D16242D, B3C3F4D5, 005A1576, FDB61089
B2925676, 597BF629, 85861D02, 53C14C52, 17D1BEF0, FDB61089, FF459078, 3D16242D, 0FD356CF, 005A1576
17D1BEF0, 6347EF78, 597BF629, 18740A16, 53C14C52, 005A1576, 927E40A8, FF459078, 5890B4F4, 0FD356CF
53C14C52, 45C8FA44, 6347EF78, EFD8A565, 18740A16, 0FD356CF, ACBB994E, 927E40A8, 1641E3FD, 5890B4F4
18740A16, AD2956AF, 45C8FA44, 1FBDE18D, EFD8A565, 5890B4F4, AD30AD24, ACBB994E, F902A249, 1641E3FD
EFD8A565, 5EAF16B7, AD2956AF, 23E91117, 1FBDE18D, 1641E3FD, 6261732E, AD30AD24, EE653AB2, F902A249
1FBDE18D, 41730D4B, 5EAF16B7, A55ABEB4, 23E91117, F902A249, 45ED27AF, 6261732E, C2B492B4, EE653AB2
23E91117, FC0CCBD3, 41730D4B, BC5ADD7A, A55ABEB4, EE653AB2, 243C5668, 45ED27AF, 85CCB989, C2B492B4
A55ABEB4, 042ECC93, FC0CCBD3, CC352D05, BC5ADD7A, C2B492B4, 82F89BD1, 243C5668, B49EBD17, 85CCB989
BC5ADD7A, 4D4D4377, 042ECC93, 332F4FF0, CC352D05, 85CCB989, 5FC74686, 82F89BD1, F159A090, B49EBD17
CC352D05, 5207002B, 4D4D4377, BB324C10, 332F4FF0, B49EBD17, B2720031, 5FC74686, E26F460B, F159A090
332F4FF0, 388278F5, 5207002B, 350DDD35, BB324C10, F159A090, 58A100F8, B2720031, 1D1A197F, E26F460B
BB324C10, 62879D70, 388278F5, 1C00AD48, 350DDD35, E26F460B, 5992068B, 58A100F8, C800C6C9, 1D1A197F
350DDD35, A30A1FD9, 62879D70, 09E3D4E2, 1C00AD48, 1D1A197F, CC290DCA, 5992068B, 8403E162, C800C6C9
1C00AD48, BDA2B31B, A30A1FD9, 1E75C18A, 09E3D4E2, C800C6C9, 863D625E, CC290DCA, 481A2D66, 8403E162
09E3D4E2, F7211DEE, BDA2B31B, 287F668C, 1E75C18A, 8403E162, 6061B5A5, 863D625E, A4372B30, 481A2D66
1E75C18A, B6A665C6, F7211DEE, 8ACC6EF6, 287F668C, 481A2D66, AA98ADB5, 6061B5A5, F5897A18, A4372B30
287F668C, 2D30FA02, B6A665C6, 8477BBDC, 8ACC6EF6, A4372B30, 2999255A, AA98ADB5, 86D69581, F5897A18
8ACC6EF6, C76D12F9, 2D30FA02, 99971ADA, 8477BBDC, F5897A18, 98237631, 2999255A, 62B6D6AA, 86D69581
8477BBDC, 516F84DF, C76D12F9, C3E808B4, 99971ADA, 86D69581, 6C472A90, 98237631, 649568A6, 62B6D6AA
99971ADA, F3FA5B05, 516F84DF, B44BE71D, C3E808B4, 62B6D6AA, 2EAD5672, 6C472A90, 8DD8C660, 649568A6
C3E808B4, D539625E, F3FA5B05, BE137D45, B44BE71D, 649568A6, C5CB48BA, 2EAD5672, 1CAA41B1, 8DD8C660
B44BE71D, D8500C99, D539625E, E96C17CF, BE137D45, 8DD8C660, 05286DFB, C5CB48BA, B559C8BA, 1CAA41B1
BE137D45, 7ECDE5B2, D8500C99, E5897B54, E96C17CF, 1CAA41B1, 88396DD2, 05286DFB, 2D22EB17, B559C8BA
E96C17CF, 681D30B9, 7ECDE5B2, 40326761, E5897B54, B559C8BA, 333F2212, 88396DD2, A1B7EC14, 2D22EB17
E5897B54, 960F7BFD, 681D30B9, 3796C9FB, 40326761, 2D22EB17, C699295B, 333F2212, E5B74A20, A1B7EC14
40326761, 6770E498, 960F7BFD, 74C2E5A0, 3796C9FB, A1B7EC14, BFD68874, C699295B, FC8848CC, E5B74A20
3796C9FB, 75EB06C5, 6770E498, 3DEFF658, 74C2E5A0, E5B74A20, BDDF3474, BFD68874, 64A56F1A, FC8848CC
74C2E5A0, 14FA827A, 75EB06C5, C392619D, 3DEFF658, FC8848CC, 8CBC87E9, BDDF3474, 5A21D2FF, 64A56F1A
3DEFF658, 804B0068, 14FA827A, AC1B15D7, C392619D, 64A56F1A, CDDA6EBF, 8CBC87E9, 7CD1D2F7, 5A21D2FF
C392619D, 475BA81B, 804B0068, EA09E853, AC1B15D7, 5A21D2FF, 656C7DA3, CDDA6EBF, F21FA632, 7CD1D2F7
AC1B15D7, D26BC25D, 475BA81B, 2C01A201, EA09E853, 7CD1D2F7, 76D66CA3, 656C7DA3, 69BAFF37, F21FA632
EA09E853, DBC5A2CB, D26BC25D, 6EA06D1D, 2C01A201, F21FA632, C9B17F72, 76D66CA3, B1F68D95, 69BAFF37
2C01A201, 77367F5E, DBC5A2CB, AF097749, 6EA06D1D, 69BAFF37, 65A60151, C9B17F72, 59B28DDB, B1F68D95
6EA06D1D, 8155A6B4, 77367F5E, 168B2F6F, AF097749, B1F68D95, 33F3AC81, 65A60151, C5FDCB26, 59B28DDB
AF097749, C90C4D38, 8155A6B4, D9FD79DC, 168B2F6F, 59B28DDB, 9BFB827D, 33F3AC81, 98054596, C5FDCB26
168B2F6F, 9762713B, C90C4D38, 569AD205, D9FD79DC, C5FDCB26, DDC8130E, 9BFB827D, CEB204CF, 98054596
D9FD79DC, 7EBF9C32, 9762713B, 3134E324, 569AD205, 98054596, C24C2C79, DDC8130E, EE09F66F, CEB204CF
569AD205, 20EFFA01, 7EBF9C32, 89C4EE5D, 3134E324, CEB204CF, F255847E, C24C2C79, 204C3B77, EE09F66F
3134E324, 75B7117F, 20EFFA01, FE70C9FA, 89C4EE5D, EE09F66F, DCD63949, F255847E, 30B1E709, 204C3B77
89C4EE5D, A96BE4C7, 75B7117F, BFE80483, FE70C9FA, 204C3B77, 5B99238D, DCD63949, 5611FBC9, 30B1E709
FE70C9FA, 5E3201FC, A96BE4C7, DC45FDD6, BFE80483, 30B1E709, B43484F4, 5B99238D, 58E52773, 5611FBC9
BFE80483, 2CF95A98, 5E3201FC, AF931EA5, DC45FDD6, 5611FBC9, 52325A09, B43484F4, 648E356E, 58E52773
DC45FDD6, 1393F0C3, 2CF95A98, C807F178, AF931EA5, 58E52773, D015577D, 52325A09, D213D2D0, 648E356E
AF931EA5, BB49CCF7, 1393F0C3, E56A60B3, C807F178, 648E356E, BB9C87C4, D015577D, C9682548, D213D2D0
C807F178, 6A330EB4, BB49CCF7, 4FC30C4E, E56A60B3, D213D2D0, B1BB1A2E, BB9C87C4, 555DF740, C9682548
E56A60B3, 14E58204, 6A330EB4, 2733DEED, 4FC30C4E, C9682548, AC77F96D, B1BB1A2E, 721F12EE, 555DF740
4FC30C4E, 79AAF53E, 14E58204, CC3AD1A8, 2733DEED, 555DF740, 1774D326, AC77F96D, EC68BAC6, 721F12EE
2733DEED, 210769B3, 79AAF53E, 96081053, CC3AD1A8, 721F12EE, A625F112, 1774D326, DFE5B6B1, EC68BAC6
CC3AD1A8, F44B53A7, 210769B3, ABD4F9E6, 96081053, EC68BAC6, 5DCA4D12, A625F112, D34C985D, DFE5B6B1
96081053, 7C1E3640, F44B53A7, 1DA6CC84, ABD4F9E6, DFE5B6B1, EBC4D9C6, 5DCA4D12, 97C44A98, D34C985D
ABD4F9E6, 06B59EE8, 7C1E3640, 2D4E9FD1, 1DA6CC84, D34C985D, 095F37FD, EBC4D9C6, 29344977, 97C44A98
1DA6CC84, C422C3CD, 06B59EE8, 78D901F0, 2D4E9FD1, 97C44A98, 5BBEE487, 095F37FD, 13671BAF, 29344977
2D4E9FD1, AD864025, C422C3CD, D67BA01A, 78D901F0, 29344977, BF5B2529, 5BBEE487, 7CDFF425, 13671BAF
78D901F0, 29A83BB5, AD864025, 8B0F3710, D67BA01A, 13671BAF, FB5747C5, BF5B2529, FB921D6E, 7CDFF425
D67BA01A, 626E3910, 29A83BB5, 190096B6, 8B0F3710, 7CDFF425, DD935A5F, FB5747C5, 6C94A6FD, FB921D6E
8B0F3710, A719D8BC, 626E3910, A0EED4A6, 190096B6, FB921D6E, 27754F3A, DD935A5F, 5D1F17ED, 6C94A6FD
190096B6, BA84C782, A719D8BC, B8E44189, A0EED4A6, 6C94A6FD, 4F5CA4A5, 27754F3A, 4D697F76, 5D1F17ED
A0EED4A6, 9F6887A9, BA84C782, 6762F29C, B8E44189, 5D1F17ED, 325AFE7E, 4F5CA4A5, D53CE89D, 4D697F76
B8E44189, 3A88288C, 9F6887A9, 131E0AEA, 6762F29C, 4D697F76, 86AFE021, 325AFE7E, 7292953D, D53CE89D
6762F29C, AB23F78F, 3A88288C, A21EA67D, 131E0AEA, D53CE89D, C97F9EA1, 86AFE021, 6BF9F8C9, 7292953D
131E0AEA, 7299044A, AB23F78F, 20A230EA, A21EA67D, 7292953D, 9F60751C, C97F9EA1, BF80861A, 6BF9F8C9
A21EA67D, 6A3F10CF, 7299044A, 8FDE3EAC, 20A230EA, 6BF9F8C9, 1E9CE713, 9F60751C, FE7A8725, BF80861A
20A230EA, 1A1B904D, 6A3F10CF, 641129CA, 8FDE3EAC, BF80861A, C13F038A, 1E9CE713, 81D4727D, FE7A8725
8FDE3EAC, 0B2CDC01, 1A1B904D, FC433DA8, 641129CA, FE7A8725, BF627814, C13F038A, 739C4C7A, 81D4727D
641129CA, D563BFDC, 0B2CDC01, 6E413468, FC433DA8, 81D4727D, 5FCCBADE, BF627814, FC0E2B04, 739C4C7A
```

The hash-code is the following 160-bit string.

```
8E B2 08 F7 E0 5D 98 7A 9B 04 4A 8E 98 C6 B0 87 F1 5A 0B FC
```

Documents
Document Sharing Hub
© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006

A.1.4 Example 4

In this example the data-string is the 14-byte string consisting of the ASCII-coded version of

‘message digest’

The hash-code is the following 160-bit string.

```
5D 06 89 EF 49 D2 FA E5 72 B8 81 B1 23 A8 5F FA 21 59 5F 36
```

A.1.5 Example 5

In this example the data-string is the 26-byte string consisting of the ASCII-coded version of

‘abcdefghijklmnopqrstuvwxyz’

The hash-code is the following 160-bit string.

```
F7 1C 27 10 9C 69 2C 1B 56 BB DC EB 5B 9D 28 65 B3 70 8D BC
```

A.1.6 Example 6

In this example the data-string is the 62-byte string consisting of the ASCII-coded version of

‘ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789’

The hash-code is the following 160-bit string.

```
B0 E2 0B 6E 31 16 64 02 86 ED 3A 87 A5 71 30 79 B2 1F 51 89
```

A.1.7 Example 7

In this example the data-string is the 80-byte string consisting of the ASCII-coded version of eight repetitions of

‘1234567890’

The hash-code is the following 160-bit string.

```
9B 75 2E 45 57 3D 4B 39 F4 DB D3 32 3C AB 82 BF 63 32 6B FB
```

A.1.8 Example 8

In this example the data-string is the 56-byte string consisting of the ASCII-coded version of

‘abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq’

After the padding process, the two 16-word blocks derived from the data-string are as follows.

```
64636261  65646362  66656463  67666564  68676665  69686766  6A696867  6B6A6968
6C6B6A69  6D6C6B6A  6E6D6C6B  6F6E6D6C  706F6E6D  71706F6E  00000080  00000000

00000000  00000000  00000000  00000000  00000000  00000000  00000000  00000000
00000000  00000000  00000000  00000000  00000000  00000000  000001C0  00000000
```

The following are (hexadecimal representations of) the successive values of the variables X_0, X_1, X_2, X_3, X_4, X'_0, X'_1, X'_2, X'_3, X'_4, obtained during the processing of the first block.

```
67452301, EFCDAB89, 98BADCFE, 10325476, C3D2E1F0, 67452301, EFCDAB89, 98BADCFE, 10325476, C3D2E1F0
```

Documents
Document Sharing Hub

© ISO 2012

Full document details: BS ISO/IEC 10118-3:2004+A1:2006