ISO/IEC ~~DIS~~FDIS 14888-4:~~2023~~2024(E)

ISO/IEC JTC 1 SC 27/WG 2

Secretariat: DIN

Date: ~~2023-08-28~~2024-02-20

Information security — Digital signatures with appendix — Part 4: Stateful hash-based mechanisms

Style Definition: Normal: Font: (Asian) Japanese, Line spacing: At least 12 pt, Tab stops: Not at 20.15 pt

Style Definition: Heading 1: Font: Font color: Auto, (Asian) Japanese, Indent: Left: 0 pt, First line: 0 pt, Line spacing: Exactly 13.5 pt, Outline numbered + Level: 1 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0 pt + Tab after: 21.6 pt + Indent at: 21.6 pt, Don't hyphenate, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 21.6 pt

Style Definition: Heading 2: Font: Bold, Font color: Auto, (Asian) Japanese, Line spacing: Exactly 12.5 pt, Outline numbered + Level: 2 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0 pt + Tab after: 18 pt + Indent at: 0 pt, Don't hyphenate, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 18 pt + 20.15 pt

Style Definition: Heading 3: Font: Bold, Font color: Auto, (Asian) Japanese, Line spacing: Exactly 11.5 pt, Outline numbered + Level: 3 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0 pt + Tab after: 36 pt + Indent at: 0 pt, Don't hyphenate, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 20.15 pt

Style Definition: Heading 4: Font: Bold, Font color: Auto, (Asian) Japanese, Line spacing: Exactly 11.5 pt, Outline numbered + Level: 4 + Numbering Style: 1, 2, 3, … + Start at: 1 + Alignment: Left + Aligned at: 0 pt + Tab after: 54 pt + Indent at: 0 pt, Don't hyphenate, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 20.15 pt

Style Definition ...

Style Definition ...

Formatted: Font: 11.5 pt, Font color: Auto

Formatted ...

Formatted: Font: 11.5 pt, Font color: Auto

Formatted: Font: Font color: Auto

Formatted: Font: Not Bold, Font color: Auto

Formatted: Font: Font color: Auto

Formatted: Font: Not Bold, Font color: Auto

Formatted: Font: Font color: Auto

Formatted: Font: Not Bold, Font color: Auto

Formatted: Font: Font color: Auto

Formatted: Font: Not Bold, Font color: Auto

Formatted: Font: Font color: Auto

Formatted: Font: Not Bold, Font color: Auto

Formatted ...

ISO ~~copyright office~~Copyright Office
CP 401 • ~~Ch. de Blandonnet 8~~

CH-1214 Vernier, Geneva
Phone: + 41 22 749 01 11
~~Fax: +41 22 749 09 47~~

~~Email: copyright@iso.org~~

Email: copyright@iso.org
Website: ~~www.iso.org~~www.iso.org
Published in Switzerland

Formatted: Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border)

Formatted: English (United States)

Formatted: zzCopyright, Indent: Left: 0 pt, Right: 0 pt, Space Before: 0 pt, No page break before, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 25.7 pt + 481.15 pt

Formatted

Formatted: Font: 11 pt, Font color: Auto

Formatted: English (United States)

Formatted: English (United States)

Formatted: Font: 11 pt, Font color: Auto

Formatted: English (United States)

Formatted: Font: 11 pt, Font color: Auto

Formatted: English (United States)

Formatted: English (United States)

Formatted: zzCopyright, Indent: Left: 0 pt, First line: 0 pt, Right: 0 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 25.7 pt + 481.15 pt

Formatted: Font: 11 pt, Font color: Auto

Formatted: English (United States)

Formatted: English (United States)

Formatted: Font: 11 pt, Font color: Auto

Formatted: zzCopyright, Indent: Left: 0 pt, First line: 0 pt, Right: 0 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 25.7 pt + 481.15 pt

Formatted: English (United States)

Formatted: English (United States)

Formatted: Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border)

Formatted: Right, Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Contents

Formatted: Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Line spacing: single, Tab stops: Not at 487.6 pt

Formatted: Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Formatted: Font: 11 pt, Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Tab stops: Not at 487.6 pt

Formatted: Right, Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Formatted: Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Line spacing: single, Tab stops: Not at 487.6 pt

Formatted: Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Formatted: Font: 11 pt, Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Tab stops: Not at 487.6 pt

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.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see ~~www.iso.org/directives~~www.iso.org/directives or ~~www.iec.ch/members_experts/refdocs~~www.iec.ch/members_experts/refdocs).

ISO and IEC draw attention to the possibility that the implementation of this document may involve the use of (a) patent(s). ISO and IEC take no position concerning the evidence, validity or applicability of any claimed patent rights in respect thereof. As of the date of publication of this document, ISO and IEC had not received notice of (a) patent(s) which may be required to implement this document. However, implementers are cautioned that this may not represent the latest information, which may be obtained from the patent database available at ~~www.iso.org/patents and https://patents.iec.ch~~www.iso.org/patents and https://patents.iec.ch. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see ~~www.iso.org/iso/foreword.html~~.www.iso.org/iso/foreword.html. In the IEC, see ~~www.iec.ch/understanding-standards~~www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

A list of all parts in the ISO/IEC 14888 series can be found on the ISO and IEC websites.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at ~~www.iso.org/members.html~~www.iso.org/members.html and ~~www.iec.ch/national-committees~~www.iec.ch/national-committees.

Formatted: Right, Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Formatted: Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Line spacing: single, Tab stops: Not at 487.6 pt

Formatted: Space After: 36 pt, Line spacing: Exactly 12 pt, Border: Top: (No border), Bottom: (No border), Left: (No border), Right: (No border), Between : (No border), Tab stops: Not at 235.15 pt + 470.3 pt

Formatted: Font: 12 pt, Bold, Font color: Auto

Introduction

Digital ~~Signatures~~signatures with appendix are designed to offer integrity, authentication and non-repudiation. ISO/IEC 14888-2 specifies the class of digital signature mechanisms in which the security is based on the difficulty of integer factorization. ISO/IEC 14888-3 specifies the class in which the security is based on computing discrete logarithms ~~(see ISO/IEC 14888-3).~~ Unfortunately, if and when a large-scale general purpose quantum computer becomes available, all of these techniques will no longer be secure for practical key sizes~~[.[1].]~~.

This document specifies a class of digital signatures whose security depends only on the security of the underlying hash function. At the time of publication of this document, standardized hash functions are believed to be secure even against attacks using large scale quantum computers. Hence, the schemes specified in this document do not suffer from the same problems as the schemes specified in ISO/IEC 14888-2 and ISO/IEC 14888-3.

The ~~Hash-Based Signature~~hash-based signature (HBS) schemes specified in this document are stateful schemes, ~~–~~ whereby the private key is part of the state of the scheme. This means that at every signature generation, state information held by the signer must be updated, ~~since~~as otherwise the security of the scheme is compromised. Therefore, when deploying any of the schemes specified in this document, it is ~~mandatory to implement~~ expected that robust state-management practices are implemented to ensure that state information is correctly updated.

Formatted: Font: 11 pt, Font color: Auto

Formatted: Footer, Space Before: 0 pt, After: 0 pt, Tab stops: Not at 487.6 pt

Information security — Digital signatures with appendix — Part 4: Stateful hash-based mechanisms

1 Scope

This document specifies stateful digital signature mechanisms with appendix, where the level of security is determined by the security properties of the underlying hash function.

This document also provides requirements for implementing basic state management, which is needed for the secure deployment of the stateful schemes described in this document.

2 Normative references

There are no normative references in this document.

3 Terms and definitions

For the purposes of this document, the following terms and definitions given in [external document reference xxx] and the following apply.

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

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

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

3.1
Authenticationauthentication path
list of hash values that show that a specific node belongs to a *Merkle tree* (3.5)

3.2
balanced binary tree
ordered tree in which each node has exactly two other nodes that are directly subordinate

3.3
Binarybinary tree
ordered tree in which each node has at most two other nodes that are directly subordinate

[SOURCE: ISO/IEC 2382:2015, 2121636, modified — notes to entry have been removed.]

3.4
L-tree
unbalanced *binary tree* (3.3) used to compress the Winternitz+ One-Time Signature Scheme (WOTS+) public keys in the eXtended Merkle Signature Scheme (XMSS)

3.5
Merkle tree
balanced binary tree (3.2), where each node of the tree corresponds to the hash of the labels of the child nodes

3.6
one-time signature
digital signature scheme where the security is limited to signing a single message for a given key pair

3.7

~~State~~
state
state information
information regarding key usage stored with an eXtended Merkle Signature Scheme (XMSS) or Leighton-Micali signature scheme (LMS) private key

3.8
~~State~~state management
processes by which the *state* (3.7) is updated with each signature

Note 1 to entry: This is a crucial part of any stateful hash-based signature, as incorrect state information can lead to signature forgeries.

3.9
Winternitz parameter
parameter in the Winternitz *one-time signature* (3.6) scheme, which allows a trade-off between signature size and computation time

4 Symbols and abbreviated terms

4.1 Symbols

`0x00`	the hexadecimal representation of the 0-byte
$a \mid\mid b$	concatenation of the bit sequences *a* and *b* in the order specified
$a * b$	multiplication of *a* and *b*
a & b	bitwise logical AND of two bit strings *a* and *b*
a XOR *b*	bitwise logical EXCLUSIVE OR of two bit strings *a* and *b*
a mod *b*	the remainder when *a* is divided by *b*
$a << x$	the result of left-shifting a bit-string *a* by *x* positions (e.g. `0x08` << 1 = `0x10`)
$a >> x$	the result of right-shifting a bit-string *a* by *x* positions (e.g. `0x08` >> 1 = `0x04`)
ceil(*x*)	the least integer greater than or equal to *x*
d	the number of layers of sub-trees in XMSS-MT
floor(*x*)	the greatest integer less than or equal to *x*
~~*h*~~ *h*	total height of the Merkle tree (or hyper tree for XMSS-MT and HSS)
	NOTE 1 ~~The~~ *h* is the parameter which controls the maximum number of signatures, which are given by 2^h.
lb(*x*)	the base 2-logarithm of *x*
~~*n*~~ *n*	the length in bytes of one element of the private key, public key or signature, and the length of the message representative
m	the arbitrary length message to be signed in XMSS, XMSS-MT, LM-OTS, LMS, and HSS~~.~~
M'	the message representative signed in XMSS and XMSS-MT~~.~~
toByte(*x*,*y*)	~~computes~~computation of the *y*-byte string containing the binary representation of *x* in big-endian byte order
w	the Winternitz parameter used in the XMSS and XMSS_MT algorithms

NOTE 2 It_w controls the trade-off between computation time and signature size. Smaller values of w/W lead to faster computations, while larger values of w lead to smaller signatures.

W	the Winternitz parameter used in the LMS and HSS algorithms. W and w are related as follows: $w = 2^W$
$x[i]$	the ith element of the array x
$[X]_y$	the result of truncating X to its leftmost y bits, e.g. $[\texttt{0xfe}]_4 = \texttt{0xf}$

4.34.2 Abbreviated terms

HBS	hash-based signature
HSS	hierarchical signature scheme
LMS	Leighton-Micali signature scheme
OTS	one-time signature
RBG	non-deterministic random bit generator
WOTS+	Winternitz+ One-Time signature scheme
XMSS	eXtended Merkle Signature Scheme
XMSS-MT	eXtended Merkle Signature Scheme Multi Tree

65 XMSS and XMSS-MT

6.15.1 General

The XMSS scheme is a stateful hash-based signature scheme for which only a limited number of signatures can be created using a particular private key. The security of XMSS is based on the hardness of the Target Collision Resistance (TCR) problem, and XMSS has been proven to be secure in the standard model (see [13]).

The XMSS-MT scheme, a variant of the XMSS scheme, is a stateful hash-based signature scheme that supports a larger number of signatures than XMSS. XMSS-MT inherits all the security properties of XMSS [14].

The OIDs for these algorithms shall be in accordance with Annex A. Test vectors can be found in Annex C.

6.25.2 Common Building Blocksbuilding blocks

6.2.15.2.1 General

The XMSS and the XMSS-MT schemes are described in a unified way since they employ common building blocks.

6.2.25.2.2 Address Formatformat

The *ADRS* input has three different address formats (see Table 1). Each layout is appropriate for a different step of the algorithms:

- the OTS address format is for the hash calls in the one-time signature schemes;
- the L-tree address format is for hashes used in the L-trees;
- the hash tree address format is for the Merkle-tree construction.

An L-tree is an unbalanced binary hash tree used to compute the leaves of the main XMSS binary hash tree.

Formatted ... Formatted ... Formatted: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted: Font: Italic, Font color: Auto Formatted ... Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted ... Formatted: Font: Font color: Auto Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted ... Formatted ... Formatted: Font: 12 pt Formatted: cite_bib, Font: Formatted: cite_bib, Font: Superscript Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted ... Formatted: Font: Font color: Auto Formatted Table

Each *ADRS* address format consists of six 32-bit fields and one 64-bit field. The fields are encoded as unsigned integers. The *layerAddress* field refers to the layer in a multi-tree construction (0 if not using multi-trees). The *treeAddress* refers to the position (from left to right) of a tree in a multi-tree construction (0 if not using multi-trees). The *field* type differentiates the three *ADRS* layouts: 0 for OTS, 1 for L-tree and 2 for Hash tree. The field *keyAndMask* is used to distinguish if the hash call intends to generate a key (0) or a bitmask (1) for an OTS address. In the case of the L-tree and Hash tree address, *keyAndMask* is: (0) to generate a key, (1) to generate the most significant n bytes of the $2n$-byte bitmask and (2) to generate the least significant n bytes of the $2n$-byte bitmask.

Table 1 — XMSS ~~Address Formats~~address formats

OTS ~~Address~~address	L-tree ~~Address~~address	Hash tree ~~Address~~address
layerAddress (32 bits)	layerAddress (32 bits)	layerAddress (32 bits)
treeAddress (64 bits)	treeAddress (64 bits)	treeAddress (64 bits)
type = 0 (32 bits)	type = 1 (32 bits)	type = 2 (32 bits)
OTSAddress (32 bits)	ltreeAddress (32 bits)	padding = 0 (32 bits)
chainAddress (32 bits)	treeHeight (32 bits)	treeHeight (32 bits)
hashAddress (32 bits)	treeIndex (32 bits)	treeIndex (32 bits)
keyAndMask (32 bits)	keyAndMask (32 bits)	keyAndMask (32 bits)

NOTE Each XMSS Address in this table is 32-bytes.

The field *OTSAddress* encodes the index of the OTS key pair within the tree, the *chainAddress* encodes the chain index and finally the *hashAddress* encodes the hash function call index within the chain.

In the L-tree address layout, the *ltreeAddress* field encodes the index of the leaf computed with this L-tree. The *treeHeight* encodes the height of the node used as input for the next computation inside the L-tree. The *treeIndex* encodes the index of the node at that height, inside the L-tree.

In the ~~Tree Hash~~tree hash address format, the *padding* field is always 0. The *treeHeight* encodes the height of the tree node being used for the next computation, followed by the *treeIndex* which is the node index at that height. For the L-tree address layout, the *keyAndMask* field can be: 0 (key), 1 (first bitmask), and 2 (second bitmask).

~~6.2.3~~5.2.3 Required cryptographic functions

~~6.2.3.1~~5.2.3.1 General

The XMSS scheme uses the following cryptographic primitives:

— Keyed Hash-Function F(*KEY*, *M*), where *KEY* is an n-byte key, *M* is an n-byte message and the output is n bytes.

- Keyed Hash-Function H(*KEY*, *M*), where *KEY* is an *n*-byte key, *M* is a 2*n*-byte message and the output is *n* bytes.
- Keyed Hash-Function H_msg(*KEY*, *M*), where *KEY* is a 3*n*-byte key, *M* is an arbitrary length message and the output is *n* bytes.
- Pseudo-Random Function PRF(*KEY*, *M*), where *KEY* is an *n*-byte key, *M* is a 32-byte message and the output is *n* bytes.
- Pseudo-Random Function PRF_{Keygen}(*KEY*, *M*), where *KEY* is an *n*-byte key, *M* is a 32-byte message and the output is *n* bytes. This function is optional and only used if WOTS+ private keys are generated pseudo-randomly.
- Pseudo-Random Function PRF_{Keygen_MT}(*KEY*, *M*), where *KEY* is an *n*-byte key, *M* is a 32-byte message and the output is *n* bytes. This function is only used in XMSS_keygen if called from XMSS_MT_keygen.
- A non-deterministic random number generator RBG that provides at least a security level of 8*n* bits.

These functions shall be implemented with SHA2-256 (Dedicated Hash-Function 4 ~~of~~defined in ISO/IEC 10118-3), SHAKE256 (~~Annex C.2 of~~see ISO/IEC 10118-3~~)~~ :2018, C.2) as described below (see Annex B for further considerations on these instantiations).

~~6.2.3.2~~5.2.3.2 Functions Based on SHA2-256

When using SHA2-256 as the underlying hash function and *n*=32, the following constructions shall be used.

- F(*KEY*, *M*): SHA2-256(toByte(0, 32) || *KEY* || *M*)
- H(*KEY*, *M*): SHA2-256(toByte(1, 32) || *KEY* || *M*)
- H_msg(*KEY*, *M*): SHA2-256(toByte(2, 32) || *KEY* || *M*)
- PRF(*KEY*, *M*): SHA2-256(toByte(3, 32) || *KEY* || *M*)
- PRF_{Keygen}(*KEY*, *M*): SHA2-256(toByte(4, 32) || *KEY* || *M*)
- PRF_{Keygen_MT}(*KEY*, *M*): SHA2-256(toByte(5, 32) || *KEY* || *M*)

When using SHA2-256 as the underlying hash function and *n*=24, the following constructions shall be used.

- F(*KEY*, *M*): SHA2-256/192(toByte(0, 4) || *KEY* || *M*)
- H(*KEY*, *M*): SHA2-256/192(toByte(1, 4) || *KEY* || *M*)
- H_msg(*KEY*, *M*): SHA2-256/192(toByte(2, 4) || *KEY* || *M*)
- PRF(*KEY*, *M*): SHA2-256/192(toByte(3, 4) || *KEY* || *M*)
- PRF_{Keygen}(*KEY*, *M*): SHA2-256/192(toByte(4, 4) || *KEY* || *M*)
- PRF_{Keygen_MT}(*KEY*, *M*): SHA2-256/192(toByte(5, 4) || *KEY* || *M*)

~~6.2.3.3~~5.2.3.3 Functions Based on SHAKE

When using SHAKE256 and *n*=32, the following constructions shall be used.

- F(*KEY*, *M*): SHAKE256(toByte(0, 32) || *KEY* || *M*, 256)
- H(*KEY*, *M*): SHAKE256(toByte(1, 32) || *KEY* || *M*, 256)

Formatted: Font: 12 pt, Font color: Auto
Formatted: Font: 12 pt, Font color: Auto
Formatted: Font: 12 pt, Font color: Auto
Formatted Table

— H_msg(*KEY*, *M*): SHAKE256(toByte(2, 32) || *KEY* || *M*, 256)

— PRF(*KEY*, *M*): SHAKE256(toByte(3, 32) || *KEY* || *M*, 256)

— PRF_{Keygen}(*KEY*, *M*): SHAKE256(toByte(4, 32) || *KEY* || *M*, 256)

— PRF_{Keygen_MT}(KEY, M): SHAKE256(toByte(5, 32) || *KEY* || *M*, 256)

When using SHAKE256 and *n*=24, the following constructions shall be used.

— F(*KEY*, *M*): SHAKE256(toByte(0, 4) || *KEY* || *M*, 192)

— H(*KEY*, *M*): SHAKE256(toByte(1, 4) || *KEY* || *M*, 192)

— H_msg(*KEY*, *M*): SHAKE256(toByte(2, 4) || *KEY* || *M*, 192)

— PRF(*KEY*, *M*): SHAKE256(toByte(3, 4) || *KEY* || *M*, 192)

— PRF_{Keygen}(*KEY*, *M*): SHAKE256(toByte(4, 4) || *KEY* || *M*, 192)

— PRF_{Keygen_MT}(*KEY*, *M*): SHAKE256(toByte(5, 4) || *KEY* || *M*, 192)

~~6.2.4~~5.2.4 Auxiliary functions

~~6.2.4.1~~5.2.4.1 General

Both XMSS and XMSS-MT make use of two auxiliary functions, namely the base_w function and the chain function. The specifications of these auxiliary functions are given in 5.2.4.2 and 5.2.4.3.

~~6.2.4.2~~5.2.4.2 base_w auxiliary function

This function is used in the signing and verification process to convert a string of bytes into a sequence of base *w* integers (i.e. integers between 0 and *w*-1).

Algorithm: base_w(*X*, *w*, *out_len*).

Input: A sequence of bytes *X* = *X*[0], … , *X*[*len_x* - 1] of length *len_x*, the base *w* which shall be 4 or 16, and the output length *out_len*.

Output: A sequence of integers *Q* = *Q*[0],..., *Q*[*out_len*-1] of length *out_len* from the set {0,1,...,*w*-1}

Steps:

a) Set *in* = 0, *out* = 0, *total* = 0, *bits* = 0.

b) For *i* from 0 to *out_len* - 1:

——1) If *bits* is equal to 0 then

——i) Set *total* = *X*[*in*].

——ii) Set *in* = *in* + 1.

——iii) Set *bits* = 8.

——2) Set *bits* = *bits* - lb(*w*).

——3) Set *Q*[*out*] = (*total* >> *bits*) & (*w* - 1).

——4) Set *out* = *out* + 1.

c) Return *Q*.

Formatted: Font: 12 pt, Font color: Auto
Formatted: Font: 12 pt, Font color: Auto
Formatted: Font: 12 pt, Font color: Auto
Formatted Table