INTERNATIONAL STANDARD

ISO/IEC 7816-8

Fifth edition
2021-08

AMENDMENT 1
2023-11

Identification cards — Integrated circuit cards —

Part 8: Commands and mechanisms for security operations

AMENDMENT 1: Interoperability for the interchange of security operations using quantum safe cryptography

Cartes d'identification — Cartes à circuit intégré —

Partie 8: Commandes et mécanismes pour les opérations de sécurité

AMENDEMENT 1: Interopérabilité pour l'échange d'opérations de sécurité utilisant la cryptographie quantique sécurisée

Reference number
ISO/IEC 7816-8:2021/Amd. 1:2023(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Foreword

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

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 or 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. 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. In the IEC, see www.iec.ch/understanding-standards.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 17, *Cards and security devices for personal identification*.

A list of all parts in the ISO/IEC 7816 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 and www.iec.ch/national-committees.

Identification cards — Integrated circuit cards —

Part 8:
Commands and mechanisms for security operations

AMENDMENT 1: Interoperability for the interchange of security operations using quantum safe cryptography

Normative references

Change ISO/IEC 7816-4 to ISO/IEC 7816-4:2020

Terms and definitions

Add the following new term entries:

3.8
certificate chain
ordered list of certificates that starts with an end-entity certificate, includes one or more certificate authority (CA) certificates, and ends with the end-entity certificate's root CA certificate, where each certificate in the chain is the certificate of the CA that issued the previous certificate

3.9
classic McEliece
code-based quantum safe key encapsulation algorithm retained in the course of the third round of the National Institute of Standards and Technology (NIST) contest

3.10
code-based
cryptosystem based on error correcting code

3.11
common parameter
public value that is used to control the operation of a cryptographic algorithm or that is used by a cryptographic algorithm to compute outputs

3.12
crypto-agility
property that permits changing or upgrading cryptographic algorithms or parameters

[SOURCE: ETSI TR 103 619[14]]

3.13
crystals-dilithium
lattice-based quantum safe signature algorithm as selected by the National Institute of Standards and Technology (NIST) contest for standardization

3.14
crystals-kyber
lattice-based quantum safe key encapsulation algorithm as selected by the National Institute of Standards and Technology (NIST) contest for standardization

3.15
discrete logarithm
computation of logarithms with regards to multiplicative cyclic groups

Note 1 to entry: This problem is considered as a hard mathematical problem when the number is large, and is the basis of some asymmetric cryptography scheme.

3.16
El Gammal
asymmetric encryption/decryption protocol based on the discrete logarithm problem and using a throwable mask

3.17
falcon
lattice-based quantum safe signature algorithm as selected by the National Institute of Standards and Technology (NIST) contest for standardization

3.18
frodoKEM
lattice-based quantum safe key encapsulation algorithm retained in the course of the third round of the National Institute of Standards and Technology (NIST) contest

3.19
hash-based
digital signatures constructed using hash functions

3.20
hybrid certificate
certificate secured using both a regular asymmetric signature algorithm and a quantum safe signature algorithm

3.21
key encapsulation algorithm
class of encryption techniques designed to secure symmetric cryptographic key material for transmission using asymmetric (public-key) algorithms

Note 1 to entry: The term used by National Institute of Standards and Technology (NIST) is “Key Encapsulation Mechanism”.

3.22
lattice-based
cryptosystem based on lattice problems

3.23
Leighton Micali signature
LMS
hash-based quantum safe signature algorithm

Note 1 to entry: This algorithm is defined in Reference [15].

3.24
Merkle tree
data structure where the data is hashed and combined until there is a singular root hash that represents the entire structure

3.25
NTRU
lattice-based quantum safe key encapsulation algorithm retained in the course of the third round of the National Institute of Standards and Technology (NIST) contest

3.26
public key parameters
set of components and characteristics describing the public key

3.27
private key parameters
set of components and characteristics describing the private key

3.28
quantum safe
characteristic of an algorithm or cryptosystem that is secure against both quantum and classical computers

3.29
quantum safe cryptography
cryptographic systems that are secure against both quantum and classical computers

Note 1 to entry: This encompasses any cryptosystem (e.g. symmetric, asymmetric).

3.30
Saber
lattice-based quantum safe key encapsulation algorithm retained in the course of the third round of the National Institute of Standards and Technology (NIST) contest

3.31
sphincs+
hash-based quantum safe signature algorithm as selected by the National Institute of Standards and Technology (NIST) contest for standardization

Note 1 to entry: See Table AMD.1.20.

3.32
XMSS
hash-based quantum safe signature algorithm

Note 1 to entry: The algorithm is defined in Reference [15].

Clause 4

Add the following before BCD:

AlgID Algorithm Identifier

Add the following after CRT:

CRT(2) optimized mode of computation for RSA algorithm exploiting the properties of the Chinese remainder theorem, and making use of a private key under a reduced form

Add the following after GQ2:

HSS hierarchical signature system

Add the following after LDS:

LMS Hash-based Leighton Micali Signature

LM-OTS Leighton Micali One-Time Signature

Add the following after OID:

PKI Public Key Infrastructure

Add the following after PSO:

QSC Quantum Safe Cryptography

Add the following after SEID:

SFM regular mode of computation of RSA algorithm

Add the following after TLV:

XMSS Extended Merkle (Hierarchical) Signature Scheme

Subclause 5.1

Add the following at the end of the first paragraph of subclause 5.1:

This document aims additionally at providing a set of generic QSC features. These generic features cater to a variety of QSC algorithms. This approach seeks to enhance this document with crypto-agility means and hybrid cryptography capabilities.

Subclause 5.2

Add the following subclause heading under subclause 5.2:

5.2.1 General

Subclause 5.2.1

Replace NOTE 2 with the following:

NOTE 2 The private key can be stored in

— an internal EF the reference of which is known before issuing the command, or

— a DO'7F48' as cardholder private key template, or

— a QSC template DO'7F75' featuring private key with private key parameters, or

— a QSC template DO'7F76' featuring private key with private key parameters alongside common parameters.

Replace NOTE 3 with the following:

NOTE 3 The public key can be stored for example in

— a DO'7F49' as cardholder public key template, or

— a QSC template DO'7F75' featuring exclusively public key parameters, or

— a QSC template DO'7F76' featuring public key parameters alongside common parameters, see examples on Table AMD.1.2 to Table AMD.1.7.

Subclause 5.2.1

In the bullet list under NOTE 4, add “or” to the first and second bullet. Replace “.” with “, or” at the end of the fourth bullet; append to the fourth bullet the following:

— a QSC template DO’7F75’ (INS = ‘47’) encapsulating public key and QSC key component data objects from Table AMD.1.1, or

— a QSC template DO’7F76’ (INS = ‘47’) encapsulating public key and QSC common parameter component data objects from Table AMD.1.1.

EXAMPLE 1 The QSC templates are provided in Table AMD.1.2 and Table AMD.1.5 for the hash-based signature algorithm.

EXAMPLE 2 The QSC templates are provided in Table AMD.1.3, Table AMD.1.4, Table AMD.1.6 and Table AMD.1.7 for the lattice-based algorithm.

Subclause 5.2.1

Change the EXAMPLE to EXAMPLE 3

Subclause 5.2.1

Change the sentence right before existing Table 4 as follows:

For the coding of the DO stating information about the private part of the key pair, Table 4 applies except for QSC. For QSC, the private key shall be nested into template DO’7F75’ or DO’7F76’ using the DOs as proposed in Table AMD.1.1.

Subclause 5.2

Add the following new subclauses 5.2.2 to 5.2.6 after 5.2.1:

5.2.2 QSC template data objects

This QSC template is a generic template catering to any type of QSC, e.g. hash-based or lattice-based, and its parameters, features general purposes data objects in its key information part whereas the key specific parameters, e.g. for private or public key or common parameters, are nested in template key value part. QSC template data objects are described in Table AMD.1.1.

Table AMD.1.1 — QSC key components within template DO‘7F75’, DO‘7F76’ or DO‘7F77’ with cardinality indication

Tag	Description	Cardinality	
		at creation	at update
’06’	**OID**, conditional (present at creation and if DO’80’ is absent) [c]	0 or 1	0
’80’	**AlgID**, conditional (present at creation and if first DO’06’ is absent)	0 or 1	0
’81’	**Key type**, conditional (present at creation, absent at update); unique per template	1 or 2	0
’82’	**Key size**, conditional (present at creation, absent at update) [b]	From 0 up to n	0
’8E’	**Identifier of external common parameters**, conditional (optional at creation, absent at update)	0 or 1	0
’8F’	**Identifier of the QSC template**, conditional (optional at creation and at update)	0 or 1	0 or 1
’5C’	**Parameters Organization Descriptor,** optional (may nest tag ’81’, tag ’82’ or tag ‘83’) [d] **‘81’** denotes private key parameters **‘82’** denotes public key parameters **‘83’** denotes common parameters	From 0 up to n	From 0 up to n
’06’	**Application-specific QSC parameters mapping**	0 or 1	0 or 1
’9X’	**Key value** (either private, public key, or common parameters), inferred by Key type (DO’81’), by template tag (DO’7F75’, DO’7F76’, DO’7F77’) and by Parameters Organization Descriptor (DO’5C’) [a]	0 or 1	0 or 1

NOTE 1 Use of DO’83’ to DO’8D’ is RFU, can serve for future indication of further characteristics, e.g. type of memory (transient, persistent);

NOTE 2 If common parameters are merged within private or public key template, the Key Type (DO’81’) can be replicated (e.g. hash-based protocol).

NOTE 3 DO’8F’ is the Private/Public Key identifier to be used by the external world to request a digital signature computation/verification or encryption/decryption from the ICC, e.g. the value of DO’8F’ can be assigned to DO’84’ to identify a private key from within a CRT (control reference template). DO’8F’ can be assigned to DO’83’ value in order to identify the public key from within a CRT.

NOTE 4 Additional private tags that can be used in template ‘7F75’ to nest application specific data, e.g. pre-generated private keys or proprietary parameter(s) involved in key generation or state maintenance, are outside interoperability scope.

NOTE 5 DO’8E’ is no longer useful if common parameters and private/public key are nested in the same template.

[a] Key value component (‘9X’) may be provisioned entirely at once or in several parts at creation and update.

[b] DO’82’ may be replicated for each level of the Merkle Tree with HSS hash-based scheme (see Figures F.6 and F.7).

[c] This DO’06’ describing the algorithm shall be the first DO in the template; only one OID describing the algorithm shall be present. Furthermore, OID describing the algorithm and DO AlgID are mutually exclusive. When occurring right after a Tag List DO’5C’, a DO’06’ is meant for QSC parameter mapping and is not mutually exclusive with AlgID; this DO’06’ value is out of scope of this document.

[d] The usage of DO’81’, DO‘82’ and DO‘83’ are described under EXAMPLES 4 in the text below.

For hash-based LMS private key under template DO’7F75’, see Table AMD.1.8; for hash-based LMS merged private key and common parameters under template DO’7F76’, see Table AMD.1.9. For lattice-based Crystals-Dilithium private key under template DO’7F75’, see Table AMD.1.10; for lattice-based Crystals-Dilithium private key and common parameters merged under template DO’7F76’, see Table AMD.1.12. For lattice-based Falcon private key under template DO’7F75’, see Table AMD.1.11; for lattice-based Falcon private key and common parameters merged under template DO’7F76’, see Table AMD.1.13.

The templates DO‘7F77’ nesting only common parameters for hash-based LMS key, lattice-based Crystals-Dilithium, lattice-based Falcon are presented respectively on Table AMD.1.14, Table AMD.1.15 and Table AMD.1.16.

Each QSC template comprises key information data objects and key value data objects with Parameters Organization Descriptor data object. Key information data objects consist of OID, AlgID, Key type, Key size, identifier of external common parameters and identifier of the QSC template data object. Key value data objects are set of DO'9X' inferred by Key type (DO'81'), see Table AMD.1.22 for details. Five different structures can be nested under QSC templates DO'7F75', DO'7F76' or DO'7F77' as follows:

— private key under QSC template DO'7F75';

— public key under QSC template DO'7F75';

— common parameters under QSC template DO'7F77';

— common parameters and private key under QSC template DO'7F76';

— common parameters and public key under QSC template DO'7F76'.

Usually, common parameter(s) are implicitly known as soon as the cryptographic algorithm is known. Therefore template '7F75' may suffice instead of '7F76'.

The problem of scarcity of context-specific tags arises whenever more than 15 parameters are needed to describe a structure within a QSC template, i.e. with lattice-based Crystals-Dilithium. Therefore, to allow for nesting replicated and ordered primitive tags (bit b6 set to 0) while covering more than 15 parameters the optional Tag List data object DO'5C' (see ISO/IEC 7816-4:2020, 8.4.3) serving as Parameters Organization Descriptor shall be used right after the key information to indicate recursive tag numbering (i.e. cyclic tag numbering) whenever employed within the template. The use of DO'5C' may denote rearrangement of the QSC templates that are mapped explicitly in this document or in ISO/IEC 7816-6 (e.g. it can be useful when some mapped parameters are hidden in a QSC template or when common parameter and public or private key parameters are swapped).

{'5C'-L-(tag1- tag2-…- tagN)} denotes replicated tags in the template as follows: tag numbering starts with tag'90' till tag1; then tag numbering resumes with tag'90' right after tag1 and incremented by one till tag2; then tag numbering resumes with tag'90' right after tag2 and incremented by one till next tag in the Tag List DO'5C' etc. See example of implementation of DO'5C' below.

EXAMPLES 1

{'5C'-'01'-('95')} means tags numbering of a sequence starting from DO'90' is incremented by one and replicated once right after tag '95' resulting in a template as follows:

{'7F75'-L-{key information-{'5C'-01-('95')}-{'90'-L-V}-{'91'-L-V}-{'92'-L-V}-{'93'-L-V}-{'94'-L-V}-{'**95'**-L-V}-{'**90**'-L-V}-{'91'-L-V}-{'92'-L-V}…}}

{'5C'-'02'-('95''93')} means tags numbering of a sequence starting from DO'90' is incremented by one and replicated twice right after tag '95' then tag '93' resulting in a template as follows:

{'7F75'-L-{key information-{'5C'-'02'-('95', '93')}-{'90'-L-V}-{'91'-L-V}-{'92'-L-V}-{'93'-L-V}-{'94'-L-V}-{'**95'**-L-V}-{'**90**'-L-V}-{'91'-L-V}-{'92'-L-V}-{'**93'**-L-V}-{'**90**'-L-V}-{'91'-L-V}-{'92'-L-V}-{'93'-L-V}-{'94'-L-V…}}

DO'5C' may be used even when less than 15 parameters are present in the template to indicate the way such template shall be updated or expanded with further data objects when replication is deemed useful.

For tag numbering that restarts with a tag different from DO'90', data object '5C' shall use the following coding with the context-specific DO'9F2X' as indicator.

{'5C'-L-(tag1-'9F2X_1'-tag2-'9F2X_2'-…-tagN-'9F2X_n')} denotes replicated tags in the template as follows: tag numbering starts with tag'90' till tag1; then tag numbering resumes with tag'9X_1' right after tag1 and incremented by one till tag2; then tag numbering resumes with tag'9X_2' right after tag2 and incremented by one till next tag in the Tag List DO'5C' etc. See example of implementation of DO'5C' below.

EXAMPLES 2

{‘5C’-‘03’-(‘95’-‘9F22’)} means tags numbering of a sequence starting from DO‘92’ is incremented by one and replicated once right after tag ‘95’ resulting in a template as follows:

{‘7F75’-L-{key information-{‘5C’-03-(‘95’-‘9F22’)}-{‘90’-L-V}-{‘91’-L-V}-{‘92’-L-V}-{‘93’-L-V}-{‘94’-L-V}-{‘**95**’-L-V}-{‘**92**’-L-V}-{‘93’-L-V}-{‘94’-L-V}…}}

{‘5C’-‘07’-(‘95’-‘9F23’-‘94’-‘9F26’-‘97’)} means:

— for ‘95’-‘9F23’ : right after tag ‘95’, a new sequence is started with tag numbering from ‘93’ (instead of DO’90’), and incremented by one;

— for ‘94’-‘9F26’: right after tag ‘94’, a new sequence is started with tag numbering from ‘96’ (instead of DO’90’), and incremented by one;

— for ‘97’: right after tag ‘97’, a new sequence is started with tag numbering from ‘90’, and incremented by one.

{‘7F75’-L-{key information-{‘5C’-‘07’-(‘95’-‘9F23’-‘94’-‘9F26’-‘97’)}-{‘90’-L-V}-{‘91’-L-V}-{‘92’-L-V}-{‘93’-L-V}-{‘94’-L-V}-{‘**95**’-L-V}-{‘**93**’-L-V}-{‘**94**’-L-V}-{‘**96**’-L-V}-{‘**97**’-L-V}-{‘90’-L-V}-{‘91’-L-V}-}-{‘92’-L-V}-{‘93’-L-V}-{‘94’-L-V}-{‘95’-L-V}-{‘96’-L-V}-{‘97’-L-V}-{‘98’-L-V}-{‘99’-L-V}…}}

Where there is no tag before the DO’9F2X’ in the DO’5C’, i.e. {‘5C’-L-(‘9F2X’-…)}, it denotes that tag numbering starts with tag’9X’. See example of implementation of DO’5C’ below.

EXAMPLES 3

{‘5C’-‘02’-(‘9F23’)} means that following current DO’5C’, tag numbering starts with tag ‘93’, resulting in a template as follows:

{‘7F75’-L-{key information-{‘5C’-‘02’-(‘9F23’)}-{‘**93**’-L-V}-{‘94’-L-V}-{‘95’-L-V}…}}

In addition to its properties described above, DO’5C’ may serve to organize parameters into dedicated containers within QSC templates, whereby simplifying explicitly template parsing without ambiguity. To this aim, the three data objects DO’81’, DO’82’ and DO’83’ are nested in DO’5C’ to indicate respectively the presence of private key parameters, public key parameters and common parameters followed by a series of tags as described above, see examples below.

EXAMPLES 4

{‘5C’-‘02’-(‘81’-‘92’)} denotes that following DO’5C’, the current QSC template contains private key parameters numbered as a sequence starting from DO‘90’ and incremented by one, and replicated once right after tag ‘92’, resulting in a template as follows:

{‘7F75’-L-{key information-{‘5C’-‘02’(‘81’, ‘92’)}-{‘90’-L-V}-{‘91’-L-V}-{‘**92’**-L-V}-{‘**90’**-L-V}-{‘91’-L-V}-{‘92’-L-V}-{‘93’-L-V}-{‘94’-L-V}…}}

{‘5C’-‘01’-(‘83’)} denotes that following DO’5C’, the current QSC template contains, common parameters numbered as a sequence starting from DO‘90’, resulting in a template as follows:

{‘7F77’-L-{key information-{‘5C’-‘01’-(‘83’)}-{‘90’-L-V}-{‘91’-L-V}-{‘92’-L-V}-{‘90’-L-V}-{‘94’-L-V}…}}

The main benefit of such a tag organization with DO’5C’ is that it makes different arbitrary parameters numbering stay interoperable with each other within the same data object generation.

Table AMD.1.2 to Table AMD.1.16 describe key template structures featuring ‘5C’DO usage.

When using Tag List DO‘5C’ for parameters ordering into containers, the actual mapping of such parameters (i.e. assignment of a given DO’9X’ to a specific QSC component), may depend on the implementation, in which case, the Tag List may be immediately followed with a DO’06’ denoting the implementation authority (e.g. a specification) that determines the mapping. Otherwise, the by-default mapping is the one described in this document, for instance Crystals-Dilithium, Falcon, LMS. Crystals-Dilithium key sizes are given as examples in key templates; actual values may be checked against dilithium v3.1. See Reference [16].

Table AMD.1.2 — QSC template DO'7F75' encapsulating Hash-based LMS public key

Tag	L	Value			Note
0x7F75	Var.				QSC template encapsulating hash-based LMS public key
		Tag	**Length**	**Value**	
		0x80	0x04	0xFE0101XX	Algorithm identifier: Hash-based LMS and 'XX' indicating hash function (see 5.2.3)
		0x81	0x02	0x0012	Key type: ALG_TYPE_LMS_PUBLIC (see 5.2.4)
		0x82	0x02	0x0100	Key size (e.g. SHA-256)
		0x8E	Var.		Identifier of external common parameters
		0x8F	Var.		Identifier of the QSC template (same as private key)
		0x5C	0x01	0x82	Parameters Organization Descriptor see Table AMD.1.1
		0x90	Var.		Public key (root of tree)

NOTE 1 This table describes an example of LMS parameters for public key with leaves (level 2) and root (level 1) as per example from Reference [23] TestCase 2 (p.54). As a general description, in a hash-based hierarchical signature system (HSS) comprising a hierarchy of Merkle trees, L represents the number of stages of Merkle trees. The number of leaves of each Merkle tree valuates to 2^h where h is the height of the Merkle tree, and the leaves represent tuples of signature key (secret and public) on top of those trees. The parameter q is a 32-bit integer (called index) that indicates the leaf of the Merkle tree where the OTS public key can be looked up. Each leaf of the tree is comprised of the value of the public key of an LM-OTS public/private key pair. Each node within the tree has a unique node number, and the leaves have node numbers 2^h, (2^h)+1, (2^h)+2, ..., (2^h)+(2^h)-1. In general, the j-th node at level i has node number 2^i + j. Thus the root node has node number 1 (i.e. "root level = 2^0 = 1"), see Reference [23] for details.

NOTE 2 0xABCD denotes the hexadecimal number 'ABCD' for easier reference from implementers.

See hash-based schemes taxonomy in Annex F, Figure F.1.

Table AMD.1.3 — QSC template DO'7F75' encapsulating Lattice-based Crystals-Dilithium public key

Tag	L	Value			Note
0x7F75	Var.				QSC template encapsulating Lattice-based Crystals-Dilithium public key
		Tag	**Length**	**Value**	
		0x80	0x04	see 5.2.3	AlgID
		0x81	0x02	see Table AMD.1.22	keytype "Crystals-Dilithium public key"
		0x82	0x02	0x04A0	keysize (1312 = rho + t1) values: 1312, 1952, 2592
		0x8E	0x01	0x02	Identifier of external common parameters
		0x8F	0x01	0x01	Identifier of the QSC template (same as private = keypair)
		0x5C	0x01	0x82	see Table AMD.1.1
		0x90	0x20	0x.....	*rho(length=32), same for all Crystals-Dilithium NIST levels*
		0x91	0x820500	0x.....	*t1(length=1280)* for NIST levels 2, 3 and 5, values: 1280, 1920, 2560

NOTE 1 See keysize according to NIST security levels in Reference [16].

NOTE 2 0xABCD denotes the hexadecimal number 'ABCD' for easier reference from implementers.

Table AMD.1.4 — QSC template DO'7F75' encapsulating Lattice-based Falcon public key

Tag	L	Value			Note
0x7F75	Var.	**Tag**	**Length**	**Value**	
		0x80	0x04	0xFE0105XX	AlgID Lattice-based FALCON and 'XX' indicating hash function (see 5.2.3)
		0x81	0x02	see Table AMD.1.22	keytype "Falcon public key"
		0x82	0x02	0x0381	keysize (=897) values = 897,1793
		0x8E	0x01	0x02	Identifier of external common parameters
		0x8F	0x01	0x01	Identifier of the QSC template (same as private = keypair)
		0x5C	0x01	0x82	
		0x90	0x820381	0x......	*h(length=897)* value = 897,1793

NOTE 1 For recommended Falcon parameters see, e.g. public keysize on[17].

NOTE 2 0xABCD denotes the hexadecimal number 'ABCD' for easier reference from implementers.

Table AMD.1.5 — QSC template DO'7F76' encapsulating Hash-based LMS public key and common parameters

Tag	L	Value			Note
0x7F76	Var.	**Tag**	**Length**	**Value**	
		0x80	0x04	see 5.2.3	AlgID
		0x81	0x02	see Table AMD.1.22	keytype "LMS public key"
		0x82	0x01	0x20	keysize (e.g. SHA256)
		0x8F	0x01	0x01	Identifier of the QSC template (same as private = keypair)
		0x5C[a]	0x01	0x83	Parameters Organization Descriptor; indicates common parameters, see Table AMD.1.1
		0x90	0x10	0xD08F...	*LMS Identifier level1*
		0x91	0x04	0x00000006	*LMS type level 1 (e.g. LM_SHA256_M32_H10)*
		0x92	0x04	0x00000003	*LM-OTS type level 1 (e.g. LMOTS_SHA256_N32_W4)*
		0x5C	0x01	0x82	Parameters Organization Descriptor; indicates public key parameters, see Table AMD.1.1
		0x90	0x20	0x32A5...	*public key (root of the tree)*
		0x5C	0x01	0x83	Parameters Organization Descriptor; indicates common parameters, see Table AMD.1.1
		0x90	0x10	0x215F...	*LMS Identifier level2*
		0x91	0x04	0x00000005	*LMS type level 2*
		0x92	0x04	0x00000004	*LM-OTS type level 2*

NOTE 1 Parameter values borrowed from example in Reference [23] TestCase 2 (p.54) with leaves = level 2 and root = level 1, see Note 1 to Table AMD.1.2.

NOTE 2 DO'8E' (identifier of external common parameters) is no longer useful if common parameters and public key are nested in the same template.

NOTE 3 0xABCD denotes the hexadecimal number 'ABCD' for easier reference from implementers.

[a] Instead of using multiple instances of Tag List DO'5C', one unique instance may be coded as '5C058392829083' for the same purpose.

