Stream: Internet Engineering Task Force (IETF)
RFC: 0000
Updates: 3279
Category: Standards Track
Published: 11 October 2019
ISSN: 2070-1721

Author: G.C. C. F. Pereira
evolutionQ Inc. / IQC, UWaterloo

Internet X.509 Public Key Infrastructure: Additional Post-Quantum Signature Algorithms

Abstract

This document updates [RFC3279] by specifying additional algorithm identifiers and ASN.1 encoding rules for the nine selected post-quantum digital signature algorithms in the second round of NIST standardization process [NIST-PQ] when using SHA3 or SHAKE as the underlying hashing algorithms. This specification applies to the Internet X.509 Public Key infrastructure (PKI) [RFC5280] and its post-quantum counterpart [PQ-X.509], when post-quantum digital signatures are used to sign certificates and certificate revocation lists (CRLs).

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc0000.

Copyright Notice

Table of Contents

1. Introduction

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This document profiles material under standardization by the NIST Post-Quantum Cryptography standardization process [NIST-PQ]. In particular, it provides the definition of algorithm identifiers and ASN.1 encoding formats for post-quantum digital signatures and subject public keys used in the post-quantum setting of the Internet X.509 Public Key Infrastructure (PKI) [RFC5280], [PQ-X.509]. The nine digital signature algorithm candidates in the second round of NIST process are considered along with their multiple security categories. Although not all nine algorithms may become actual standards, this specification will certainly cover the standardized ones.

Precisely, this document defines additional contents for the "signatureAlgorithm", "signatureValue", and "signature" fields within Internet X.509 certificates and CRLs [RFC5280] when these objects are signed using post-quantum signature algorithms with SHA3 or SHAKE hash algorithms [FIPS202]. The SHA3 and SHAKE instances explicitly used in the object identifiers (OID) definitions are the ones adopted by the candidates of the second round specification of NIST.

The OIDs provided are defined under the NIST signature algorithm branch in the OID tree [NIST-OIDS], in particular, the branch joint-iso-itu-t -> country -> us -> organization -> gov -> csor -> nistAlgorithm -> sigAlgs. Note that sigAlgs contains identifiers ranging from 1-16, allocated for (EC)DSA and RSA. This specification reserves identifiers starting from 20 and above for the new post-quantum signatures with their different parameter sets.

This document extends [RFC3279], in particular Sections 2.1, 2.2.2, and 2.2.3 and complements the update [RFC5758], which focuses on the specification of extra classical signature algorithms. OIDs defined here are crucial for [PQ-X.509], which extends [RFC5280] for accepting post-quantum X.509 certificates.

1.1. Terminology

- B: bytes
- CRL: Certificate Revocation List
- KiB: Kibibytes
- MiB: Mibibytes
- NIST: National Institute of Standards and Technology.
- OID: Object Identifier
- SHA3: Secure Hash Algorithm-3
- XOF: eXtendable-Output hash Function

2. Hash Functions

This section identifies hash functions for use with post-quantum digital signature algorithms in [PQ-X.509]. The hash functions SHA3-256, SHA3-384, SHA3-512 produce 256-bit, 384-bit and 512-bit outputs, respectively. On the other hand, functions SHAKE-128 and SHAKE-256 provide extendable (and thus variable) output sizes. They are all described in the "Secure Hash Standard" [FIPS202]. It is important to note that most of signature candidates in the NIST process adopt SHAKE which is the eXtendable-Output Function (XOF) of SHA3. The OIDs for SHAKE instances are retrieved from [NIST-OIDS]. The hash functions adopted by the NIST candidates have the following OIDs:

id-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) 8 }

id-sha3-384 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) 9 }

id-sha3-512 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) 10 }

id-shake-128 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) 11} }

id-shake-256 OBJECT IDENTIFIER ::= { joint-iso-itu-t(2) country(16) us(840) organization(1) gov(101) csor(3) nistAlgorithm(4) hashAlgs(2) 12} }

Where id-sha3-<output size> refers to the Permutation-Based Hash functions and id-shake-<output size> to the eXtendable-Output Hash functions defined in [FIPS202]. When one of these OIDs appears in an AlgorithmIdentifier, all implementations MUST accept both NULL and absent parameters as legal and equivalent encodings. Conforming certification authority (CA) implementations SHOULD use SHA3-256, SHA3-384, SHA3-512, SHAKE-128 or SHAKE-256 when generating post-quantum certificates or CRLs according to [PQ-X.509].

3. Post-quantum Digital Signature Algorithms

This section defines OIDs for the following signature algorithms: Crystals-Dilithium, Falcon, GeMSS, LUOV, MQDSS, PICNIC, qTESLA, Rainbow and SPHINCS+ when instantiated with SHA3-256, SHA3-384, SHA3-512, SHAKE-128 or SHAKE-256. Note that each algorithm can offer parameter sets for at most five quantum security categories (numbered 1 to 5), as suggested by NIST [NIST-PQ].

Instead of following the OID name convention for classical signatures [RFC5758] consisting of id-<signature algorithm>-with-<hash function> that uniquely identifies parameter sets in those cases, the convention id-<signature algorithm>-<categoryX>-<hash function> is employed here in order to avoid potential ambiguities appearing in the post-quantum cases. To see this, note that it is possible to find parameter sets for different security categories using the same underlying hash function. For instance, SHAKE-128 (or SHAKE-256) is frequently used across multiple security categories of a same post-quantum signature. Thus the combination algorithm name plus hash output size does not uniquely identify a parameter set. The same issue happens if algorithm name and security category is used since a single security category can have multiple parameter sets with different hash functions. For example, SPHINCS+ have options with SHA3 and SHAKE for a same security category. Remark. Some signature algorithms offer parameter sets whose security may lie in between two different security categories due to the lack of a precise understanding of the main attacks or due to a loose reduction. In those cases, the convention adopted is id-<signature algorithm>-<categoryXY>[-<hash function>] where X and Y are the lower and higher categories.

3.1. CRYSTALS-Dilithium Signature

CRYSTALS-Dilithium is a digital signature whose security is based on the hardness of lattice problems [CRYSTALS-Dilithium]. It provides four parameter sets for the security categories 1, 2, 3 and 4, and offers a comparative small footprint for the combination public key + signature size, which ranges from about 2-5 KiB. All operations (KeyGen, Sign and Verify) are relatively efficient to compute (hundreds of Kcycles using AVX2). CRYSTALS-Dilithium SHALL be used in conjunction with SHAKE-256 for all security categories above.

When SHAKE-256 is used, the OIDs are:

id-dilithium-category1-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-dilithium-category1-shake(3) 20 }.

id-dilithium-category2-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-dilithium-category2-shake(3) 21 }.

id-dilithium-category3-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-dilithium-category3-shake(3) 22 }.

id-dilithium-category4-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-dilithium-category4-shake(3) 23 }.

When the id-dilithium-category1-shake256, id-dilithium-category2-shake256, id-dilithium-category3-shake256 or id-dilithium-category4-shake256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-dilithium-category1-shake256, id-dilithium-category2-shake256, id-dilithium-category3-shake256 or id-dilithium-category4-shake256.

3.2. FALCON Signature

FALCON is a digital signature whose security is based on the hardness of lattice problems [FALCON-ref]. It provides two parameter sets for security categories 1 and 4-5 and also offers relatively small footprints for the public key + signature size metric, which ranges from about 1.5-3.0 KiB. In terms of computational efficiency, KeyGen is relatively slow (tens of Mcycles), Sign is moderate (hundreds of Kcycles) and Verify is relatively fast (hundreds of Kcycles). FALCON SHALL be used in conjunction with SHAKE-256 for all security categories above.

When SHAKE-256 is used, the OIDs are:

id-falcon-category1-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-falcon-category1-shake(3) 24 }.

id-falcon-category45-shake256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-falcon-category45-shake(3) 25 }.

When the id-falcon-category1-shake256 or id-falcon-category45-shake256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-falcon-category1-shake256, or id-falcon-category45-shake256.

3.3. GeMSS Signature

GeMSS is a digital signature whose security is based on the hardness of solving non-linear system of multivariate equations over finite fields [GeMSS-ref]. It provides nine parameter sets, three for each one of the security categories 1, 3 and 5. For each category, GeMSS uses a name convention to distinguish between the three parameter sets, i.e., GeMSS, BlueGeMSS and RedGeMSS. In terms of space, public-keys are relatively large (about 400 KiB - 3 MiB) but signatures are very small (about 33-75 B). In terms of computational efficiency, KeyGen is relatively slow (hundreds of Mcycles), Sign is moderate (few million cycles) and Verify is relatively fast (hundreds of Kcycles). GeMSS SHALL be used in conjunction with a SHA3 hash function with output sizes of 256, 384 and 512 bits for matching the three security categories above, respectively.

When SHA3 is used, the OIDs are:

id-gemss-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category1-sha3(3) 26 }.

id-gemssblue-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category1-sha3(3) 27 }.

id-gemssred-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category1-sha3(3) 28 }.

id-gemss-category3-sha3-384 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category3-sha3(3) 29 }.

id-gemssblue-category3-sha3-384 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category3-sha3(3) 30 }.

id-gemssred-category3-sha3-384 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category3-sha3(3) 31 }.

id-gemss-category5-sha3-512 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category5-sha3(3) 32 }.

id-gemssblue-category5-sha3-512 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category5-sha3(3) 33 }.

id-gemssred-category5-sha3-512 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-gemss-category5-sha3(3) 34 }.

When the id-gemss[color]-category1-sha3-256, id-gemss[color]-category3-sha3-384 or id-gemss[color]-category5-sha3-512 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-gemss[color]-category1-sha3-256, id-gemss[color]-category3-sha3-384, or id-gemss[color]-category5-sha3-512.

3.4. LUOV Signature

The LUOV is a digital signature whose security is based on the hardness of solving non-linear system of multivariate equations over finite fields [LUOV]. It provides 6 parameter sets, two for each one of the security categories 2, 4 and 5. In order to differentiate parameter sets within the same security category, LUOV uses the name convention LUOV-r-m-v, where r, m, and v are integers indicating the extension degree, number of equations and number of vinegar variables, respectively. Note that it is likely that r, m and v may slightly change over time as cryptanalysis evolve and fixing specific numbers is likely to make this specification obsolete in a near future. Therefore we use the name convention LUOV and LUOV-high, where LUOV simply means the smaller extension degree r (between the two possible parameter sets) and LUOVhigher means the larger extension degree r. In terms of space, LUOV public keys have moderate size (12-75 KiB) and signatures are small (300-4400 B). In terms of computational efficiency, KeyGey, Sign and Verify have moderate speed (a few Mcycles, hundreds of Kcycles and hundreds of Kcycles, respectively). LUOV SHALL be used in conjunction with SHAKE-128 for matching category 2 or SHAKE-256 for categories 4 and 5.

When SHAKE is used, the OIDs are:

id-luov-category2-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category2-shake(3) 35 }.

id-luovhigher-category2-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category2-shake(3) 36 }.

id-luov-category4-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category4-shake(3) 37 }.

id-luovhigher-category4-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category4-shake(3) 38 }.

id-luov-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category5-shake(3) 39 }.

id-luovhigher-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-luov-category5-shake(3) 40 }.

When the id-luov[-higher]-category2-shake-128, id-luov[-higher]-category4-shake-256 or id-luov[-higher]-category5-shake-256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-luov[-higher]-category2-shake-128, id-luov[-higher]-category4-shake-256 or id-luov[-higher]-category5-shake-256.

3.5. MQDSS Signature

The MQDSS is a digital signature whose security is based on the hardness of solving non-linear system of multivariate equations over finite fields [MQDSS]. It provides two parameter sets, one for security category 1-2 (in between) and another for category 3-4. In terms of space, public keys are small (46-64 B) and signatures are moderate (about 2.6-5.5 KB). From a computational efficiency point of view, KeyGen, Sign and Verify have moderate speed (few million cycles). MQDSS SHALL be used in conjunction with SHAKE-256 for matching the security categories above.

When SHAKE is used, the OIDs are:

id-mqdss-category12-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-mqdss-category12-shake(3) 41 }.

id-mqdss-category34-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-mqdss-category34-shake(3) 42 }.

When the id-mqdss-category12-shake-256 or id-mqdss-category34-shake-256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-mqdss-category12-shake-256 or id-mqdss-category34-shake-256.

3.6. PICNIC Signature

The PICNIC is a digital signature whose security is based on the hardness of breaking symmetric primitives like block ciphers and hash functions [PICNIC]. It provides nine parameter sets, three for each one of the security categories 1, 3 and 5. For each category, PICNIC uses a name convention to distinguish between the three parameter sets, i.e., picnic-L<X>-FS, picnic-L<X>-UR and picnic2-L<X>-FS, where <X> specifies the category. In terms of space, PICNIC public keys are small (32-64 B) and signatures have moderate sizes (1.7-26.2 KiB). From a computational efficiency point of view, KeyGen is fast (tens of Kcycles), Sign and Verify have moderate speed (a few up to hundreds of Mcycles). PICNIC SHALL be used in conjunction with SHAKE-128 for matching category 1 or SHAKE-256 for categories 3 and 5.

When SHAKE is used, the OIDs are:

id-picnicl1fs-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category1-shake(3) 43 }.

id-picnicl1ur-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category1-shake(3) 44 }.

id-picnic2l1fs-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category1-shake(3) 45 }.

id-picnicl1fs-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category3-shake(3) 46 }.

id-picnicl1ur-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category3-shake(3) 47 }.

id-picnic2l1fs-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category3-shake(3) 48 }.

id-picnicl1fs-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category5-shake(3) 49 }.

id-picnicl1ur-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category5-shake(3) 50 }.

id-picnic2l1fs-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-picnic-category5-shake(3) 51 }.

When the id-<picnic name>-category1-shake-128, id-<picnic name>-category3-shake-256 or id-<picnic name>-category5-shake-256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-<picnic name>-category1-shake-128, id-<picnic name>-category3-shake-256 or id-<picnic name>-category5-shake-256.

3.7. qTesla Signatures

The qTesla is a digital signature whose security is based on the hardness of lattice problems [qTesla]. qTesla features two types of design with respect to security, i.e., provable and heuristic security. For the provably-secure versions, two parameter sets are provided for categories 1 and 3. For the heuristic design, 10 parameter sets are given targeting security categories 1, 2, 3 and 5. Half of the heuristic parameters are intended for compressed public keys. After round 2 submission to NIST, all the compressed parameters were broken and acknowledged by the authors. Therefore, the compressed parameter sets are omitted in this document, and thus the remaining 5 heuristic plus the two provably-secure are presented. The heuristic versions of qTesla adopt the name convention qTesla-<X>[-size] where <X> is a Roman numeral for the category and -size means optimized for size. The provably-secure versions adopt the convention qTesla-p-<X>. qTesla SHALL be used in conjunction with SHAKE-128 for matching category 1 or SHAKE-256 for matching categories 2, 3 and 5.

When SHAKE is used, the OIDs are:

id-qteslai-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category1-shake(3) 52 }.

id-qteslais-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category1-shake(3) 53 }.

id-qteslapi-category1-shake-128 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category1-shake(3) 54 }.

id-qteslai-category2-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category2-shake(3) 55 }.

id-qteslais-category2-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category2-shake(3) 56 }.

id-qteslai-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category3-shake(3) 57 }.

id-qteslais-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category3-shake(3) 58 }.

id-qteslapi-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category3-shake(3) 59 }.

id-qteslai-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category5-shake(3) 60 }.

id-qteslais-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-qtesla-category5-shake(3) 61 }.

When the id-<qtesla name>-category1-shake-128, id-<qtesla name>-category2-shake-256, id-<qtesla name>-category3-shake-256 or id-<qtesla name>-category5-shake-256 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-<qtesla name>-category1-shake-128, id-<qtesla name>-category2-shake-256, id-<qtesla name>-category3-shake-256 or id-<qtesla name>-category5-shake-256.

3.8. Rainbow Signatures

The Rainbow is a digital signature whose security is based on the hardness of solving non-linear system of multivariate equations over finite fields [RAINBOW]. It provides 6 parameter sets, two for each one of the security categories 1, 3 and 5. Half of the parameter sets is intended for a compressed key variant of Rainbow. The following name convention for distinguishing between compressed and non-compressed variants is adopted: id-rainbow[-compressed]-category<X>-<hash function> In terms of space, Rainbow public keys are moderate to large (58-492 KiB for the compressed versions) and signatures are small (64-204 B). From a computational efficiency point of view, KeyGen has moderate speed (tens of Mcyles), Sign and Verify are fast (hundreds of Kcycles). Rainbow SHALL be used in conjunction with SHA3-256, SHA3-384 and SHA3-512 for matching categories 1, 3 and 5, respectively.

When SHA3 is used, the OIDs are:

id-rainbow-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-rainbow-category1(3) 62 }.

id-rainbow-category3-sha3-384 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-rainbow-category3(3) 63 }.

id-rainbow-category5-sha3-512 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-rainbow-category5(3) 64 }.

When the id-rainbow-category1-sha3-256, id-rainbow-category3-sha3-384 or id-rainbow-category5-sha3-512 algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-rainbow-category1-sha3-256, id-rainbow-category3-sha3-384 or id-rainbow-category5-sha3-512.

3.9. SPHINCS+ Signature

SPHINCS+ is a digital signature whose security is based on problems from hash functions [SPHINCS-PLUS]. It provides a reasonable amount of parameter sets per security category covering categories 1, 3 and 5. It adopts the name convention SPHINCS+-<W>-<X>-<Y>-<Z> where W is the underlying hash function, X can be one of the integers 128, 192 or 256 indicating the hash output size and thus the security category, Y can be either s (size-optimized implementation) or f (fast implementation) and Z can be either simple (no bitmasks used in the underlying hash) or robust (bitmasks included). In terms of space, SPHINCS+ public keys are small (32-64 B) and signatures have moderate sizes (8-50 KiB). From a computational efficiency point of view, KeyGen, Sign and Verify have moderate to slow speed (ranging from a few to hundreds of Mcycles). This document considers only instances based on the NIST standardized hash functions [FIPS202], i.e., SHAKE and SHA3.

When SHAKE or SHA3 is used, the OIDs are:

id-sphincsp128ssimple-category1-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category1-shake(3) 65 }.

id-sphincsp128srobust-category1-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category1-shake(3) 66 }.

id-sphincsp128fsimple-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category1-sha3(3) 67 }.

id-sphincsp128frobust-category1-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category1-sha3(3) 68 }.

id-sphincsp192ssimple-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category3-shake(3) 69 }.

id-sphincsp192srobust-category3-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category3-shake(3) 70 }.

id-sphincsp192fsimple-category3-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category3-sha3(3) 71 }.

id-sphincsp192frobust-category3-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category3-sha3(3) 72 }.

id-sphincsp256ssimple-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category5-shake(3) 73 }.

id-sphincsp256srobust-category5-shake-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category5-shake(3) 74 }.

id-sphincsp256fsimple-category5-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category5-sha3(3) 75 }.

id-sphincsp256frobust-category5-sha3-256 OBJECT IDENTIFIER ::= { joint-iso-ccitt(2) country(16) us(840) organization(1) gov(101) csor(3) algorithms(4) id-sphincsp-category5-sha3(3) 76 }.

When the id-sphincsp-<X>-<Y>-<Z>-category1-<hash function>, id-sphincsp-category3-<hash function> or id-sphincsp-category5-<hash function> algorithm identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding SHALL omit the parameters field. That is, the AlgorithmIdentifier SHALL be a SEQUENCE of one component, the OID id-sphincsp-<X>-<Y>-<Z>-category1-<hash function>, id-sphincsp-category3-<hash function> or id-sphincsp-category5-<hash function>.

4. References

[CRYSTALS-Dilithium] Ducas, L., Kiltz, E., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and D. Stehle, "Crystals-dilithium: A lattice-based digital signature scheme", IACR Cryptology ePrint Archive , 2017 , <https://eprint.iacr.org/2017/633>.

[FALCON-ref] Fouque, P-A., Hoffstein, J., Kirchner, P., Lyubashevsky, V., Pornin, T., Prest, T., Ricosset, T., Seiler, G., Whyte, W., and Z. Zhang, "Falcon: Fast-Fourier lattice-based compact signatures over NTRU", IACR Cryptology ePrint Archive , 2018 , <https://falcon-sign.info/>.

[FIPS202] National Institute of Standards and Technology (NIST), "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions", August 2015 , <http://dx.doi.org/10.6028/NIST.FIPS.202>.

[GeMSS-ref] Casanova, A., Faugere, J-C., Macario-Rat, G., Patarin, J., Perret, L., and J. Ryckeghem, "GeMSS: A Great Multivariate Short Signature", 2017 , <https://www-polsys.lip6.fr/Links/NIST/GeMSS.html>.

[LUOV] Beullens, W. and B. Preneel, "LUOV: Field lifting for smaller UOV public keys", IACR Cryptology ePrint Archive , 2017 , <https://eprint.iacr.org/2017/776>.

[MQDSS] Hulsing, A., Rijneveld, J., Samardjiska, S., and P. Schwabe, "From 5-pass MQ-based identification to MQ-based signatures", IACR Cryptology ePrint Archive , 2016 , <https://eprint.iacr.org/2016/708>.

[NIST-OIDS] National Institute of Standards and Technology (NIST), "Computer Security Objects Register", October 2019 , <https://csrc.nist.gov/Projects/Computer-Security-Objects-Register/Algorithm-Registration>.

[NIST-PQ] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography", October 2019 , <https://www.nist.gov/pqcrypto>.

[PICNIC] Chase, M., Derler, D., Goldfeder, S., Orlandi, C., Ramacher, S., Rechberger, C., Slamanig, D., and G. Zaverucha, "Post-quantum zero-knowledge and signatures from symmetric-key primitives", DOI 10.1145/3133956.3133997, ACM SIGSAC Conference on Computer and Communications Security pp. 1825--1842, 2017 , <https://doi.org/10.1145/3133956.3133997>.

[PQ-X.509] C. F. Pereira, G.C. and B. Neill, "To be specified", RFC XYZ, October 2019 , <https://www.rfc-editor.org/info/rfcXYZ>.

[qTesla] Alkim, E., L.M. Barreto, P.S., Bindel, N., Longa, P., and J. E. Ricardini, "The Lattice-Based Digital Signature Scheme qTESLA", IACR Cryptology ePrint Archive , October 2019 , <https://eprint.iacr.org/2019/085>.

[RAINBOW] Ding, J. and D. Schmidt, "Rainbow, a New Multivariable Polynomial Signature Scheme", 2005 , <https://link.springer.com/chapter/10.1007/11496137_12>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997 , <https://www.rfc-editor.org/info/rfc2119>.

[RFC3279] Bassham, L., Polk, W., and R. Housley, "Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 3279, April 2002 , <https://www.rfc-editor.org/info/rfc3279>.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, DOI 10.17487/RFC5280, May 2008 , <https://www.rfc-editor.org/info/rfc5280>.

[RFC5758] Dang, Q., Santesson, S., Moriarty, K., Brown, D., and T. Polk, "Internet X.509 Public Key Infrastructure: Additional Algorithms and Identifiers for DSA and ECDSA", RFC 5758, January 2010 , <https://www.rfc-editor.org/info/rfc5758>.

[SPHINCS-PLUS] Bernstein, D.J., Dobraunig, C., Eichlseder, M., Fluhrer, S., Gazdag, S-L., Hulsing, A., Kampanakis, P., Kolbl, S., Lange, T., Lauridsen, M., Mendel, F., Niederhagen, R., Rechberger, C., Rijneveld, J., and P. Schwabe, "SPHINCS+", 2017 , <https://sphincs.org/>.

Author's Address

Geovandro C. C. F. Pereira
evolutionQ Inc. / IQC, University of Waterloo
Email: geovandro.pereira@evolutionq.com