Workgroup: LAMPS WG
Internet-Draft:
draft-ietf-massimo-lamps-pq-pkix-00
Published: 8 July 2022
Intended Status: Standards Track
Expires: 9 January 2023
Authors: J. Massimo AWS
P. Kampanakis AWS
S. Turner sn3rd
B. Westerbaan Cloudflare

Algorithms and Identifiers for Post-Quantum Algorithms in the Internet X.509 Public Key Infrastructure

Abstract

Digital signatures are used within X.509 certificates, Certificate Revocation Lists (CRLs), and to sign messages. This document describes the conventions for using Dilithium quantum-resistant signatures in Internet X.509 certificates and certifiate revocation lists. The conventions for the associated post-quantum signatures, subject public keys, and private key are also described.

[EDNOTE: This draft is not expected to be finalized before the NIST PQC Project has standardized PQ algorithms. After NIST has standardized its first algorithms, this document will replace TBD, with the appropriate algorithms and parameters before proceeding to ratification. The algorithm Dilithium has been added as an example in this draft, to provide a more detailed illustration of the content - it by no means indicates its inclusion in the final version. This specification will use object identifiers for the new algorithms that are assigned by NIST, and will use placeholders until these are released.]

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 9 January 2023.

Table of Contents

1. Introduction

The US National Institute of Standards and Technology (NIST) Post-Quantum Cryptography (PQC) effort has defined quantum-resistant public key cryptographic algorithm standards [NIST-PQC]. This document specifies the use of these Post-Quantum public key algorithms with Public Key Infrastructure X.509 (PKIX) certificates and Certificate Revocation Lists (CRLs) using object identifiers algorithms assigned by NIST.

This specification includes conventions for the signatureAlgorithm, signatureValue, signature, and subjectPublicKeyInfo fields within Internet X.509 certificates and CRLs [RFC5280], like [RFC3279] did for classic cryptography and [RFC5480] did for elliptic curve cryptography. It describes the encoding of digital signatures and public keys generated with quantum-resistant signature algorithm Dilithium.

1.1. Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. Identifiers

This specification uses placeholders for object identifiers until the identifiers for the new algorithms are assigned by NIST.

The AlgorithmIdentifier type, which is included herein for convenience, is defined as follows:

```
AlgorithmIdentifier  ::=  SEQUENCE  {
    algorithm   OBJECT IDENTIFIER,
    parameters  ANY DEFINED BY algorithm OPTIONAL
}
```

NOTE: The above syntax is from [RFC5280] and matches the version used therein, i.e., the 1988 ASN.1 syntax. See [RFC5912] for ASN.1 copmatible with the 2015 ASN.1 syntax.

The OIDs are:

```
id-dilithiumTBD OBJECT IDENTIFIER ::= { joint-iso-itu-t(2)
         country(16) us(840) organization(1) gov(101) csor(3)
         nistAlgorithm(4) sigAlgs(3) TBD }
```

The contents of the parameters component for each algorithm are absent.

3. Dilithium Signatures in PKIX

Dilithium is a digital signature scheme built upon the Fiat-Shamir-with-aborts framework [Fiat-Shamir]. The security is based upon the hardness of lattice problems over module lattices [Dilithium]. Dilithium provides three parameter sets for the security categories 2, 3 and 5.

Signatures are used in a number of different ASN.1 structures. As shown in the ASN.1 representation from [RFC5280] below, in an X.509 certificate, a signature is encoded with an algorithm identifier in the signatureAlgorithm attribute and a signatureValue attribute that contains the actual signature.

```
Certificate  ::=  SEQUENCE  {
   tbsCertificate       TBSCertificate,
   signatureAlgorithm   AlgorithmIdentifier,
   signatureValue       BIT STRING  }
```

Signatures are also used in the CRL list ASN.1 representation from [RFC5280] below. In a X.509 CRL, a signature is encoded with an algorithm identifier in the signatureAlgorithm attribute and a signatureValue attribute that contains the actual signature.

```
CertificateList  ::=  SEQUENCE  {
   tbsCertificate       TBSCertList,
   signatureAlgorithm   AlgorithmIdentifier,
   signatureValue       BIT STRING  }
```

The identifiers defined in Section 2 can be used as the AlgorithmIdentifier in the signatureAlgorithm field in the sequence Certificate/CertificateList and the signature field in the sequence TBSCertificate/TBSCertList in certificates CRLs, respectively, [RFC5280]. The parameters of these signature algorithms are absent, as explained in Section 2.

The signatureValue field contains the corresponding Dilithium signature computed upon the ASN.1 DER encoded tbsCertificate [RFC5280].

Conforming Certification Authority (CA) implementations **MUST** specify the algorithms explicitly by using the OIDs specified in Section 2 when encoding Dilithium signatures in certificates and CRLs. Conforming client implementations that process certificates and CRLs using Dilithium **MUST** recognize the corresponding OIDs. Encoding rules for Dilithium signature values are specified Section 2.

When the id-dilithiumTBD identifier appears in the algorithm field as an AlgorithmIdentifier, the encoding **MUST** omit the parameters field. That is, the AlgorithmIdentifier **SHALL** be a SEQUENCE of one component, the OID id-dilithiumTBD.

4. Dilithium Public Keys in PKIX

In the X.509 certificate, the subjectPublicKeyInfo field has the SubjectPublicKeyInfo type, which has the following ASN.1 syntax:

```
SubjectPublicKeyInfo  ::=  SEQUENCE  {
    algorithm         AlgorithmIdentifier,
    subjectPublicKey  BIT STRING
}
```

The public parameters for Dilithium are based upon a polynomial ring R_q for prime q. A (k*l) public matrix A is produced, consisting of polynomials whose coefficients are sampled uniformly at random from the integers modulo q. This sampling is performed by expanding a nonce (rho) using an XOF.

The Dilithium public key **MUST** be encoded using the ASN.1 type DilithiumPublicKey:

```
DilithiumPublicKey ::= OCTET STRING
```

where DilithiumPublicKey is a concatenation of rho and t1. Here, rho is the nonce used to seed the XOF to produce the matrix A, and t1 is a vector encoded in 320*k bytes where k is the rank of the vector over the polynomial ring R_q. These parameters **MUST** be encoded as a single OCTET STRING. The size required to hold a DilithiumPublicKey public key element is therefore 32+320*k bytes.

The id-dilithiumTBD identifier defined in Section 2 **MUST** be used as the algorithm field in the SubjectPublicKeyInfo sequence [RFC5280] to identify a Dilithium public key.

The intended application for the key is indicated in the keyUsage certificate extension; see Section 4.2.1.3 of [RFC5280]. If the keyUsage extension is present in a certificate that indicates id-dilithiumTBD in the SubjectPublicKeyInfo, then the at least one of following **MUST** be present:

digitalSignature; or
nonRepudiation; or
keyCertSign; or
cRLSign.

Requirements about the keyUsage extension bits defined in [RFC5280] still apply.

Conforming CA implementations **MUST** specify the X.509 public key algorithm explicitly by using the OIDs specified in Section 2 when using Dilithium public keys in certificates and CRLs. Conforming client implementations that process Dilithium public keys when processing certificates and CRLs **MUST** recognize the corresponding OIDs.

5. Dilithium Private Keys

A Dilithium private key is encoded as DilithiumPrivateKey in the privateKey field as an OCTET STRING. Dilithium public keys are optionally distributed in the publicKey field of the PrivateKeyInfo structure.

The ASN.1 encoding for a Dilithium private key is as follows:

```
DilithiumPrivateKey ::= SEQUENCE {
    rho         BIT STRING,         - nonce/seed
    K           BIT STRING,         - key/seed
    tr          BIT STRING,         - PRF bytes (CRH in spec.)
    s1          BIT STRING,         - vector l
    s2          BIT STRING,         - vector k
    t0          BIT STRING,         - encoded vector
    PublicKey   IMPLICIT DilithiumPublicKey OPTIONAL
}
```

Dilithium offers both deterministic and randomized signing. The deterministic version creates a signature based on a function of the key K and the message, whereas the randomized version instead selects these values at random. The randomized version can be invoked by leaving K as EMPTY.

A fully populated Dilithium private key consists of 6 parameters. The size necessary to hold all private key elements is 32+32+32+32*[(k+l)*ceiling(log(2*eta+1))+13*k] bytes. The description of k, l, and eta as well as public key and secret key sizes for security levels 2, 3, and 5 can be found in Figure 1 of the Appendix.

6. ASN.1 Module

This section includes the ASN.1 module for Post-Quantum algorithms in X.509. This module does not come from any previously existing RFC. This module references [RFC5912].

```
[ EDNOTE: Add ASN.1 here ]

  PKIX1-PQ-Algorithms { iso(1) identified-organization(3) dod(6)
     internet(1) security(5) mechanisms(5) pkix(7) id-mod(0)
     id-mod-pkix1-PQ-algorithms(X) }

  DEFINITIONS EXPLICIT TAGS ::=

  BEGIN

  -- EXPORTS ALL;

  IMPORTS

  -- FROM RFC 5912

  PUBLIC-KEY, SIGNATURE-ALGORITHM, DIGEST-ALGORITHM, SMIME-CAPS
  FROM AlgorithmInformation-2009
    { iso(1) identified-organization(3) dod(6) internet(1)
      security(5) mechanisms(5) pkix(7) id-mod(0)
      id-mod-algorithmInformation-02(58) }

  --
  -- Public Key (pk-) Algorithms
  --
  PublicKeys PUBLIC-KEY ::= {
    -- This expands PublicKeys from RFC 5912
    pk-dilithiumTBD |
    pk-TBD-TBD,
    ...
  }

  -- The hashAlgorithm is mda-shake256
  -- The XOF seed rho is 32 bytes
  -- The vector t1 is 320*k bytes
  -- These are encoded as a single string
  pk-dilithiumTBD PUBLIC-KEY ::= {
    IDENTIFIER id-dilithiumTBD
    KEY DilithiumPublicKey
    PARAMS ARE absent
    CERT-KEY-USAGE { nonRepudiation, digitalSignature,
                     keyCertSign, cRLSign }
    PRIVATE-KEY DilithiumPrivateKey
  }

  END
```

7. IANA Considerations

Extensions in certificates and CRLs are identified using object Identifiers (OIDs). The creation and delegation of these arcs is to be determined.

IANA is requested to register the id-mod-pkix1-PQ-algorithms OID for the ASN.1 module identifier found in Section 5 in the "SMI Security for PKIX Module Identifier" registry.

8. Security Considerations

The Security Considerations section of [RFC5280] applies to this specification as well.

The digital signature scheme defined within this document are modeled under existentially unforgeable digital signatures with respect to an adaptive chosen message attack (EUF-CMA). For the purpose of estimating security strength, it has been assumed that the attacker has access to signatures for no more than 2^{64} chosen messages.

EDNOTE: Discuss implications of not hash-then-sign. Implications in performance too.

Within the hash-then-sign paradigm, hash functions are used as a domain restrictor over the message to be signed. By pre-hashing, the onus of resistance to existential forgeries becomes heavily reliant on the collision-resistance of the hash function in use. As well as this security goal, the hash-then-sign paradigm also has the ability to improve performance by reducing the size of signed messages. As a corollary, hashing remains mandatory even for short messages and assigns a further computational requirement onto the verifier. This makes the performance of hash-then-sign schemes more consistent, but not necessarily more efficient. Dilithium diverges from the hash-then-sign paradigm by hashing the message during the signing procedure (at the point in which the challenge polynomial). However, due to the fact that Dilithium signatures may require the signing procedure to be repeated several times for a signature to be produced, Dilithium implementations can make use of pre-hashing the message to prevent rehashing with each attempt.

EDNOTE: Discuss side-channels for Dilithium. .

Dilithium has been designed to provide side-channel resilience by eliminating a reliance on Gaussian sampling. While deliberate design decisions such as these can help to deliver a greater ease of secure implementation - particularly against side-channel attacks - it does not necessarily provide resistance to more powerful attacks such as differential power analysis. Some amount of side-channel leakage has been demonstrated in parts of the signing algorithm (specifically the bit-unpacking function), from which a demonstration of key recovery has been made over a large sample of signatures. Masking countermeasures exist for Dilithium, but come with a performance overhead.

A fundamental security property also associated with digital signatures is non-repudiation. Non-repudiation refers to the assurance that the owner of a signature key pair that was capable of

generating an existing signature corresponding to certain data cannot convincingly deny having signed the data. The digital signature scheme Dilithium possess three security properties beyond unforgeability, that are associated with non-repudiation. These are exclusive ownership, message-bound signatures, and non-resignability. These properties are based tightly on the assumed collision resistance of the hash function used (in this case SHAKE-256). Exclusive ownership is a property in which a signature sigma uniquely determines the public key and message for which it is valid. Message-bound signatures is the property that a valid signature uniquely determines the message for which it is valid, but not necessarily the public key. Non-resignability is the property in which one cannot produce a valid signature under another key given a signature sigma for some unknown message m. These properties are not provided by classical signature schemes such as DSA or ECDSA, and have led to a variety of attacks such as Duplicate-Signature Key Selection (DSKS) attacks , and attacks on the protocols for secure routing. A full discussion of these properties in Dilithium can be found at [CDFFJ21]. These properties are dependent, in part, on unambiguous public key serialization. It for this reason the public key structure defined in Section 4 is intentionally encoded as a single OCTET STRING.

9. References

9.1. Normative References

[NIST-PQC] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography", 2016, <https://csrc.nist.gov/Projects/post-quantum-cryptography/post-quantum-cryptography-standardization/Call-for-Proposals>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

[RFC5912] Hoffman, P. and J. Schaad, "New ASN.1 Modules for the Public Key Infrastructure Using X.509 (PKIX)", RFC 5912, DOI 10.17487/RFC5912, June 2010, <https://www.rfc-editor.org/info/rfc5912>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

9.2. Informative References

[CDFFJ21]
Cremers, Cas., Düzlü, S., Fiedler, R., Fischlin, M., and C. Janson, "BUFFing signature schemes beyond unforgeability and the case of post-quantum signatures", In Proceedings of the 42nd IEEE Symposium on Security and Privacy, 2021, <https://eprint.iacr.org/2020/1525.pdf>.

[Dilithium] Bai, S., Ducas, L., Lepoint, T., Lyubashevsky, V., Schwabe, P., Seiler, G., and D. Stehlé, "CRYSTALS-Dilithium Algorithm Specifications and Supporting Documentation", 2021, <https://pq-crystals.org/dilithium/data/dilithium-specification-round3-20210208.pdf>.

[Fiat-Shamir] Lyubashevsky, V., "Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures", International Conference on the Theory and Application of Cryptology and Information Security, 2009, <https://www.iacr.org/archive/asiacrypt2009/59120596/59120596.pdf>.

[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, DOI 10.17487/RFC3279, April 2002, <https://www.rfc-editor.org/info/rfc3279>.

[RFC5480] Turner, S., Brown, D., Yiu, K., Housley, R., and T. Polk, "Elliptic Curve Cryptography Subject Public Key Information", RFC 5480, DOI 10.17487/RFC5480, March 2009, <https://www.rfc-editor.org/info/rfc5480>.

Appendix A. Acknowledgements

We would like to thank ... for their insightful comments.

Appendix B. Appendix

Instead of defining the strength of a quantum algorithm in a traditional manner using precise estimates of the number of bits of security, NIST has instead elected to define a collection of broad security strength categories. Each category is defined by a comparatively easy-to-analyze reference primitive that cover a range of security strengths offered by existing NIST standards in symmetric cryptography, which NIST expects to offer significant resistance to quantum cryptanalysis. These categories describe any attack that breaks the relevant security definition that must require computational resources comparable to or greater than those required for: Level 1 - key search on a block cipher with a 128-bit key (e.g., AES128), Level 2 - collision search on a 256-bit hash function (e.g., SHA256/ SHA3-256), Level 3 - key search on a block cipher with a 192-bit key (e.g., AES192), Level 4 - collision search on a 384-bit hash function (e.g. SHA384/ SHA3-384), Level 5 - key search on a block cipher with a 256-bit key (e.g., AES 256).

The parameter sets defined for NIST security levels 2, 3 and 5 are listed in the Figure 1, along with the resulting public key and private key sizes in bytes.

Security Level	n	q	(k,l)	eta	gamma1	Public Key(B)	Private Key(B)
2	256	8380417	(4,4)	2	2^17	1312	2528
3	256	8380417	(6,5)	4	2^19	1952	4000
5	256	8380417	(8,7)	2	2^19	2596	4864

Figure 1

Authors' Addresses

Jake Massimo
AWS
United States of America

Email: jakemas@amazon.com

Panos Kampanakis
AWS
United States of America

Email: kpanos@amazon.com

Sean Turner
sn3rd

Email: sean@ssn3rd.com

Bas Westerbaan
Cloudflare

Email: bas@westerbaan.name