Internet X.509 Public Key Infrastructure - Algorithm Identifiers for Kyber

Abstract

Kyber is a key-encapsulation mechanism (KEM). This document specifies algorithm identifiers and ASN.1 encoding format for Kyber in public key certificates. The encoding for public and private keys are also provided.

\[EDNOTE: This document is not expected to be finalized before the NIST PQC Project has standardized PQ algorithms. This specification will use object identifiers for the new algorithms that are assigned by NIST, and will use placeholders until these are released.

About This Document

This note is to be removed before publishing as an RFC.

The latest revision of this draft can be found at https://lamps-wg.github.io/kyber-certificates/#go.draft-ietf-lamps-kyber-certificates.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-lamps-kyber-certificates/.


Source for this draft and an issue tracker can be found at https://github.com/lamps-wg/kyber-certificates.

Status of This Memo

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

Kyber is a key-encapsulation mechanism (KEM) standardized by the US NIST PQC Project [PQCProj]. This document specifies the use of the
Kyber algorithm at three security levels: Kyber512, Kyber768, and Kyber1024, in X.509 public key certificates; see [RFC5280]. Public and private key encodings are also specified.

1.1. ASN.1 and Kyber Identifiers

An ASN.1 module [X680] is included for reference purposes. Note that as per [RFC5280], certificates use the Distinguished Encoding Rules; see [X690]. Also note that NIST defined the object identifiers for the Kyber algorithms in an ASN.1 modulle; see (TODO insert reference).

1.2. Applicability Statement

Kyber certificates are used in protocols where the public key is used to generate and encapsulate a shared secret used to derive a symmetric key used to encrypt a payload; see [I-D.ietf-lamps-kyber]. To be used in TLS, Kyber certificates could only be used as end-entity identity certificates and would require significant updates to the protocol; see [I-D.celi-wiggers-tls-authkem].

2. Conventions and Definitions

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.

3. Algorithm Identifiers

Certificates conforming to [RFC5280] can convey a public key for any public key algorithm. The certificate indicates the algorithm through an algorithm identifier. An algorithm identifier consists of an object identifier and optional parameters.

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

AlgorithmIdentifier{ALGORITHM-TYPE, ALGORITHM-TYPE:AlgorithmSet} ::=  
SEQUENCE {  
    algorithm ALGORITHM-TYPE.&id({AlgorithmSet}),  
    parameters ALGORITHM-TYPE.  
    &Params({AlgorithmSet}{@algorithm}) OPTIONAL  
}

: The above syntax is from [RFC5912] and is compatible with the 2021 ASN.1 syntax [X680].
The fields in AlgorithmIdentifier have the following meanings:

*algorithm identifies the cryptographic algorithm with an object identifier.

*parameters, which are optional, are the associated parameters for the algorithm identifier in the algorithm field.

Section 4 includes object identifiers for Kyber-512, Kyber-768, and Kyber-1024. For all of these OIDs, the parameters MUST be absent.

It is possible to find systems that require the parameters to be present. This can be due to either a defect in the original 1997 syntax or a programming error where developers never got input where this was not true. The optimal solution is to fix these systems; where this is not possible, the problem needs to be restricted to that subsystem and not propagated to the Internet.

4. Kyber Public Key Identifiers

The AlgorithmIdentifier for a Kyber public key MUST use one of the id-alg-kyber object identifiers listed below, based on the security level. The parameters field of the AlgorithmIdentifier for the Kyber public key MUST be absent.

When any of the Kyber AlgorithmIdentifier appears in the SubjectPublicKeyInfo field of an X.509 certificate, the key usage certificate extension MUST only contain keyEncipherment Section 4.2.1.3 of [RFC5280].
As noted in Section 3, the values for these object identifiers will be assigned by NIST. Once assigned, they will be added to a future revision of this document.

5. Subject Public Key Fields

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

SubjectPublicKeyInfo {PUBLIC-KEY: IOSet} ::= SEQUENCE {
  algorithm AlgorithmIdentifier {PUBLIC-KEY, {IOSet}},
  subjectPublicKey BIT STRING
}

: The above syntax is from [RFC5912] and is compatible with the 2021 ASN.1 syntax [X680].

The fields in SubjectPublicKeyInfo have the following meaning:

*algorithm is the algorithm identifier and parameters for the public key (see above).
subjectPublicKey contains the byte stream of the public key. The algorithms defined in this document always encode the public key as TODO pick format e.g., exact multiple of 8 bits?.

The following is an example of a Kyber-512 public key encoded using the textual encoding defined in [RFC7468]:

-----BEGIN PUBLIC KEY-----
TODO insert example public key
-----END PUBLIC KEY------

6. Private Key Format

"Asymmetric Key Packages" [RFC5958] describes how to encode a private key in a structure that both identifies what algorithm the private key is for and allows for the public key and additional attributes about the key to be included as well. For illustration, the ASN.1 structure OneAsymmetricKey is replicated below. The algorithm-specific details of how a private key is encoded are left for the document describing the algorithm itself.

OneAsymmetricKey ::= SEQUENCE {
  version                  Version,
  privateKeyAlgorithm      SEQUENCE {
    algorithm                PUBLIC-KEY.&id({PublicKeySet}),
    parameters               PUBLIC-KEY.&Params({PublicKeySet}
                               {@privateKeyAlgorithm.algorithm})
                              OPTIONAL
  }
  privateKey               OCTET STRING (CONTAINING
                               PUBLIC-KEY.&PrivateKey({PublicKeySet}
                               {@privateKeyAlgorithm.algorithm}),
                               attributes           [0] Attributes OPTIONAL,
    ...,
    [[2: publicKey       [1] BIT STRING (CONTAINING
       PUBLIC-KEY.&Params({PublicKeySet}
       {@privateKeyAlgorithm.algorithm})
       OPTIONAL,
       ...]
  }

PrivateKey ::= OCTET STRING

Publickey ::= BIT STRING

: The above syntax is from [RFC5958] and is compatible with the 2021 ASN.1 syntax [X680].

For the keys defined in this document, the private key is always an opaque byte sequence. The ASN.1 type PqckemPrivateKey is defined in this document to hold the byte sequence. Thus, when encoding a
OneAsymmetricKey object, the private key is wrapped in a PqckemPrivateKey object and wrapped by the OCTET STRING of the "privateKey" field.

PqckemPrivateKey ::= OCTET STRING

The following is an example of a Kyber-512 private key encoded using the textual encoding defined in [RFC7468]:

-----BEGIN PRIVATE KEY-----
TODO inser example private key
-----END PRIVATE KEY-------

The following example, in addition to encoding the Kyber-512 private key, has an attribute included as well as the public key. As with the prior example, the textual encoding defined in [RFC7468] is used:

-----BEGIN PRIVATE KEY-----
TODO inser example private key with attribute
-----END PRIVATE KEY-------

: There exist some private key import functions that have not implemented the new ASN.1 structure OneAsymmetricKey that is defined in [RFC5958]. This means that they will not accept a private key structure that contains the public key field. This means a balancing act needs to be done between being able to do a consistency check on the key pair and widest ability to import the key.

7. ASN.1 Module

TODO ASN.1 Module

8. Security Considerations

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

[EDNOTE: Discuss side-channels for Kyber TBD1.]

9. IANA Considerations

This document will have some IANA actions.

10. References

10.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/


10.2. Informative References


[National Institute of Standards and Technology, "Post-Quantum Cryptography Project", 20 December 2016,

Acknowledgments

TODO acknowledge.

Authors' Addresses

Sean Turner
sn3rd

Email: sean@sn3rd.com

Panos Kampanakis
AWS

Email: kpanos@amazon.com

Jake Massimo
AWS

Email: jakemas@amazon.com

Bas Westerbaan
Cloudflare

Email: bas@westerbaan.name