CRYPTONEXT
SECURITY

THE NEW GENERATION OF
QUANTUM RESISTANT AND SOVEREIGN
CRYPTOGRAPHY

# Use of KYBER in the Cryptographic Message Syntax (CMS)

IETF LAMPS 115

*draft-ietf-lamps-cms-kyber*

Julien Prat (CryptoNext Security)
Mike Ounsworth (Entrust)

NOVEMBER 2022

# CHANGES BETWEEN DRAFT-PERRET-PRAT-LAMPS-CMS-PQ-KEM-01 AND DRAFT-IETF-LAMPS-CMS-KYBER-00

**Editorial:**

- Editorial comments from Carl Wallace and Michael Richardson were taken into account

**RecipientInfo Conventions:**

- Use of KeyTransRecipientInfo to communicate algorithm info
- KeyTransRecipientInfo value MUST have the following values:
  - keyEncryptionAlgorithm.algorithm MUST be *id-kem-trans* OID (KEM-TRANS mechanism)
  - keyEncryptionAlgorithm.parameters MUST be a value of type *GenericKemTransParameters*
  - encryptedKey MUST be the encrypted keying data (*EK*) output by the KEM-TRANS Mechanism

**Algorithm limitations:**

- Algorithms to be used in KEM-TRANS are limited to Kyber:

| Security Level | KEM | KDF | WRAP |
|---|---|---|---|
| 128 bits | KYBER512 | HKDF-SHA256 | AES128-WRAP |
| 192 bits | KYBER768 | HKDF-SHA384 | AES192-WRAP |
| 256 bits | KYBER1024 | HKDF-SHA512 or NULL | AES256-WRAP |

# OPEN POINTS TO BE DISCUSSED

**Certificate Conventions:**

Depend on the work item PQC-PKIX

**New OIDs to be defined:**

- id-kem-trans (KEM-TRANS mechanism)
- id-kyber512, id-kyber768, id-kyber1024 (KYBER algorithms)

**Should we limit the number of algorithm combination to 3, depending on the security level consistency?**

- 128 bits
- 192 bits
- 256 bits

CRYPTONEXT SECURITY

# Thank you !

Julien PRAT
julien.prat@cryptonext-security.com

contact@cryptonext-security.com
www.cryptonext-security.com
https://www.linkedin.com/company/cryptonext-security

THE NEW GENERATION OF
QUANTUM RESISTANT AND SOVEREIGN
CRYPTOGRAPHY

# Back Up Slides

CRYPTONEXT SECURITY

THE NEW GENERATION OF
QUANTUM RESISTANT AND SOVEREIGN
CRYPTOGRAPHY

# DESIGN RATIONALES

**RFC Purpose:**
Define how to use Kyber within the Cryptographic Message Syntax (CMS)

**CMS Context:**
One of the typical use case of the CMS Envelopped-Data Content is to:

1. randomly generate a CEK,
2. encrypt the data with a symmetric algorithm using this CEK
3. individually send the CEK to one or more recipients protected by asymmetric cryptography in a RecipientInfo object.

**Requirements:**
Need to define a new Key Transport mechanism fulfilling the following requirements:

- the Key Transport Mechanism SHALL be secure against quantum computers.
- the Key Transport Mechanism SHALL be able to take the Content-Encryption Key (CEK) as input.

=> Definition of the **KEM-TRANS mechanism**

# Key Encapsulation Mechanism – Definition

A key encapsulation mechanism (KEM) is an asymmetric cryptographic algorithm allowing secret sharing between two entities.

KEM consisting of 3 functions:

- Key generation **KeyGen**() :
  - Returns a public key and a private key (PK, SK)
- Encapsulation **Encaps**(PK):
  - Takes as input the public key
  - Returns a ciphertext CT and a shared secret SS
- Decapsulation **Decaps**(SK, CT):
  - Takes as input the private key and the ciphertext
  - Returns the shared secret SS

=> Impossible to encrypt a fixed CEK with KEM

# KEY DERIVATION FUNCTION – DEFINITION

A key derivation function (KDF) is a cryptographic algorithm that derives one or more secret keys from a secret value using a pseudorandom function.

KDF consists of 1 function:

- Key Derivation **Derive**(SS, KEK_LEN) :
    - Takes as input a shared secret SS and the length of the output secret key KEK_LEN
    - Returns a secret key KEK

# Wrapping Algorithm – Definition

A wrapping algorithm (WRAP) is a symmetric cryptographic algorithm protecting data in confidentiality and in integrity.

WRAP consists of 2 functions:

- Wrapping **Wrap**(KEK, K) :
  - Takes as input a wrapping key KEK and a plaintext key K
  - Returns a wrapped key WK
- Unwrapping **Unwrap**(KEK, WK):
  - Takes as input a wrapping key KEK and a wrapped key WK
  - Returns the plaintext key K

# KEM-TRANS MECHANISM - DESCRIPTION

**Assumptions:**

Sender has been provided with :

- *recipPubKey*: the recipient's public key for KEM.
- ***K***: the keying data to be transported, length is compatible with the chosen WRAP algorithm.

**Sender's operations:**

1. (SS, CT) = KEM.encaps(recipPubKey)
2. KEK = KDF.derive(SS, kekLen)
3. WK = WRAP.wrap(KEK, **K**)
4. EK = (WK || CT)

**Recipient's operations:**

1. (WK || CT) = EK
2. SS = KEM.decaps(recipPrivKey, CT)
3. KEK = KDF.derive(SS, kekLen)
4. **K** = WRAP.Unwrap(KEK, WK)

**=> KEM-TRANS mechanism allows the transport of any keying data, including CMS CEK**

**=> KEM-TRANS mechanism can be instantiated with any KEM algorithm, including a Quantum-Safe KEM, making the KEM-TRANS mechanism Quantum-Safe**

# KEM-TRANS MECHANISM - USE IN CMS

**RecipientInfo Conventions:**

- RecipientInfo Type MUST be KeyTransRecipientInfo
- KeyTransRecipientInfo value MUST have the following values:
  - keyEncryptionAlgorithm.algorithm MUST be id-kem-trans OID (KEM-TRANS mechanism)
  - keyEncryptionAlgorithm.parameters MUST be a value of type GenericKemTransParameters
  - encryptedKey MUST be the encrypted keying data (*EK*) output by the KEM-TRANS Mechanism

**Certificate Conventions:**

- Key Usage Extension MUST contain only the value *keyEncipherment*
- Subject Public Key Info MUST be set to *id-alg-xxx-kem* OID (KEM algorithm)

CRYPTONEXT
SECURITY

# Thank you !

Julien PRAT
julien.prat@cryptonext-security.com

contact@cryptonext-security.com
www.cryptonext-security.com
https://www.linkedin.com/company/cryptonext-security

THE NEW GENERATION OF
QUANTUM RESISTANT AND SOVEREIGN
CRYPTOGRAPHY