Workgroup: IPSECME
Internet-Draft: draft-kampanakis-ml-kem-ikev2
Published: 19 February 2024
Intended Status: Standards Track
Expires: 22 August 2024
Authors: P. Kampanakis, Amazon Web Services
G. Ravago, Amazon Web Services

Post-quantum Hybrid Key Exchange with ML-KEM in the Internet Key Exchange Protocol Version 2 (IKEv2)

Abstract

[EDNOTE: The intention of this draft is to get IANA KE codepoints for ML-KEM. It could be a standards track draft given that ML-KEM will see a lot of adoption, an AD sponsored draft, or even an individual stable draft which gets codepoints from Expert Review. The approach is to be decided by the IPSECME WG.]

NIST recently standardized ML-KEM, a new key encapsulation mechanism, which can be used for quantum-resistant key establishment. This draft specifies how to use ML-KEM as an additional key exchange in IKEv2 along with traditional key exchanges. This Post-Quantum Traditional Hybrid Key Encapsulation Mechanism approach allows for negotiating IKE and Child SA keys which are safe against cryptanalytically-relevant quantum computers and theoretical weaknesses in ML-KEM.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

A Cryptanalytically-relevant Quantum Computer (CRQC), if it became a reality, could threaten public key encryption algorithms used today for key exchange. Someone storing encrypted communications which use (Elliptic Curve) Diffie-Hellman ((EC)DH) to negotiate keys could decrypt these communications in the future after a CRQC was available. This includes Internet Key Exchange Protocol Version 2 (IKEv2, which the security is based on using the (EC)DH key exchange in the IKE_SA_INIT messages.

To address this concern, [RFC8784] introduced Post-quantum Preshared Keys as a temporary option for stirring a pre-shared key of adequate entropy in the derived Child SA encryption keys in order to provide quantum-resistance. Since then, [RFC9242] defined how to do additional large message exchanges by using new IKE_INTERMEDIATE or IKE_FOLLOWUP_KE messages. As post-quantum keys are usually larger than common network Maximum Transport Units (MTU), IKE_INTERMEDIATE messages can be fragmented which could allow for the peers to do post-quantum key exchanges without IP fragmentation. [RFC9370] defined how to do up to seven additional key exchanges by using

IKE_INTERMEDIATE or IKE_FOLLOWUP_KE messages and deriving new SKEYSEED and KEYMAT key materials. This allows for new post-quantum key exchanges to be used in the derived IKE and Child SA keys and provide quantum resistance.

NIST has been working on a public project [NIST-PQ] for standardizing quantum-resistant algorithms which include key encapsulation and signatures. At the end of Round 3, they picked Kyber as the first Key Encapsulation Mechanism (KEM) for standardization [I-D.draft-cfrg-schwabe-kyber-04]. Kyber was then standardized as Module-Lattice-based Key-Encapsulation Mechanism (ML-KEM) in [FIPS203-ipd]. ML-KEM was standardized in 2024 [FIPS203]. [EDNOTE: Reference normatively the ratified version [I-D.draft-cfrg-schwabe-kyber-04] if it is ever ratified. Otherwise keep a normative reference of [FIPS203]. And remove the reference to [FIPS203-ipd].]

This document describes how ML-KEM can be used as the quantum-resistant KEM in IKEv2 by using one additional IKE_INTERMEDIATE or IKE_FOLLOWUP_KE key exchange after an initial key exchange in IKE_SA_INIT or CREATE_CHILD_SA respectively. This approach of combining a quantum-resistant with a classical algorithm, is commonly called Post-Quantum Traditional (PQ/T) Hybrid [I-D.ietf-pquip-pqt-hybrid-terminology-02] key exchange and combines the security of a well-established algorithm with relatively new quantum-resistant algorithms which could theoretically have unknown issues. The result is a new Child SA key or an IKE or Child SA rekey with keying material which is safe against a CRQC. This specification is a profile of [RFC9370] and registers new algorithm identifiers for ML-KEM key exchanges in IKEv2.

1.1. KEMs

In the context of the NIST Post-Quantum Cryptography Standardization Project [NIST-PQ], key exchange algorithms are formulated as KEMs, which consist of three steps:

* 'KeyGen() -> (pk, sk)': A probabilistic key generation algorithm, which generates a public key 'pk' and a secret key 'sk'.

* 'Encaps(pk) -> (ct, ss)': A probabilistic encapsulation algorithm, which takes as input a public key 'pk' and outputs a ciphertext 'ct' and shared secret 'ss'.

* 'Decaps(sk, ct) -> ss': A decapsulation algorithm, which takes as input a secret key 'sk' and ciphertext 'ct' and outputs a shared secret 'ss', or in some cases a distinguished error value.

The main security property for KEMs standardized by NIST is indistinguishability under adaptive chosen ciphertext attacks (IND-

CCA2), which means that shared secret values should be indistinguishable from random strings even given the ability to have arbitrary ciphertexts decapsulated. IND-CCA2 corresponds to security against an active attacker, and the public key / secret key pair can be treated as a long-term key or reused. A weaker security notion is indistinguishability under chosen plaintext attacks (IND-CPA), which means that the shared secret values should be indistinguishable from random strings given a copy of the public key. IND-CPA roughly corresponds to security against a passive attacker, and sometimes corresponds to one-time key exchange.

1.2. ML-KEM

ML-KEM is a standardized lattice-based key encapsulation mechanism [FIPS203]. [EDNOTE: Reference normatively the ratified version [I-D.draft-cfrg-schwabe-kyber-04] if it is ever ratified. Otherwise keep a normative reference of [FIPS203].]

ML-KEM is using Module Learning with Errors as its underlying primitive which is a structured lattices variant that offers good performance and relatively small and balanced key and ciphertext sizes. ML-KEM was standardized with three parameters, ML-KEM-512, ML-KEM-768, and ML-KEM-1024. These were mapped by NIST to the three security levels defined in the NIST PQC Project, Level 1, 3, and 5. These levels correspond to the hardness of breaking AES-128, AES-192 and AES-256 respectively.

This specification introduces ML-KEM-768 and ML-KEM-1024 to IKEv2 key exchanges as conservative security level parameters which will not have material performance impact on IKEv2/IPsec tunnels which usually stay up for long periods of time. Since the ML-KEM-768 and ML-KEM-1024 public key and ciphertext sizes can exceed the typical network MTU, these key exchanges could require two or three network IP packets from both the initiator and the responder. [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

1.3. 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.

2. ML-KEM in IKEv2

2.1. ML-KEM in IKE_INTERMEDIATE or CREATE_CHILD_SA messages

ML-KEM key exchanges can be negotiated in IKE_INTERMEDIATE or IKE_FOLLOWUP_KE messages as defined in [RFC9370]. We summarize them here for completeness.

Section 2.2.2 of [RFC9370] specifies that KEi(0), KEr(0) are regular key exchange messages in the first IKE_SA_INIT exchange which end up generating a set of keying material, SK_d, SK_a[i/r], and SK_e[i/r]. The peers then perform an IKE_INTERMEDIATE exchange, carrying new Key Exchange payloads. These are protected with the SK_e[i/r] and SK_a[i/r] keys which were derived from the IKE_SA_INIT as per Section 3.3.1 of [RFC9242]. KEi(1) and KEr(1) are the subsequent key exchange messages which carry the ML-KEM public key of a keypair (sk, pk) generated by the initiator with ML-KEM KeyGen() and the 256-bit ML-KEM shared secret SK(1) encapsulated by the responder to a ciphertext ct by using Encaps(pk) respectively. The public key and the ciphertext are encoded as raw bytes in little-endian encoding. [EDNOTE: Confirm this makes sense.] The initiator then decapsulates the 256-bit ML-KEM shared secret SK(1) from the ciphertext ct by using its private key sk in Decaps(sk, ct). Both peers have now reached a common SK(1) at the end of this KE(1) key exchange. The ML-KEM shared secret is stirred into new keying material SK_d, SK_a[i/r], and SK_e[i/r] as defined in Section 2.2.2 of [RFC9370]. Afterwards the peers continue to the IKE_AUTH exchange phase as defined in Section 3.3.2 of [RFC9242].

ML-KEM can also be used to create or rekey a Child SA or rekey the IKE SA by using a IKE_FOLLOWUP_KE message after a CREATE_CHILD_SA message. After the ML-KEM additional key exchange KE(1) has taken place using and IKE_FOLLOWUP_KE exchange, the IKE or Child SA are rekeyed by stirring the new ML-KEM shared secret SK(1) in SKEYSEED and KEYMAT as specified in Section 2.2.4 of [RFC9370].

ML-KEM-768 and ML-KEM-1024 public keys and ciphertexts can exceed typical network MTUs (1500 bytes). Thus, IKE_INTERMEDIATE messages carrying ML-KEM public keys and ciphertexts may be IKEv2 fragmented as per [RFC7383]. IKE_FOLLOWUP_KE messages carrying ML-KEM public keys and ciphertexts cannot be IKEv2 fragmented. Thus, ML-KEM-1024 Key Exchange Method identifier TBD37 **SHOULD** only be used in IKE_INTERMEDIATE exchanges. It **SHOULD NOT** be used in IKE_FOLLOWUP_KE messages until there is a separate document which defines how such exchanges are split in several messages. [EDNOTE: Confirm ML-KEM-768 fits the MTU with captures, otherwise recommend against ML-KE-768 in IKE_FOLLOWUP_KE as well.] [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

Although, this document focuses on using ML-KEM as the second key exchange in a PQ/T Hybrid KEM [I-D.ietf-pquip-pqt-hybrid-terminology-02] scenario, ML-KEM-768 Key Exchange Method identifier TBD36 **MAY** be used in IKE_SA_INIT as a quantum-resistant-only key exchange because the encapsulation key is 1184 bytes, which assuming an additional 250 bytes of IKE header data, can fit in typical network MTUs of 1500 bytes. [EDNOTE: Confirm it fits the MTU with captures.] ML-KEM-1024 Key Exchange Method identifier TBD37 **SHOULD NOT** be used in IKE_SA_INIT messages which could exceed typical network MTUs and cannot be IKEv2 fragmented. [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

2.2. Key Exchange Payload

HDR, the IKE header, of the IKE_INTERMEDIATE messages carrying the ML-KEM key exchange has a Next Payload value of 34 (Key Exchange), Exchange Type of 43 (IKE_INTERMEDIATE) and Message ID of 1 assuming this is the first additional key exchange (ADDKE1). For IKE_FOLLOWUP_KE messages carrying the ML-KEM key exchange, the Exchange Type would be 44 (IKE_FOLLOWUP_KE).

The IKE_INTERMEDIATE or IKE_FOLLOWUP_KE payload is shown below as defined in Section 3.4 of [RFC7296]:

```
                     1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| Next Payload  |C|  RESERVED   |         Payload Length        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Key Exchange Method Num     |           RESERVED            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
~                       Key Exchange Data                       ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

* Payload Length: The ML-KEM-768 public key is 1184 bytes, so the Payload Length field from the initiator would be 1192. The ML-KEM-768 ciphertext is 1088 bytes, so the Payload Length from the responder would be 1096. The ML-KEM-1024 public key is 1568 bytes, so the Payload Length field from the initiator would be 1576. The ML-KEM-1024 ciphertext is 1568 bytes, so the Payload Length from the responder would be 1576. [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

* The Key Exchange Method Num identifier is TBD36 for ML-KEM-768 or TBD37 for ML-KEM-1024. [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

* The Key Exchange Data is the 1184 or 1568 octets of the ML-KEM-768 or ML-KEM-1024 public key respectively for the message from the initiator. The response from the responder is 1088 or 1568 octets as the size of the ML-KEM-768 or ML-KEM-1024 ciphertexts respectively. [EDNOTE: Consider adding ML-KEM-512 which would fit in one packet.]

2.3. Recipient Tests

Receiving and handling of malformed ML-KEM public key or ciphertext **MUST** follow the input validation described in [FIPS203]. [EDNOTE: Reference normatively the ratified version [I-D.draft-cfrg-schwabe-kyber-04] if it is ever ratified. Otherwise keep a normative reference of [FIPS203].] In particular, entities receiving the ML-KEM public key to encapsulate to **MUST** perform the type and modulus checks in Sections 6.1 of [FIPS203] and reject the ML-KEM public key, if malformed. Entities receiving an ML-KEM ciphertext for decapsulation **MUST** perform the ciphertext and decapsulation key type checks in Section 6.2 of [FIPS203] and reject the ciphertext or key, if malformed. [EDNOTE: Reference normatively the ratified version [I-D.draft-cfrg-schwabe-kyber-04] if it is ever ratified. Otherwise keep a normative reference of [FIPS203].] These checks could be performed separately before performing the encapsulation or decapsulation steps or be part of them.

Note that during decapsulation, ML-KEM uses implicit rejection which leads the decapsulating entity to implicitly reject the decapsulated shared secret by setting it to a hash of the ciphertext together with a random value stored in the ML-KEM secret when the re-encrypted shared secret does not match the original one. [EDNOTE: Confirm implicit rejection is still used after [FIPS203] is ratified or change this paragraph.]

3. Security Considerations

All security considerations from [RFC9242] and [RFC9370] apply to the ML-KEM exchanges described in this specification.

The ML-KEM public key generated by the initiator and the ciphertext generated by the responder use randomness (usually a seed) which must be independent of any other random seed used in the IKEv2 negotiation. For example, at the initiator, the ML-KEM and (EC)DH keypairs used in a PQ/T Hybrid key exchange should not be generated from the same seed.

Although ML-KEM is IND-CCA2 secure, reusing the same ML-KEM keypair does not offer forward secrecy. The initiator should generate a new ML-KEM keypair with every ML-KEM key exchange.

4. IANA Considerations

IANA is requested to assign two values for the names "mlkem-768" and "mlkem-1024" in the IKEv2 "Transform Type 4 - Key Exchange Method Transform IDs" and has listed this document as the reference. The Recipient Tests field should also point to this document:

Number	Name	Status	Recipient Tests	Reference
TBD35	mlkem-512		[TBD, this draft, Section 2.3],	[TBD, this draft] [EDNOTE: Consider adding ML-KEM-512.]
TBD36	mlkem-768		[TBD, this draft, Section 2.3],	[TBD, this draft]
TBD37	mlkem-1024		[TBD, this draft, Section 2.3],	[TBD, this draft]
37-1023	Unassigned			

Table 1: Updates to the IANA "Transform Type 4 - Key Exchange Method Transform IDs" table

5. References

5.1. Normative References

[FIPS203] "*** BROKEN REFERENCE ***".

[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/rfc/rfc2119>.

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/rfc/rfc7296>.

[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/rfc/rfc8174>.

[RFC9242] Smyslov, V., "Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 9242, DOI 10.17487/RFC9242, May 2022, <https://www.rfc-editor.org/rfc/rfc9242>.

[RFC9370] Tjhai, CJ., Tomlinson, M., Bartlett, G., Fluhrer, S., Van Geest, D., Garcia-Morchon, O., and V. Smyslov, "Multiple Key Exchanges in the Internet Key Exchange Protocol

Version 2 (IKEv2)", RFC 9370, DOI 10.17487/RFC9370, May 2023, <https://www.rfc-editor.org/rfc/rfc9370>.

5.2. Informative References

[FIPS203-ipd] National Institute of Standards and Technology (NIST), "Module-Lattice-based Key-Encapsulation Mechanism Standard", NIST Federal Information Processing Standards, 24 August 2023, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.ipd.pdf>.

[I-D.draft-cfrg-schwabe-kyber-04] Schwabe, P. and B. Westerbaan, "Kyber Post-Quantum KEM", Work in Progress, Internet-Draft, draft-cfrg-schwabe-kyber-04, 2 January 2024, <https://datatracker.ietf.org/doc/html/draft-cfrg-schwabe-kyber-04>.

[I-D.ietf-pquip-pqt-hybrid-terminology-02] D, F., "Terminology for Post-Quantum Traditional Hybrid Schemes", Work in Progress, Internet-Draft, draft-ietf-pquip-pqt-hybrid-terminology-02, 2 February 2024, <https://datatracker.ietf.org/doc/html/draft-ietf-pquip-pqt-hybrid-terminology-02>.

[NIST-PQ] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography", https://csrc.nist.gov/projects/post-quantum-cryptography .

[RFC7383] Smyslov, V., "Internet Key Exchange Protocol Version 2 (IKEv2) Message Fragmentation", RFC 7383, DOI 10.17487/RFC7383, November 2014, <https://www.rfc-editor.org/rfc/rfc7383>.

[RFC8784] Fluhrer, S., Kampanakis, P., McGrew, D., and V. Smyslov, "Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security", RFC 8784, DOI 10.17487/RFC8784, June 2020, <https://www.rfc-editor.org/rfc/rfc8784>.

Acknowledgments

The authors would like to thank Valery Smyslov, Graham Bartlett, Scott Fluhrer, Ben S, and Leonie Bruckert for their valuable feedback.

Authors' Addresses

Panos Kampanakis
Amazon Web Services

Email: kpanos@amazon.com

Gerardo Ravago
Amazon Web Services

Email: gcr@amazon.com