Transport Layer Security
Internet-Draft
Intended status: Informational
Expires: 7 May 2026

D. Connolly
SandboxAQ

3 November 2025

ML-KEM Post-Quantum Key Agreement for TLS 1.3
draft-ietf-tls-mlkem-05

Abstract

This memo defines ML-KEM-512, ML-KEM-768, and ML-KEM-1024 as NamedGroups and and registers IANA values in the TLS Supported Groups registry for use in TLS 1.3 to achieve post-quantum (PQ) key establishment.

About This Document

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

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-tls-mlkem/.

Discussion of this document takes place on the Transport Layer Security Working Group mailing list (mailto:tls@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/tls/. Subscribe at https://www.ietf.org/mailman/listinfo/tls/.

Source for this draft and an issue tracker can be found at https://github.com/tlswg/draft-ietf-tls-mlkem.

Status of This Memo

Table of Contents

1. Introduction

1.1. Motivation

FIPS 203 (ML-KEM) [FIPS203] is a FIPS standard for post-quantum [RFC9794] key establishment via lattice-based key establishment mechanism (KEM). Having a purely post-quantum (not hybrid) key establishment option for TLS 1.3 is necessary for migrating beyond hybrids and for users that want or need post-quantum security without hybrids.

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. Key encapsulation mechanisms

This document models key establishment as key encapsulation mechanisms (KEMs), which consist of three algorithms:

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

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

* Decaps(sk, ct) -> shared_secret: A decapsulation algorithm, which takes as input a secret decapsulation key sk and ciphertext ct and outputs a shared secret shared_secret.

ML-KEM-512, ML-KEM-768 and ML-KEM-1024 conform to this interface:

* ML-KEM-512 has encapsulation keys of size 800 bytes, expanded decapsulation keys of 1632 bytes, decapsulation key seeds of size 64 bytes, ciphertext size of 768 bytes, and shared secrets of size 32 bytes

* ML-KEM-768 has encapsulation keys of size 1184 bytes, expanded decapsulation keys of 2400 bytes, decapsulation key seeds of size 64 bytes, ciphertext size of 1088 bytes, and shared secrets of size 32 bytes

* ML-KEM-1024 has encapsulation keys of size 1568 bytes, expanded decapsulation keys of 3168 bytes, decapsulation key seeds of size 64 bytes, ciphertext size of 1568 bytes, and shared secrets of size 32 bytes

4. Construction

The KEMs are defined as NamedGroups, sent in the supported_groups extension. Section 4.2.7 of [RFC8446]

4.1. Negotiation

Each parameter set of ML-KEM is assigned an identifier, registered by IANA in the TLS Supported Groups registry:

```
enum {

     ...,

      /* ML-KEM Key Establishment Methods */
      mlkem512(0x0200),
      mlkem768(0x0201),
      mlkem1024(0x0202)

     ...,

} NamedGroup;
```

4.2. Transmitting encapsulation keys and ciphertexts

The public encapsulation key and ciphertext values are each directly encoded with fixed lengths as in [FIPS203].

In TLS 1.3 a KEM public encapsulation key pk or ciphertext ct is represented as a KeyShareEntry Section 4.2.8 of [RFC8446]:

```
struct {
    NamedGroup group;
    opaque key_exchange<1..2^16-1>;
} KeyShareEntry;
```

These are transmitted in the extension_data fields of KeyShareClientHello and KeyShareServerHello extensions:

```
struct {
    KeyShareEntry client_shares<0..2^16-1>;
} KeyShareClientHello;

struct {
    KeyShareEntry server_share;
} KeyShareServerHello;
```

The KeyShareClientHello includes a list of KeyShareEntry structs that represent the key establishment algorithms the client supports. For each parameter of ML-KEM the client supports, the corresponding KeyShareEntry consists of a NamedGroup that indicates the appropriate parameter, and a key_exchange value that is the pk output of the KeyGen algorithm.

For the client’s share, the key_exchange value contains the pk output of the corresponding KEM NamedGroup’s KeyGen algorithm.

For the server’s share, the key_exchange value contains the ct output of the corresponding KEM NamedGroup’s Encaps algorithm.

For all parameter sets, the server MUST perform the encapsulation key check described in Section 7.2 of [FIPS203] on the client’s encapsulation key, and abort with an illegal_parameter alert if it fails.

For all parameter sets, the client MUST check if the ciphertext length matches the selected parameter set, and abort with an illegal_parameter alert if it fails.

If ML-KEM decapsulation fails for any other reason, the connection MUST be aborted with an internal_error alert.

4.3. Shared secret calculation

The shared secret output from the ML-KEM Encaps and Decaps algorithms over the appropriate keypair and ciphertext results in the same shared secret shared_secret as its honest peer, which is inserted into the TLS 1.3 key schedule in place of the (EC)DHE shared secret, as shown in Figure 1.

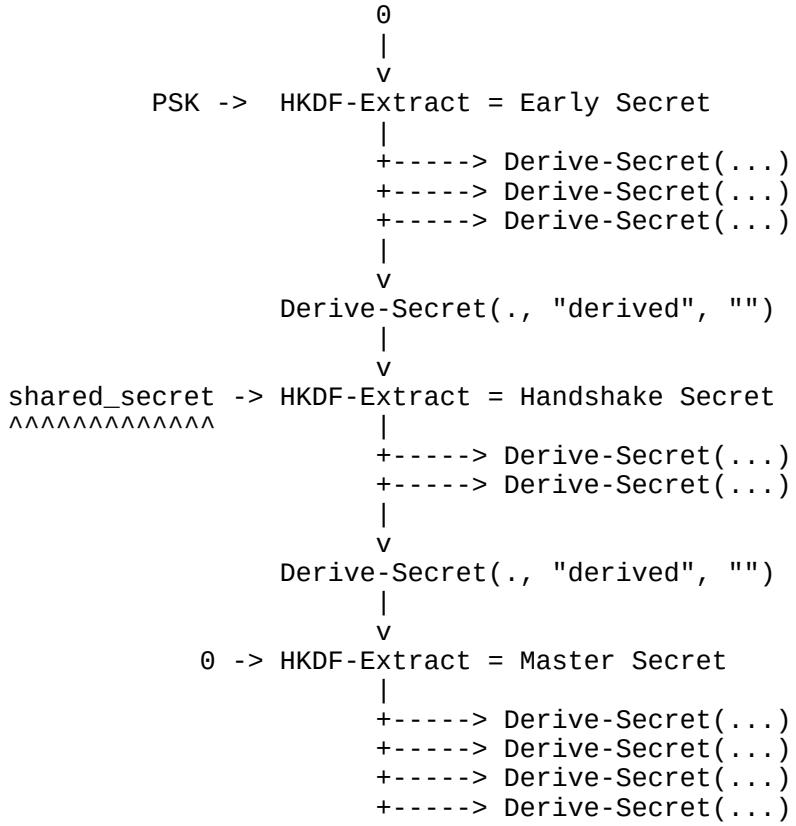

Figure 1: Key schedule for key establishment

5. Security Considerations

5.1. IND-CCA

The main security property for KEMs is indistinguishability under adaptive chosen ciphertext attack (IND-CCA), which means that shared secret values should be indistinguishable from random strings even given the ability to have other arbitrary ciphertexts decapsulated. IND-CCA corresponds to security against an active attacker, and the public key / secret key pair can be treated as a long-term key or reused. ML-KEM satisfies IND-CCA security in the random oracle model [KYBERV].

TLS 1.3 does not prohibit key re-use; some implementations may use the same ephemeral public key for more than one key establishment at the cost of limited forward secrecy. Care must be taken to ensure that keys are only re-used if the algorithms from which they are

derived are designed to be secure under key-reuse. ML-KEM's IND-CCA security satisfies this requirement such that the public key/secret key pair can be used long-term or re-used without compromising the security of the keys. However, it is still recommended that implementations avoid re-use of any keys (including ML-KEM keys) to ensure perfect forward secrecy.

Implementations MUST NOT reuse randomness in the generation of ML-KEM ciphertexts.

5.2. Binding properties

TLS 1.3's key schedule commits to the the ML-KEM encapsulation key and the ciphertext as the key_exchange field as part of the key_share extension are populated with those values are included as part of the handshake messages, providing resilience against re-encapsulation attacks against KEMs used for key establishment [CDM23].

6. IANA Considerations

This document requests/registers three new entries to the TLS Named Group (or Supported Group) registry, according to the procedures in Section 6 of [tlsiana].

Value: 0x0200

Description: MLKEM512

DTLS-OK: Y

Recommended: N

Reference: This document

Comment: FIPS 203 version of ML-KEM-512

Value: 0x0201

Description: MLKEM768

DTLS-OK: Y

Recommended: N

Reference: This document

Comment: FIPS 203 version of ML-KEM-768

Value: 0x0202

Description: MLKEM1024

DTLS-OK: Y

Recommended: N

Reference: This document

Comment: FIPS 203 version of ML-KEM-1024

7. References

7.1. Normative References

[FIPS203] "Module-lattice-based key-encapsulation mechanism standard", National Institute of Standards and Technology (U.S.), DOI 10.6028/nist.fips.203, August 2024, <https://doi.org/10.6028/nist.fips.203>.

[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>.

[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>.

[RFC8446] Rescorla, E., "The Transport Layer Security (TLS) Protocol Version 1.3", RFC 8446, DOI 10.17487/RFC8446, August 2018, <https://www.rfc-editor.org/rfc/rfc8446>.

7.2. Informative References

[AVIRAM] Nimrod Aviram, Benjamin Dowling, Ilan Komargodski, Kenny Paterson, Eyal Ronen, and Eylon Yogev, "[TLS] Combining Secrets in Hybrid Key Exchange in TLS 1.3", 1 September 2021, <https://mailarchive.ietf.org/arch/msg/tls/F4SVeL2xbGPaPB2GW_GkBbD_a5M/>.

[CDM23] Cremers, C., Dax, A., and N. Medinger, "Keeping Up with the KEMs: Stronger Security Notions for KEMs and automated analysis of KEM-based protocols", 2023, <https://eprint.iacr.org/2023/1933.pdf>.

[DOWLING] Dowling, B., Fischlin, M., G端nther, F., and D. Stebila, "A Cryptographic Analysis of the TLS 1.3 Handshake Protocol", Springer Science and Business Media LLC, Journal of Cryptology vol. 34, no. 4, DOI 10.1007/s00145-021-09384-1, July 2021, <https://doi.org/10.1007/s00145-021-09384-1>.

[FO] Fujisaki, E. and T. Okamoto, "Secure Integration of Asymmetric and Symmetric Encryption Schemes", Springer Science and Business Media LLC, Journal of Cryptology vol. 26, no. 1, pp. 80-101, DOI 10.1007/s00145-011-9114-1, December 2011, <https://doi.org/10.1007/s00145-011-9114-1>.

[HHK] Hofheinz, D., H旦velmanns, K., and E. Kiltz, "A Modular Analysis of the Fujisaki-Okamoto Transformation", Springer International Publishing, Lecture Notes in Computer Science pp. 341-371, DOI 10.1007/978-3-319-70500-2_12, ISBN ["9783319704999", "9783319705002"], 2017, <https://doi.org/10.1007/978-3-319-70500-2_12>.

[HPKE] Barnes, R., Bhargavan, K., Lipp, B., and C. Wood, "Hybrid Public Key Encryption", RFC 9180, DOI 10.17487/RFC9180, February 2022, <https://www.rfc-editor.org/rfc/rfc9180>.

[hybrid] Stebila, D., Fluhrer, S., and S. Gueron, "Hybrid key exchange in TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-hybrid-design-16, 7 September 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-hybrid-design-16>.

[KYBERV] "Formally verifying Kyber Episode V: Machine-checked IND-CCA security and correctness of ML-KEM in EasyCrypt", n.d., <https://eprint.iacr.org/2024/843.pdf>.

[LUCKY13] Al Fardan, N. J. and K. G. Paterson, "Lucky Thirteen: Breaking the TLS and DTLS record protocols", n.d., <https://ieeexplore.ieee.org/iel7/6547086/6547088/06547131.pdf>.

[RACCOON] Merget, R., Brinkmann, M., Aviram, N., Somorovsky, J., Mittmann, J., and J. Schwenk, "Raccoon Attack: Finding and Exploiting Most-Significant-Bit-Oracles in TLS-DH(E)", September 2020, <https://raccoon-attack.com/>.

[RFC9794] Driscoll, F., Parsons, M., and B. Hale, "Terminology for Post-Quantum Traditional Hybrid Schemes", RFC 9794, DOI 10.17487/RFC9794, June 2025, <https://www.rfc-editor.org/rfc/rfc9794>.

[tlsiana] Salowey, J. A. and S. Turner, "IANA Registry Updates for TLS and DTLS", Work in Progress, Internet-Draft, draft-ietf-tls-rfc8447bis-15, 21 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-rfc8447bis-15>.

Acknowledgments

Thanks to Douglas Stebila for consultation on the draft-ietf-tls-hybrid-design design, and to Scott Fluhrer, Eric Rescorla, and Rebecca Guthrie for reviews.

Author’s Address

Deirdre Connolly
SandboxAQ
Email: durumcrustulum@gmail.com