Transport Layer Security
Internet-Draft
Intended status: Informational
Expires: 2 April 2026

K. Kwiatkowski
PQShield
P. Kampanakis
AWS
B. E. Westerbaan
Cloudflare
D. Stebila
University of Waterloo

29 September 2025

Post-quantum hybrid ECDHE-MLKEM Key Agreement for TLSv1.3
draft-ietf-tls-ecdhe-mlkem-01

Abstract

This draft defines three hybrid key agreements for TLS 1.3: X25519MLKEM768, SecP256r1MLKEM768, and SecP384r1MLKEM1024 which combine a post-quantum KEM with an elliptic curve Diffie-Hellman (ECDHE).

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

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

The [hybrid] document defines a framework for combining traditional key exchanges with next-generation key exchange in TLS 1.3. The goal of this approach is to provide security against both classical and quantum adversaries while maintaining compatibility with existing infrastructure and protocols.

This document applies the framework to ML-KEM, a key encapsulation mechanism defined in [NIST-FIPS-203], and specifies code points for the hybrid groups.

1.1. Motivation

This document introduces three new supported groups for hybrid post-quantum key agreements in TLS 1.3: the X25519MLKEM768, SecP256r1MLKEM768, and SecP384r1MLKEM1024 which combine ML-KEM with ECDH in the manner of [hybrid].

* The first one uses X25519 [rfc7748], is widely deployed, and often serves as the most practical choice for a single PQ/T hybrid combiner in TLS 1.3.

* The second group uses secp256r1 (NIST P-256). This group supports use cases that require both shared secrets to be generated by FIPS-approved mechanisms.

* The third group uses secp384r1 (NIST P-384). This group is intended for high-security environments that require FIPS-approved mechanisms with an increased security margin.

Key establishment using NIST curves is outlined in Section 6.1.1.2 of [KEYAGREEMENT].

1.2. Terminology

The [hybrid] document defines "traditional" algorithms as those that are already widely adopted and "next-generation" algorithms as those that are not yet widely adopted, such as post-quantum algorithms. In this document, ECDH using Curve25519, P-256, or P-384 is considered traditional, while ML-KEM is considered next-generation.

The [hybrid] document defines a "hybrid" key exchange as one that combines a traditional key exchange with a next-generation key exchange. This document uses the term "hybrid" in the same way.

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. Negotiated Groups

3.1. Client share

When the X25519MLKEM768 group is negotiated, the client's key_exchange value is the concatenation of the client's ML-KEM-768 encapsulation key and the client's X25519 ephemeral share. The size of the client share is 1216 bytes (1184 bytes for the ML-KEM part and 32 bytes for X25519).

Note: The group name X25519MLKEM768 does not adhere to the naming convention outlined in Section 3.2 of [hybrid]. Specifically, the order of shares in the concatenation has been reversed. This is due to historical reasons.

When the SecP256r1MLKEM768 group is negotiated, the client's key_exchange value is the concatenation of the secp256r1 ephemeral share and ML-KEM-768 encapsulation key. The ECDHE share is the serialized value of the uncompressed ECDH point representation as defined in Section 4.2.8.2 of [RFC8446]. The size of the client share is 1249 bytes (65 bytes for the secp256r1 part and 1184 bytes for ML-KEM).

When the SecP384r1MLKEM1024 group is negotiated, the client's key_exchange value is the concatenation of the secp384r1 ephemeral share and the ML-KEM-1024 encapsulation key. The ECDH share is the serialized value of the uncompressed ECDH point representation as defined in Section 4.2.8.2 of [RFC8446]. The size of the client share is 1665 bytes (97 bytes for the secp384r1 and the 1568 for the ML-KEM).

3.2. Server share

When the X25519MLKEM768 group is negotiated, the server's key exchange value is the concatenation of an ML-KEM ciphertext returned from encapsulation to the client's encapsulation key, and the server's ephemeral X25519 share. The size of the server share is 1120 bytes (1088 bytes for the ML-KEM part and 32 bytes for X25519).

When the SecP256r1MLKEM768 group is negotiated, the server's key exchange value is the concatenation of the server's ephemeral secp256r1 share encoded in the same way as the client share and an ML-KEM ciphertext returned from encapsulation to the client's encapsulation key. The size of the server share is 1153 bytes (1088 bytes for the ML-KEM part and 65 bytes for secp256r1).

When the SecP384r1MLKEM1024 group is negotiated, the server's key exchange value is the concatenation of the server's ephemeral secp384r1 share encoded in the same way as the client share and an ML-KEM ciphertext returned from encapsulation to the client's encapsulation key. The size of the server share is 1665 bytes (1568 bytes for the ML-KEM part and 97 bytes for secp384r1)

For all groups, the server MUST perform the encapsulation key check described in Section 7.2 of [NIST-FIPS-203] on the client's encapsulation key, and abort with an illegal_parameter alert if it fails.

For all groups, the client MUST check if the ciphertext length matches the selected group, 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.

For all groups, both client and server MUST process the ECDH part as described in Section 4.2.8.2 of [RFC8446], including all validity checks, and abort with an illegal_parameter alert if it fails.

3.3. Shared secret

For X25519MLKEM768, the shared secret is the concatenation of the ML-KEM shared secret and the X25519 shared secret. The shared secret is 64 bytes (32 bytes for each part).

For SecP256r1MLKEM768, the shared secret is the concatenation of the ECDHE and ML-KEM shared secret. The ECDHE shared secret is the x-coordinate of the ECDH shared secret elliptic curve point represented as an octet string as defined in Section 7.4.2 of [RFC8446]. The size of the shared secret is 64 bytes (32 bytes for each part).

For SecP384r1MLKEM1024, the shared secret is the concatenation of the ECDHE and ML-KEM shared secret. The ECDHE shared secret is the x-coordinate of the ECDH shared secret elliptic curve point represented as an octet string as defined in Section 7.4.2 of [RFC8446]. The size of the shared secret is 80 bytes (48 bytes for the ECDH part and 32 bytes for the ML-KEM part).

For all groups, both client and server MUST calculate the ECDH part of the shared secret as described in Section 7.4.2 of [RFC8446], including the all-zero shared secret check for X25519, and abort the connection with an illegal_parameter alert if it fails.

4. Discussion

FIPS-compliance. All groups defined in this document permit FIPS-approved key derivation as per [NIST-SP-800-56C] and [NIST-SP-800-135]. NIST's special publication 800-56Cr2 [NIST-SP-800-56C] approves the usage of HKDF [HKDF] with two distinct shared secrets, with the condition that the first one is computed by a FIPS-approved key-establishment scheme. FIPS also requires a certified implementation of the scheme, which will remain more ubiquitous for secp256r1 in the coming years. For this reason we put the ML-KEM shared secret first in X25519MLKEM768, and the ECDH shared secret first in SecP256r1MLKEM768 and SecP384r1MLKEM1024. This means that for SecP256r1MLKEM768 and SecP384r1MLKEM1024, the ECDH implementation must be certified whereas the ML-KEM implementation does not require certification. In contrast, for X25519MLKEM768, the ML-KEM implementation must be certified.

5. Security Considerations

The same security considerations as those described in [hybrid] apply to the approach used by this document. The security analysis relies crucially on the TLS 1.3 message transcript, and one cannot assume a similar hybridisation is secure in other protocols.

Implementers are encouraged to use implementations resistant to side-channel attacks, especially those that can be applied by remote attackers.

All groups defined in this document use and generate fixed-length public keys, ciphertexts, and shared secrets, which complies with the requirements described in Section 6 of [hybrid].

6. IANA Considerations

This document requests/registers three new entries to the TLS Supported Groups registry, according to the procedures in Section 6 of [tlsiana]. These identifiers are to be used with the final, ratified by NIST, version of ML-KEM which is specified in [NIST-FIPS-203].

6.1. SecP256r1MLKEM768

Value: 4587 (0x11EB)

Description: SecP256r1MLKEM768
DTLS-OK: Y
Recommended: N
Reference: This document
Comment: Combining secp256r1 ECDH with ML-KEM-768

6.2. X25519MLKEM768

Value: 4588 (0x11EC)
Description: X25519MLKEM768
DTLS-OK: Y
Recommended: N
Reference: This document
Comment: Combining X25519 ECDH with ML-KEM-768

6.3. SecP384r1MLKEM1024

Value: 4589 (0x11ED)
Description: SecP384r1MLKEM1024
DTLS-OK: Y
Recommended: N
Reference: This document
Comment: Combining secp384r1 ECDH with ML-KEM-1024

6.4. Obsoleted Supported Groups

This document obsoletes X25519Kyber768Draft00 (25497) and SecP256r1Kyber768Draft00 (25498) in the TLS Supported Groups registry.

7. References

7.1. Normative References

[KEYAGREEMENT]
Barker, E., Chen, L., Roginsky, A., Vassilev, A., and R. Davis, "Recommendation for pair-wise key-establishment schemes using discrete logarithm cryptography", National Institute of Standards and Technology, DOI 10.6028/nist.sp.800-56ar3, April 2018, <https://doi.org/10.6028/nist.sp.800-56ar3>.

[NIST-FIPS-203]
"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>.

[NIST-SP-800-135]
Dang, Q., "Recommendation for existing application-specific key derivation functions", National Institute of Standards and Technology, DOI 10.6028/nist.sp.800-135r1, 2011, <https://doi.org/10.6028/nist.sp.800-135r1>.

[NIST-SP-800-56C]
Barker, E., Chen, L., and R. Davis, "Recommendation for Key-Derivation Methods in Key-Establishment Schemes", National Institute of Standards and Technology, DOI 10.6028/nist.sp.800-56cr2, August 2020, <https://doi.org/10.6028/nist.sp.800-56cr2>.

[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>.

[rfc7748] Langley, A., Hamburg, M., and S. Turner, "Elliptic Curves for Security", RFC 7748, DOI 10.17487/RFC7748, January 2016, <https://www.rfc-editor.org/rfc/rfc7748>.

[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

[HKDF] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC Editor, DOI 10.17487/rfc5869, May 2010, <https://doi.org/10.17487/rfc5869>.

[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>.

[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>.

Appendix A. Change log

* draft-ietf-tls-ecdhe-mlkem-01:
  - Alignment with the final version of [hybrid]
  - Added new section called Discussion and moved FIPS-compliance and Failures text there.
  - The Construction section has been removed.
* draft-ietf-tls-ecdhe-mlkem-00:
  - Change a name of the draft, following adoption by TLS WG
  - Fixes references to the to NIST ECC CDH
* draft-kwiatkowski-tls-ecdhe-mlkem-03:
  - Adds P-384 combined with ML-KEM-1024
  - Adds text that describes error-handling and outlines how the client and server must ensure the integrity of the key exchange process.
  - Adds note on the incompatibility of the codepoint name X25519MLKEM768 with [hybrid].
  - Various cosmetic changes.
* draft-kwiatkowski-tls-ecdhe-mlkem-02:
  - Adds section that mentions supported groups that this document obsoletes.
  - Fix a reference to encapsulation in the FIPS 203.
* draft-kwiatkowski-tls-ecdhe-mlkem-01:
  - Add X25519MLKEM768
* draft-kwiatkowski-tls-ecdhe-mlkem-00:
  - Change Kyber name to ML-KEM
  - Swap reference to I-D.cfrg-schwabe-kyber with FIPS-203
  - Change codepoint. New value is equal to old value + 1.

* draft-kwiatkowski-tls-ecdhe-kyber-01: Fix size of key shares generated by the client and the server

* draft-kwiatkowski-tls-ecdhe-kyber-00: updates following IANA review

Authors' Addresses

Kris Kwiatkowski
PQShield
Email: kris@amongbytes.com

Panos Kampanakis
AWS
Email: kpanos@amazon.com

Bas Westerbaan
Cloudflare
Email: bas@cloudflare.com

Douglas Stebila
University of Waterloo
Email: dstebila@waterloo.ca