Transport Layer Security (TLS)
Internet-Draft
Intended status: Standards Track
Expires: 20 August 2026

G. Wang, Ed.
Huawei Int. Pte Ltd
A. Wang
Tsinghua University

16 February 2026

Post-quantum Hybrid ECDHE-SCloud+ Key Exchange for TLS 1.3
draft-wang-tls-hybrid-ecdh-scloud-01

Abstract

This draft specifies how to enable hybrid key exchange with ECDHE and SCloud+ in Transport Layer Security protocol version 1.3 (TLS 1.3) to mitigate quantum threats. SCloud+ is an unstructured lattice based KEM (key encapsulation mechanism) with post-quantum security. This draft follows the post-quantum hybrid key exchange framework specified by [TLS.Hybrid], by concatenating the public keys and ciphertexts of ECDHE and SCloud+. Three concrete hybrid key exchange schemes are specified in this draft, which are X25519SCloud+128, SecP256r1SCloud+192 and SecP384r1SCloud+256.

[EDNOTE: ...]

Status of This Memo

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

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 20 August 2026.

Copyright Notice

Table of Contents

1. Introduction

1.1. Change History

1.2. Introduction

Cryptographically relevant quantum computers (CRQC) may pose a threat to data protected using conventional cryptographic means in the near future. The harvest-now decrypt-later (HNDL) attack threatens existing data even in the absence of a CRCQ because the data can be stored until such time that one CRCQ becomes available. As introduced in [RFC9794], hybrid mechanisms that combine post-quantum (PQ) and traditional KEMs ensure continuing security even if one of the algorithms becomes insecure.

"Hybrid key exchange in TLS 1.3" [TLS.Hybrid] specifies the framework of concatenation-based approach. In this approach, each combination of hybrid key exchange is viewed as a single new key exchange method,

negotiated and transmitted using the existing TLS 1.3 mechanisms. This approach not only achieves the primary security goal of hybrid key exchange mentioned above, but also has the benefits of backwards compatibility, high performance, low latency, and no extra round trips (refer to Section 1.5 of [TLS.Hybrid]). Following this approach, two specifications are proposed to instantiate the combinations of ECDHE with ML-KEM, and SM2 with ML-KEM: "Post-quantum Hybrid ECDHE-MLKEM Key Agreement for TLSv1.3" [ECDHE-MLKEM] and "Hybrid Post-quantum Key Exchange SM2-MLKEM for TLSv1.3" [SM2-MLKEM].

This draft specifies how to instantiate hybrid key exchange in TLS 1.3 using ECDHE and the post-quantum secure KEM SCloud+ [SCloud.SSR24] [SCloud.e24]. SCloud+ and its predecessor SCloud [SCloud.e20] are based on the unstructured LWE problem. The absence of algebraic structure, such as rings or modules, in these algorithms presumes a higher level of security than for algorithms based on structured lattices, such as ML-KEM [FIPS203], in the event of attacks that exploit structure are devised. The downside of this conservative approach is reduced computational and communication efficiency.

SCloud+ utilizes ternary secrets and lattice codes to enhance noise control and ensure robust error correction during decryption, enabling smaller parameters while maintaining low decryption failure probabilities. Compared with FrodoKEM [FrodoKEM] [I-D.LBES25], a well-known PQ KEM based on the unstructured LWE problem, SCloud+ reduces the size of public key and ciphertext from between 17 to 34 percent, depending on the chosen parameter set. The encapsulation plus decapsulation time is reduced about 24 percent.

2. Requirements Language

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. KEMs and SCloud+

3.1. KEMs and LWE

Key encapsulation mechanism (KEM) is a kind of key exchange, which allows one entity to encapsulate a secret key under a (long-term or ephemeral) public key of another entity. A KEM consists of three algorithms:

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

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

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

Among the post-quantum KEM schemes, those based on the learning with errors (LWE) problem have become prevalent. It has been proven that the LWE problem is at least as hard as the approximate shortest vector problem (SVP) and the shortest independent vectors problem (SIVP) in lattices, which remain difficult even in the sense of quantum computing. This reduction also establishes the average-case hardness of LWE, making it a strong candidate for cryptographic constructions. These schemes can be broadly divided into two categories, depending on whether algebraic structure is introduceed into the LWE problem.

The first category includes schemes built on variants of the LWE problem that incorporate algebraic structured, such as the Ring-LWE and Module-LWE problems. These structured lattice schemes include the well known NIST standard ML-KEM [FIPS203] (ML referring to Module-Lattice). The second category are KEMs whose security are based on hardness of the LWE problem without any algebraic structure. These unstructured lattice schemes, include FrodoKEM [FrodoKEM], which is well known for its conservative security.

The main benefit of introducing algebraic structure is to make LWE-based schemes more compact, i.e., more efficient in both aspects of computation and communication complexity. However, the algebraic structure also complicates the procedures of reducing the structured-LWE problems to the random lattice problems (which lack such structure), such as the approximate SVP and SIVP. Recent research results demonstrate efficient quantum algorithms for the approximate Ideal-SVP, which is related to structured lattice schemes. Although it seems unlikely that these approaches can be directly extended to address the approximate Module-SVP or the Ring-LWE/Module-LWE problems, it remains unclear about the impact of algebraic structure on the security of structured-LWE KEMs（Section 1 of [SCloud.e24]).

3.2. FrodoKEM

FrodoKEM [FrodoKEM] is built on the plain LWE encryption framework, namely, in algebraically unstructured lattices. Specifically, FrodoKEM operates with matrix-matrix multiplication modulo a power-of-2 q. In its key generation and encryption, several variables (e.g., S, U, and E) are set to be matrices so that a ciphertext can pack more message bits. Additionally, FrodoKEM uses rounded Gaussian error sampling, which adheres to the plain LWE problem. This is implemented by constructing a table to store the cumulative distribution function and performing error sampling via table lookup.

As discussed in Section 3.1, unstructured-LWE based KEMs offer conservative security, compared with structured-LWE based KEMs. FrodoKEM is now one of three KEMs in the process of ISO standardization [FrodoKEM]. The algorithm details of FrodoKEM are also specified as an Internet draft in CFRG by [I-D.LBES25]. German BSI considers FrodoKEM to be ’suitable for protecting confidential information over the long term’, while French ANSSI ’encourages including FrodoKEM as a valid and conservative option for high-security applications.’

However, the computation and communication efficiency of FrodoKEM is much worse than that of ML-KEM. As analyzed in [IKEv2-FrodoKEM], the size of public key and ciphertext for FrodoKEM is about 13 times of that of ML-KEM. This poses a major obstacle to their deployment in practical systems.

3.3. SCloud+ KEM

Similar to FrodoKEM, SCloud+ [SCloud.SSR24] is also an unstructured lattice based KEM with post-quantum security. In a nutshell, SCloud+ leverages two particular techniques to significantly enhance both computation and communication efficiency: ternary secrets and lattice coding.

A ternary secret, where each entry is selected from {0,1,-1}, can be used to significantly reduce parameter sizes and improve the scheme’ s efficiency. Ternary secrets have been widely used in homomorphic encryption schemes [SCloud.e24]. For all parameters, SCloud+ employs a ternary secret with a Hamming weight equal to half of its length. In unstructured-LWE-based schemes, two of the most time-consuming operations are the generation of matrix and the matrix-vector multiplication. Employing a ternary secret in SCloud+ improves noise control during decryption and thus enables to use a smaller ciphertext modulus ensuring correct decryption. This provides an opportunity to reduce matrix sizes while maintaining the same

security level, thereby facilitating faster matrix sampling and more efficient matrix-vector multiplication. Furthermore, setting the Hamming weight of the secret to half of its length prevents it from becoming overly sparse, and also addresses potential security concerns on sparse-secret LWE.

Lattice Coding: SCloud+ carefully selects a robust error correction method to ensure a smaller choice of parameters while maintaining a proper decryption failure probability. While the use of lattice coding is common in LWE-based constructions, previous schemes often involve lattice codes with small dimensions (4 to 16, normally). Although larger-dimensional lattice codes generally offer better signal-to-noise ratios and thus stronger error correction capabilities, they require specially designed processes to efficiently map the message to the lattice code or vice versa, which poses a challenge for high-dimensional lattice codes. SCloud+ overcomes this challenge by designing efficient labeling and delabeling for Barnes-Wall lattice codes, enabling the use of 32-dimensional lattice codes for error correction without compromising the scheme’s performance.

3.4. Comparison to FrodoKEM

FrodoKEM [FrodoKEM] has 12 variants with IND-CCA security. Namely, it offers 3 NIST security levels 1, 3, and 5; the pseudorandom generator (PRG) uses AES128 or SHAKE 128; and the KEM public key can be a long-term key (standard mode) or a short-term key (ephemeral mode). Here, just FrodoKEM (standard mode), rather than eFrodoKEM, is compared with SCloud+, regardless the PRG is AES128 or SHAKE 128.

SCloud+ provides three sets of parameters, targeting 128, 192, and 256 bits of security, which are correpsonding to NIST levels 1, 3 and 5 security, respectively. Benefiting from ternary secrets and Barnes-Wall lattice codes which are carefully selected, SCloud+ achieves a very flexible parameter selection and maintains a moderate security margins (about 88 bits) for all sets of parameters while ensuring the conformed decryption failure probability. The IND-CCA security of SCloud+ is shown, and also comprehensively analyzed using potentially the most effective attacks for LWE, including primal attack, dual attack, and hybrid attack in [SCloud.SSR24].

As demonstrated in Table 1 and Table 2, compared to FrodoKEM, SCloud+ achieves better performance in both aspects of communications and computation. Here, the original size numbers in Table 1 come from Table A.5 in [FrodoKEM], and Table 6 in [SCloud.SSR24]. About the total length of public key and ciphertext, the corresponding size of SCloud+ is shorter than that of FrodoKEM for 34.5%, 30%, and 17.5%, for security levels 1, 3, and 5, respectively.

Algorithms	decapsulation key sk	encapsulation key pk	ciphterext ct	shared secret ss
FrodoKEM-640	19,888	9,616	9,752	16
SCloud+-128	14,480	7,200	5,456	16
FrodoKEM-976	31,296	15,632	15,792	24
SCloud+-192	21,968	11,136	10,832	24
FrodoKEM-1344	43,088	21,520	21,696	32
SCloud+-256	37,304	18,744	16,916	32

Table 1: Size (in bytes) of keys and ciphertexts of FrodoKEM and SCloud+

As shown in Table 2, for the corresponding computation cost for encapsulation and decapsulation together, SCloud+ is less than that of FrodoKEM for 24.5%, 16%, and 26%, for security levels 1, 3, and 5, respectively. These original numbers come from Tables 5 and 7 in [SCloud.SSR24]. Note that here, computation cost is treated as the same as operation efficiency (in 10^3 cycles) shown in Table 2, based on x86 platform [SCloud.SSR24].

Algorithms	KeyGen	Encapsulation	Decapsulation	Enc.+Dec.
FrodoKEM-640	1,375	1,541	1,474	3,015
SCloud+-128	998	1,125	1,127	2,273
FrodoKEM-976	2,786	2,993	2,814	5,807
SCloud+-192	2,226	2,418	2,417	4,859
FrodoKEM-1344	4,906	5,183	4,922	10,174
SCloud+-256	3,454	3,671	3,824	7,539

Table 2: Operation Efficiency (in 10^3 cycles) of FrodoKEM and SCloud+

4. Negotiated Groups for ECDHE-SCloud+

4.1. Client Share and Server Share

As specified in [TLS.Hybrid], each particular combination of a hybrid key exchange is represented as a NamedGroup and sent in the supported_groups extension in TLS 1.3. No internal structure or grammar is implied or required in the value of the identifier; they are simply opaque identifiers.

This section describes the client's and server's key_exchange values for each of the following three hybrids of ECDHE-SCloud+:

* X25519SCloud+128: X25519 [RFC7748] is combined with SCloud+-128.

* SecP256r1SCloud+192: SecP256r1 (NIST P-256) [RFC8446] is combined with SCloud+-192.

* SecP384r1SCloud+256: SecP384r1 (NIST P-384) [RFC8446] is combined with SCloud+-256.

When each of these three groups is negotiated, the client's key_exchange value is the concatenation of the client's ECDH ephemeral share and the SCloud+ encapsulation key. The exact size of the client share is shown in Table 3, for all three groups.

When each of these three groups is negotiated, the server's key_exchange value is the concatenation of the server's ECDH ephemeral share and the SCloud+ encapsulated ciphertext. The exact size of the server share is also shown in Table 3, for all three groups.

Groups	Client Share	Server Share	Shared Secret
X25519SCloud+128	7,232	5,488	48
SecP256r1SCloud+192	11,201	10,897	56
SecP384r1SCloud+256	18,841	17,013	80

Table 3: The Size of Client Share, Server Share and Shared Secret (in Bytes)

4.2. Shared Secret

The shared secret is also just the the concatenation of the ECDHE shared secret and the SCloud+ shared secret. Namely, the size of shared secret is 48 (32+16) bytes for X25519SCloud+128, 56 (32+24) bytes for SecP256r1SCloud+192, and 80 (48+32) bytes for SecP384r1SCloud+256. These numbers are also shown in Table 3.

As highlighted in [TLS.Hybrid], "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."

5. Security Considerations

As SCloud+ is shown as an IND-CCA secure post-quantum KEM scheme [SCloud.SSR24], the security analysis given in [TLS.Hybrid] applies here as well. At the time of writing, there are no other security issues which may need to be considered here.

6. IANA Considerations

Table 4 below gives the list of three IANA values for the three hybrid combinations with ECDHE and SCloud+, specified in this draft. These values are to be assigned by IANA, and registered under the "TLS Supported Groups" registry of Transport Layer Security (TLS) Parameters [IANA-TLS].

Value	Description	DTLS-OK	Recommended	Reference
(TBD)4591 (0x11EF)	X25519SCloud+128	Y	N	this draft
(TBD)4592 (0x11F0)	SecP256r1SCloud+192	Y	N	this draft
(TBD)4593 (0x11F1)	SecP384r1SCloud+256	Y	N	this draft

Table 4: Updates to the IANA "TLS Supported Groups"

7. Acknowledgments

...

8. Normative References

[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/info/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/info/rfc8174>.

[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/info/rfc7748>.

[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/info/rfc8446>.

[IANA-TLS] "Transport Layer Security (TLS) Parameters", the Internet Assigned Numbers Authority (IANA). , <https://www.iana.org/assignments/tls-parameters/tls-parameters.xhtml>.

[TLS.Hybrid] Stebila, D., Fluhrer, S., and S. Gueron, "Hybrid key exchange in TLS 1.3", IETF TLS WG Document, Publication Requested), December 2025, <https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/>.

[FIPS203] National Institute of Standards and Technology, "FIPS 203: Module-Lattice-Based Key-Encapsulation Mechanism Standard", Federal Information Processing Standards Publication , August 2024, <https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.203.pdf>.

[FrodoKEM] Alkim, E., Bos, J. W., Ducas, L., Longa, P., Mironov, I., Naehrig, N., Nikolaenko, V., Peikert, C., Raghunathan, A., and D. Stebila, "FrodoKEM: Learning With Errors Key Encapsulation", Preliminary Standardization Proposal submitted to ISO , September 2025, <https://frodokem.org/files/FrodoKEM_standard_proposal_20250929.pdf>.

[SCloud.SSR24] Wang, A., Zheng, Z., Zhao, C., Qiu, Z., Zeng, G., Yuan, Y., Mu, C., and X. Wang, "SCloud+: An Efficient LWE-Based KEM Without Ring/Module Structure", Proceeding of Security Standardization Research Conference 2024 (SSR 2024), LNCS 15559, pp. 147-174, Springer, April 2025, <https://doi.org/10.1007/978-3-031-87541-0>.

9. Informative References

[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/info/rfc9794>.

[I-D.LBES25] Longa, P., Bos, J. W., Ehlen, S., and D. Stebila, "FrodoKEM: key encapsulation from learning with errors", Work in Progress, Internet Draft, September 2025, <https://datatracker.ietf.org/doc/draft-longa-cfrg-frodokem/>.

[ECDHE-MLKEM] Kwiatkowski, K., Kampanakis, P., Westerbaan, B., and S. Stebila, "Post-quantum hybrid ECDHE-MLKEM Key Agreement for TLSv1.3", IETF TLS WG Document (v03), Publication Requested, November 2025, <https://datatracker.ietf.org/doc/draft-ietf-tls-ecdhe-mlkem/>.

[SM2-MLKEM] Yang, P., Peng, C., Hu, J., and S. Sun, "Hybrid Post-quantum Key Exchange SM2-MLKEM for TLSv1.3", Work in Progress (v03), Internet Draft, November 2025, <https://datatracker.ietf.org/doc/draft-yang-tls-hybrid-sm2-mlkem/>.

[SCloud.e24] Wang, A., Zheng, Z., Zhao, C., Qiu, Z., Zeng, G., and X. Wang, "SCloud+: a Lightweight LWE-based KEM without Ring/

Module Structure", IACR Cryptology ePrint Archive, 2024/1306., November 2024, <https://eprint.iacr.org/2024/1306>.

[SCloud.e20]
Zheng, Z., Wang, A., Fan, H., Zhao, C., Liu, C., and X. Zhang, "SCloud: Public Key Encryption and Key Encapsulation Mechanism Based on Learning with Errors", IACR Cryptology ePrint Archive, 2020/95., February 2020, <https://eprint.iacr.org/2020/095>.

[IKEv2-FrodoKEM]
Wang, G., Bruckert, L., and V. Smyslov, "Post-quantum Hybrid Key Exchange in IKEv2 with FrodoKEM", Work in Progress (v03), Internet Draft, December 2025, <https://datatracker.ietf.org/doc/draft-wang-ipsecme-hybrid-kem-ikev2-frodo/>.

Authors' Addresses

Guilin Wang (editor)
Huawei Int. Pte Ltd
9 North Buona Vista Drive, #13-01
The Metropolis Tower 1
SINGAPORE 138588
Singapore
Email: wang.guilin@huawei.com

Anyu Wang
Tsinghua University
Beijing
China
Email: anyuwang@tsinghua.edu.cn