QUIC
Internet-Draft
Updates: 9001 (if approved)
Intended status: Standards Track
Expires: 8 January 2026

Y. Rosomakho
Zscaler
H. Tschofenig
H-BRS
T. Reddy
Nokia

7 July 2025

Extended Key Update for QUIC
draft-ietf-quic-extended-key-update-01

Abstract

This document specifies an Extended Key Update mechanism for the QUIC protocol, building on the foundation of the TLS Extended Key Update. The TLS Extended Key Update specification enhances the TLS protocol by introducing key updates with forward secrecy, eliminating the need to perform a full handshake. This feature is particularly beneficial for maintaining security in scenarios involving long-lived connections.

This specification replaces the QUIC Key Update mechanism described in the "Using TLS to Secure QUIC" specification.

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://quicwg.org/extended-key-update/draft-ietf-quic-extended-key-update.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-quic-extended-key-update/.

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

Source for this draft and an issue tracker can be found at https://github.com/quicwg/extended-key-update.

Status of This Memo

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

This Internet-Draft will expire on 8 January 2026.

Copyright Notice

Table of Contents

1. Introduction

The QUIC protocol [QUIC] provides a secure, versatile transport for various applications, suitable for long-lived sessions in environments like industrial IoT, telecommunication networks or Virtual Private Networks (VPN), as specified in [RFC9484].

The TLS Extended Key Update [I-D.ietf-tls-extended-key-update] introduces a mechanism to enhance the security and flexibility of encrypted communication protocols by enabling frequent key updates without requiring a full handshake renegotiation. This approach allows applications to refresh their encryption keys more often using ephemeral keys, improving forward secrecy and reducing the risk of key compromise over long-lived connections. By separating key updates from the computationally expensive handshake process, the specification provides a lightweight method for maintaining robust encryption in scenarios where connections need to remain secure for extended periods.

The TLS Extended Key Update mechanism is particularly valuable in environments where interruptions to perform a full key exchange would cause significant disruption. Other encrypted communication protocols, such as IPsec [IKEv2] and SSH [SSH-TRANSPORT], include mechanisms for rekeying without interrupting active sessions. The TLS Extended Key Update specification helps protect sensitive data even in the event of a potential key compromise by enabling frequent key rotation and leveraging forward secrecy.

This specification builds on concepts from [I-D.ietf-tls-extended-key-update] and applies them to the QUIC protocol context. It thereby replaces the QUIC Key Update mechanism described in Section 6 of [QUIC-TLS]. Unlike the previous QUIC key update process, which independently updated keys based on the Key Phase bit, the extended key update mechanism derives a new shared secret using the TLS Extended Key Update procedure. This approach enables a coordinated key transition, integrating TLS for key exchange while refining the QUIC key update process to maintain QUIC-specific key derivation.

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.

Readers are assumed to be familiar with [I-D.ietf-tls-extended-key-update].

3. Extended Key Update Negotiation

QUIC peers negotiate Extended Key Update through the TLS handshake process, as outlined in Section 4 of [I-D.ietf-tls-extended-key-update]. Extended Key Update MUST NOT be used unless both QUIC peers include the TLS flags extension [I-D.ietf-tls-tlsflags] in the handshake and set the "Extended_Key_Update" flag.

Once the Extended Key Update has been successfully negotiated, QUIC peers MUST use only the Extended Key Update process defined in this document. The standard QUIC Key Update mechanism from Section 6 of [QUIC-TLS] MUST NOT be used for the duration of the session, as both Key Update and Extended Key Update use the Key Phase bit to signal the use of updated keys. The Key Phase bit is initially set to 0 and toggled to indicate a key update following the successful post-handshake exchange of Extended Key Update messages.

4. Extended Key Update Messages

Either party MAY initiate the Extended Key Update process by sending an ExtendedKeyUpdateRequest TLS handshake message in a QUIC CRYPTO frame. This message MUST NOT be sent before the QUIC handshake is confirmed, as described in Section 4.1.2 of [QUIC-TLS]. If a QUIC endpoint receives an ExtendedKeyUpdateRequest message before the handshake is complete, it MUST terminate the connection with an error of type 0x010a, equivalent to the TLS unexpected_message alert, as specified in Section 4.8 of [QUIC-TLS].

If both QUIC peers independently initiate an Extended Key Update and their ExtendedKeyUpdateRequest messages cross in flight, the conflict MUST be resolved following the clash error handling defined in [I-D.ietf-tls-extended-key-update]. Specifically, the lexicographic order of the key_exchange value in the KeyShareEntry determines which request is rejected, ensuring a coordinated key update process without advancing by two key generations.

Upon receiving an ExtendedKeyUpdateRequest, the recipient MUST respond with an ExtendedKeyUpdateResponse TLS handshake message within a QUIC CRYPTO frame. If a QUIC endpoint receives an ExtendedKeyUpdateResponse without having previously sent an ExtendedKeyUpdateRequest, it MUST treat this as a TLS protocol error and terminate the connection with an error of type 0x010a, equivalent to the TLS unexpected_message alert, as specified in Section 4.8 of [QUIC-TLS].

The ExtendedKeyUpdateRequest and ExtendedKeyUpdateResponse messages are defined in Section 5 of [I-D.ietf-tls-extended-key-update]. Any mismatch between the negotiated NamedGroup during the initial handshake and the group used in the Extended Key Update message, or an incorrect length of the encapsulated key MUST result in connection termination with error of type 0x012f, equivalent to TLS illegal_parameter alert.

If the Extended Key Update initiator receives a retry status in the ExtendedKeyUpdateResponse message, it MUST wait for the duration specified in the response before attempting another key update. The ExtendedKeyUpdateResponse message contains a delay value (in seconds) indicating how long the initiator MUST wait before retrying. The initiator MUST NOT retry within this interval and SHOULD retry once it has lapsed. If the initiator cannot proceed without an immediate Extended Key Update, it MUST terminate the connection with an error of type TBD1, equivalent to the TLS extended_key_update_required alert.

If the initiator receives a rejected status, it MAY terminate the connection with an error of type TBD1, equivalent to the TLS extended_key_update_required alert.

5. Updating the Traffic Secrets

After sending an ExtendedKeyUpdateResponse with accepted status, the responder derives new packet protection traffic secrets. The responder MUST continue using the previous secrets until it has received a packet with the Key Phase bit flipped and has successfully decrypted it using the new keys.

After receiving and succesfully processing an ExtendedKeyUpdateResponse with accepted status, the initiator derives new packet protection traffic secrets, flips the Key Phase bit for new packets, and uses the new write secret to protect them. The initiator MUST retain the old read secret until it has received a packet with a flipped Key Phase bit from the responder and succesfully decrypted it using the new read secret.

Both endpoints SHOULD retain old read secrets for some time after unprotecting a packet encrypted with the new keys. Discarding old secret too early may cause delayed packets to be discarded, which the peer may interpreted as packet loss, potentially impacting performance.

Both endpoints SHOULD retain old read secrets for some time after successfully decrypting a packet encrypted with the new keys. Discarding old secrets too early may cause delayed packets to be

discarded, which the peer may interpret as packet loss, potentially impacting performance. However, implementations may choose to discard old secrets sooner in environments where memory limitations or security policies require minimizing the lifetime of old keys. The retention period should be chosen carefully to mitigate the risk of cryptographic attacks while still allowing late-arriving packets to be processed.

Figure 1 shows this interaction graphically where the initial set of keys used (identified with @M) are replaced by updated keys (identified with @N). The value of the Key Phase bit is indicated in brackets [].

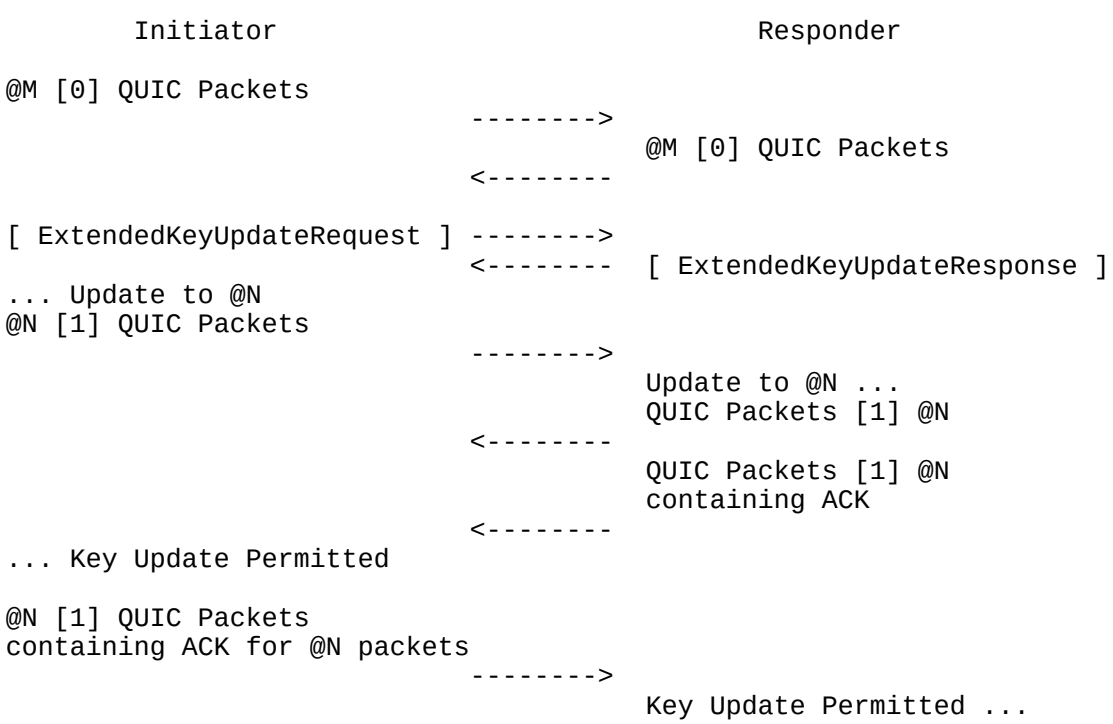

Figure 1: Extended Key Update Process in QUIC.

QUIC endpoints MUST NOT send NewKeyUpdate TLS handshake messages, defined in [I-D.ietf-tls-extended-key-update], and instead rely on the use of the Key Phase bit. Endpoints MUST treat the receipt of a TLS NewKeyUpdate message as a connection error of type 0x010a. QUIC endpoints that have agreed to the Extended Key Update process MUST NOT change the Key Phase bit without a succesful exchange of Extended Key Update TLS messages. Receiving a packet with the Key Phase bit changed without a success Extended Key Update exchange MUST be treated as a connection error of type KEY_UPDATE_ERROR (0x0e).

Key derivation function for computing the next generation of secrets is described in Section 6 of [I-D.ietf-tls-extended-key-update]. The corresponding key and IV are derived from the new secret as defined in Section 5.1 of [QUIC-TLS]. The header protection key is not updated.

6. Security Considerations

This specification describes an update to the key schedule of QUIC. Therefore, implementations MUST ensure that peers adhere strictly to the process described in this document. Packets with higher packet numbers MUST NOT be protected using an older generation of secrets, as this could compromise key synchronization and forward security.

As key exchange may be computationally intensive, responders SHOULD consider rate-limiting Extended Key Exchange requests. This can be done by responding with retry status as outlined in Section 5 of [I-D.ietf-tls-extended-key-update] and terminating connections for initiators that violate the back-off timer. This approach helps prevent excessive load on endpoints and mitigates the risk of denial-of-service attacks.

7. IANA Considerations

This document has no IANA actions.

8. References

8.1. Normative References

[I-D.ietf-tls-extended-key-update]
Tschofenig, H., T端xen, M., Reddy.K, T., Fries, S., and Y. Rosomakho, "Extended Key Update for Transport Layer Security (TLS) 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-extended-key-update-04, 3 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-extended-key-update-04>.

[I-D.ietf-tls-tlsflags]
Nir, Y., "A Flags Extension for TLS 1.3", Work in Progress, Internet-Draft, draft-ietf-tls-tlsflags-15, 15 March 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-tls-tlsflags-15>.

[QUIC] Iyengar, J., Ed. and M. Thomson, Ed., "QUIC: A UDP-Based Multiplexed and Secure Transport", RFC 9000, DOI 10.17487/RFC9000, May 2021, <https://www.rfc-editor.org/rfc/rfc9000>.

[QUIC-TLS] Thomson, M., Ed. and S. Turner, Ed., "Using TLS to Secure QUIC", RFC 9001, DOI 10.17487/RFC9001, May 2021, <https://www.rfc-editor.org/rfc/rfc9001>.

[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>.

8.2. Informative References

[IKEv2] 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>.

[RFC9484] Pauly, T., Ed., Schinazi, D., Chernyakhovsky, A., K端hlewind, M., and M. Westerlund, "Proxying IP in HTTP", RFC 9484, DOI 10.17487/RFC9484, October 2023, <https://www.rfc-editor.org/rfc/rfc9484>.

[SSH-TRANSPORT]
Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <https://www.rfc-editor.org/rfc/rfc4253>.

Acknowledgments

We would like to thank Martin Thomson and Sean Turner for their early review of this design and their invaluable feedback.

Authors’ Addresses

Yaroslav Rosomakho
Zscaler
Email: yrosomakho@zscaler.com

Hannes Tschofenig
University of Applied Sciences Bonn-Rhein-Sieg
Germany
Email: Hannes.Tschofenig@gmx.net

Tirumaleswar Reddy
Nokia
Bangalore
Karnataka
India
Email: kondtir@gmail.com