Messaging Layer Security
Internet-Draft
Intended status: Standards Track
Expires: 23 April 2026

X. Tian
B. Hale
Naval Postgraduate School
M. Mularczyk
J. Alwen
AWS

20 October 2025

Amortized PQ MLS Combiner
draft-ietf-mls-combiner-02

Abstract

This document describes a protocol for combining a traditional MLS session with a post-quantum (PQ) MLS session to achieve flexible and efficient amortized PQ confidentiality and authenticity that amortizes the computational cost of PQ Key Encapsulation Mechanisms and Digital Signature Algorithms. Specifically, we describe how to use the exporter secret of a PQ MLS session, i.e., an MLS session using a PQ ciphersuite, to seed PQ guarantees into an MLS session using a traditional ciphersuite. By supporting on-demand traditional-only key updates (a.k.a. PARTIAL updates) or hybrid-PQ key updates (a.k.a. FULL updates), we can reduce the bandwidth and computational overhead associated with PQ operations while meeting the requirement of frequent key rotations.

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://mlswg.github.io/mls-combiner/draft-ietf-mls-combiner.html. Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-mls-combiner/.

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

Source for this draft and an issue tracker can be found at https://github.com/mlswg/mls-combiner.

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 23 April 2026.

Copyright Notice

Table of Contents

1. Introduction

A fully capable quantum adversary has the ability to break fundamental underlying cryptographic assumptions of traditional asymmetric cryptography. This has led to the development of post-quantum (PQ) cryptographically secure Key Encapsulation Mechanisms (KEMs) and digital signature algorithms (DSAs) by the cryptographic research community which have been formally adopted by the National Institute of Standards and Technology (NIST), including the Module Lattice KEM (ML-KEM) and Module Lattice DSA (ML-DSA) algorithms. While these provide PQ security, ML-KEM and ML-DSA have significant overhead in terms of public key size, signature size, ciphertext size, and CPU time compared to their traditional counterparts. This has made achieving PQ entity and data authenticity particularly challenging. The hybrid approach in this draft amortizes the PQ overhead costs enabling practical PQ confidentiality or PQ confidentiality _and_ PQ authenticity.

Moreover, research arms on side-channel attacks, etc., have motivated uses of hybrid-PQ combiners that draw security from both the underlying PQ and underlying traditional components. A variety of hybrid security treatments have arisen across IETF working groups to bridge the gap between performance and security to encourage the adoption of PQ security in existing protocols, including the MLS protocol [RFC9420].

Within the MLS working group, there are various ways to approach PQ security extensions:

1. A single MLS ciphersuite for a hybrid post-quantum/traditional KEM. The

2. ciphersuite can act as a drop-in replacement for the KEM, focusing on hybrid

3. confidentiality but not authenticity, and does not incur changes elsewhere in

4. the MLS stack. As a confidentiality focus, it addresses the the harvest-now /

5. decrypt-later threat model. However, every key epoch incurs a PQ overhead cost.

6. Mechanisms that leverage hybridization as a means to not only address the

7. security balance between PQ and traditional components and achieve resistance

8. to harvest-now / decrypt-later attacks, but also use it as a means to improve

9. performance of PQ use while achieving PQ authenticity as well.

This document addresses the second topic of these work items.

2. Terminology

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.

The terms MLS client, MLS member, MLS group, Leaf Node, GroupContext, KeyPackage, Signature Key, Handshake Message, Private Message, Public Message, and RequiredCapabilities have the same meanings as in the MLS protocol [RFC9420].

3. Notation

We use terms from from MLS [RFC9420] and PQ Hybrid Terminology [I-D.ietf-pquip-pqt-hybrid-terminology]. Below, we have restated relevant terms and define new ones:

Application Message: A PrivateMessage carrying application data.

Handshake Message: A PublicMessage or PrivateMessage carrying an MLS Proposal or Commit object, as opposed to application data.

Key Derivation Function (KDF): A Hashed Message Authentication Code (HMAC)-based expand-and-extract key derivation function (HKDF) as described in [RFC5869].

Key Encapsulation Mechanism (KEM): A key transport protocol that allows two parties to obtain a shared secret based on the receiver's public key.

Post-Quantum (PQ) MLS Session: An MLS session that uses a PQ-KEM construction. It may optionally also use a PQ-DSA construction.

Traditional MLS Session: An MLS session that uses a Diffie-Hellman (DH) based KEM as described in [RFC9180].

PQ/T: A Post-Quantum and Traditional hybrid (protocol).

4. The Combiner Protocol Execution

The Amortized PQ MLS (APQ-MLS) combiner protocol runs two MLS sessions in parallel, synchronizing their group memberships. The two sessions are combined by exporting a secret from the PQ session and importing it as a Pre-Shared Key (PSK) into the traditional session. This combination process is mandatory for Commits of Add and Remove proposals in order to maintain synchronization between the sessions. However, it is optional for any other Commits (e.g. to allow for less computationally expensive traditional key rotations). Due to the higher computational costs and output sizes of PQ KEM (and signature) operations, it may be desirable to issue PQ combined (a.k.a. FULL) Commits less frequently than the traditional-only (a.k.a. PARTIAL) Commits. Since FULL Commits introduce PQ security into the MLS key schedule, the overall key schedule remains PQ-secure even when PARTIAL Commits are used. The FULL Commit rate establishes the post-quantum Post-Compromise Security (PCS) window, while the PARTIAL Commit rate can tighten the traditional PCS window even while maintaining PQ security more generally. The combiner protocol design treats both sessions as black-box interfaces so we only highlight operations requiring synchronizations in this document.

Specific update frequencies are left to the application. However, there are significant security disadvantages to infrequent FULL commits. Notably, if an application that has a threshold activity window for determining 'inactive' devices for removal, the frequency of FULL Commits MUST be greater than that threshold window; if the span between FULL Commmits exceeds the threshold window, the device MUST be considered inactive and removed from the group, even if traditional Commits are more frequent. Depending on the PARTIAL update frequency, the FULL update frequency may be significantly spread out; e.g., if a traditional update occurs at every message,

occuring frequently throughout a day, then a PQ/T update could occur once every fifty or one hundred messages. In contrast, if a traditional update occurs only once a day, then a PQ/T update frequency should occur at a far more reduced ratio to the traditional-only update frequency, for example a PQ/T update once every one or two weeks. A critical consideration is the PCS threat window of a quantum attacker within the context of the given application; FULL Commit frequencies should be calibrated accordingly.

The default way to start a APQ-MLS combined session is to create a PQ MLS session and then start a traditional MLS session with the exported PSK from the PQ session, as previously mentioned. Alternatively, a combined session can also be created after a traditional MLS session has already been running. This is done through creating a PQ MLS session with the same group members, sending a Welcome message containing the APQInfo struct in the GroupContext, and then making a FULL Commit as described in Section 4.1.

4.1. Commit Flow

Commits to proposals MAY be _PARTIAL_ or _FULL_. For a PARTIAL Commit, only the traditional session’s epoch is updated following the Propose-Commit sequence from Section 12 of [RFC9420]. For a FULL Commit, a Commit is first applied to the PQ session and another Commit is applied to the traditional session using a PSK derived from the PQ session using the DeriveExtensionSecret and apq_psk label (see Section 6.2). To ensure the correct PSK is imported into the traditional session, the sender includes information about the PSK in a PreSharedKey proposal for the traditional session’s Commit list of proposals. The information about the exported PSK is captured (shown ’=’ in the figures below for illustration purposes) by the PreSharedKeyID struct as detailed in [RFC9420]. Receivers process the PQ Commit to derive a new epoch in the PQ session and then the traditional Commit (which also includes the PSK proposal) to derive the new epoch in the traditional session.

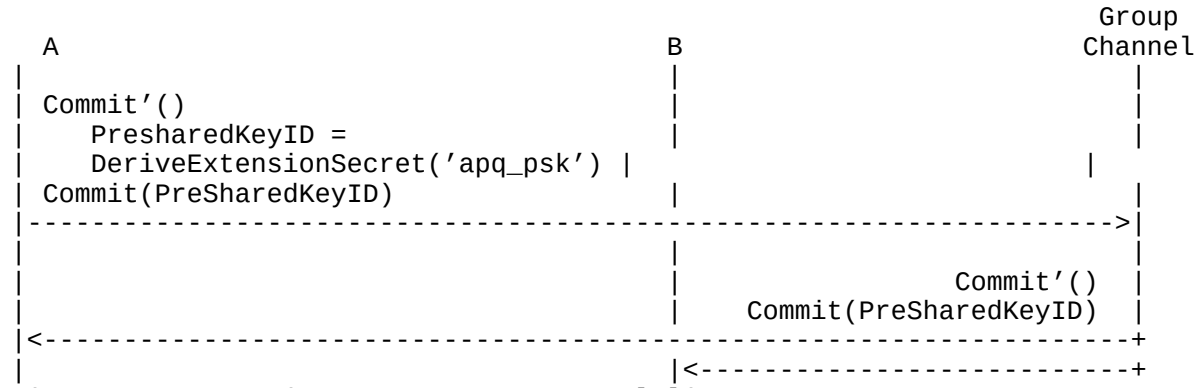

Fig 1a. FULL Commit to an empty proposal list.
Messages with ' are sent in the the PQ session.
PreSharedKeyID identifies a PSK exported from the PQ session in the new epoch following a Commit'(). The PreSharedKeyID is implicitly included in the commit in the classical session via the PreSharedKey Proposal.

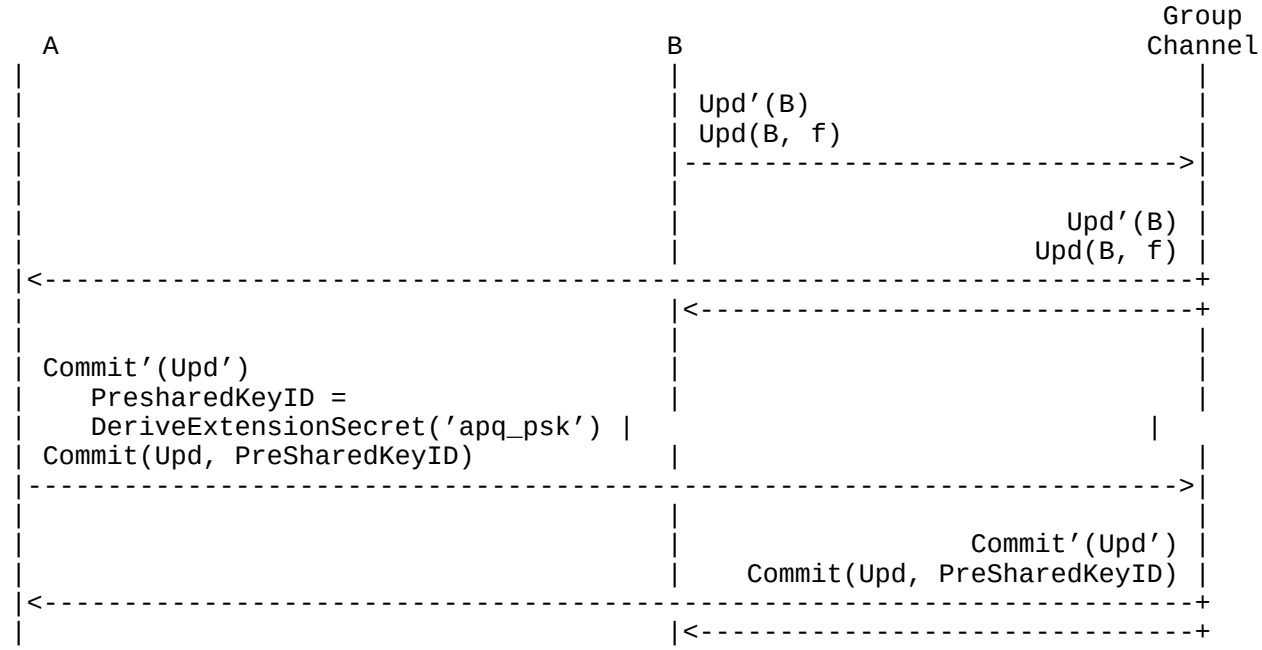

Fig 1b. FULL Commit to an Update proposal from Client B.
Messages with ' are sent in the the PQ session.

| REMARK: Fig 1b shows Client A accepting the update proposals
| from Client B as a FULL Commit. The flag f in the classical
| update proposal Upd(B, f) indicates B's intention for a FULL
| Commit to whomever Commits to its proposal.

4.2. Adding a User

User leaf nodes are first added to the PQ session following the sequence described in Section 3 of [RFC9420] except using PQ algorithms where HPKE algorithms exist. For example, a PQ-DSA signed PQ KeyPackage, i.e. containing a PQ public key, must first be published via the Distribution Service (DS). Then the associated Commit and Welcome messages will be sent and processed in the PQ session according to Section 12 of [RFC9420]. The same sequence is repeated in the standard session except following the FULL Commit combining sequence where a PreSharedKeyID proposal is additionally committed. The joiner MUST issue a FULL Commit as soon as possible after joining to achieve PCS. The FULL Commit SHOULD be the first Commit sent by the joiner.

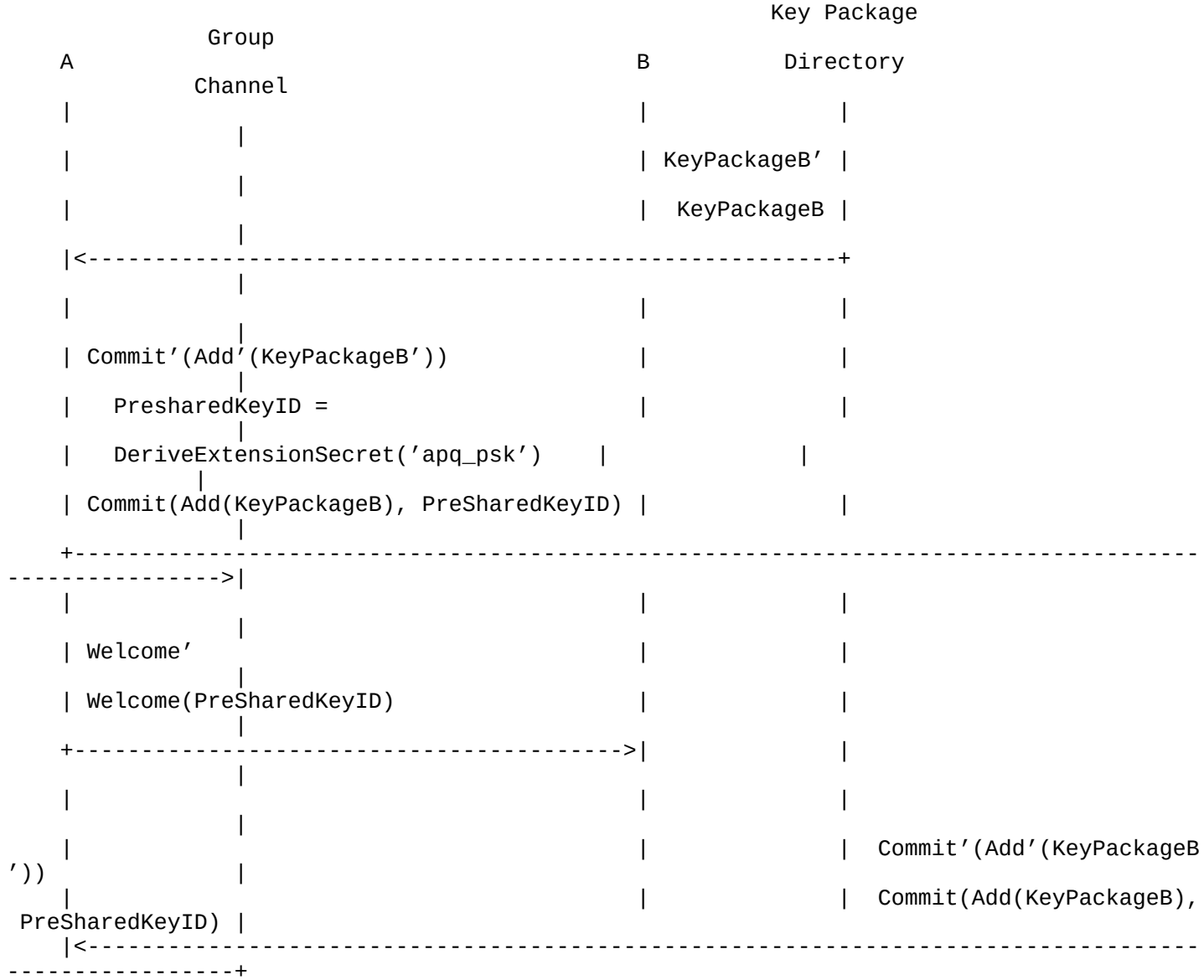

Figure 2:

Client A adds client B to the group.
Messages with ’ come from the PQ session. Processing Welcome and Commit in the traditional
session requires the PSK exported from the PQ session.

4.2.1. Welcome Message Validation

Since a client must join two sessions, the Welcome messages it receives to each session must indicate that it is not sufficient to join only one or the other. Therefore, the APQInfo struct indicating the GroupID and ciphersuites of the two sessions MUST be included in the Welcome message via serialization as a GroupContext Extension in order to validate joining the combined sessions. All members MUST verify group membership is consistent in both sessions after a join and the new member MUST issue a FULL Commit as described in Fig 1b.

4.2.2. External Joins

External joins are used by members who join a group without being explicitly added (via an Add-Commit sequence) by another existing member. The external user MUST join both the PQ session and the traditional session. As stated previously, the GroupInfo used to create the External Commit MUST contain the APQInfo struct. After joining, the new member MUST issue a FULL Commit as described in Fig 1b.

4.3. Removing a Group Member

User removals MUST be done in both PQ and traditional sessions followed by a FULL Commit Update as as described in Fig 1b. Members MUST verify group membership is consistent in both sessions after a removal.

4.4. Application Messages

APQ-MLS combiner provides PQ security to the traditional MLS session. Application messages are therefore only sent in the traditional session using the encryption_secret provided by the key schedule of the traditional session according to Section 15 of [RFC9420].

5. Modes of Operation

Security needs vary by organizations and system-specific risk tolerance and/or constraints. While this combiner protocol targets combining a PQ session and a traditional session the degree of PQ security may be tuned depending on the use-case: i.e., as PQ/T Confidentiality Only or both PQ/T Confidentiality and PQ/T Authenticity. For PQ/T Confidentiality Only, the PQ session MUST use a PQ KEM, while for PQ authenticity, the PQ session MUST use both a PQ KEM and a PQ DSA. The modes of operation are specified by the mode flag in APQInfo struct and are listed below.

5.1. PQ/T Confidentiality Only

The default mode of operation is PQ/T Confidentiality Only mode. This mode provides confidentiality and limited authenticity against quantum attackers. More precisely, it provides PQ authenticity against "outsiders", that is, against quantum attackers who do not have acces to (signature) secret keys of any group member. (Authenticity comes from the fact that the traditional session adds AEAD / MAC tags which are not available to outsiders with CRQC.) This mode does not prevent quantum impersonation attacks by other group members. That is, a group member with a CRQC can successfully impersonate another group member.

Note that an active attacker with access to a CRQC can become a group member by impersonating members in the moment they are added. As such, the authenticity guarantees outlined above only hold as long as the adversary is passive during the addition of new group members.

5.2. PQ/T Confidentiality + Authenticity

The elevated mode of operation is the PQ/T Confidentiality + Authenticity mode. Under a use environment of a cryptographically relevant quantum computer (CRQC), the threat model used in the default mode would be too weak and assurance about update authenticity is required. Recall that authenticity in MLS refers to three types of guarantees: 1) that messages were sent by a member of the group provided by the computed symmetric group key used in AEAD, 2) that key updates were performed by a valid member of the group, and 3) that a message was sent by a particular user (i.e. non-repudiation) provided by digital signatures on messages. While the symmetric group key used for AEAD in the traditional session remains protected from a CRQC adversary through the PSK from the PQ session, signatures would not be secure against forgery without using a PQ DSA to sign handshake messages nor are application messages assured to have non-repudiation against a CRQC adversary. Therefore, in the PQ/T Confidentiality + Authenticity mode, the PQ session MUST use a PQ DSA in addition to PQ KEM ciphersuites for handshake messages (the traditional session remains unchanged).

This version of PQ authenticity provides PQ authenticity to the PQ session's MLS commit messages, strengthening assurance for (1) and ensuring (2). These in turn provide PQ assurance for the key schedule from which application keys are derived in the traditional session. Application keys are used in an AEAD for protection of MLS application messages and thereby inherit the PQ security. However, it should be noted that PQ non-repudation security for application messages as described by (3) is not achieved by this mode. Achieving PQ non-repudiation on application messages would require hybrid signatures in the traditional session, with considerations to options described in [I-D.hale-pquip-hybrid-signature-spectrums].

6. Extension Requirements to MLS

The APQInfo struct contains characterizing information to signal to users that they are participating in a hybrid session. This is necessary both functionally to allow for group synchronization and as a security measure to prevent downgrading attacks to coax users into parcipating in just one of the two sessions. The group_id, cipher_suite, and epoch from both sessions (t for the traditional session and pq for the PQ session) are used as bookkeeping values to validate and synchronize group operations. The mode is a boolean value: 0 for the default PQ/T Confidentiality Only mode and 1 for the PQ/T Confidentiality + Authenticity mode.

The APQInfo struct conforms to the Safe Extensions API (see [I-D.ietf-mls-extensions]). Recall that an extension is called _safe_ if it does not modify base MLS protocol or other MLS extensions beyond using components of the Safe Extension API. This allows security analysis of our APQ-MLS Combiner protocol in isolation of the security guarantees of the base MLS protocol to enable composability of guarantees. The HPMLSInfo extension struct SHALL be in the following format:

```
struct{
    ExtensionType APQ;
    opaque extension_data<V>;
    } ExtensionContent;

struct{
    opaque t_session_group_id<V>;
    opaque PQ_session_group_id<V>;
    bool mode;
    CipherSuite t_cipher_suite;
    CipherSuite pq_cipher_suite;
    uint64 t_epoch;
    uint64 pq_epoch;
} APQInfo
```

6.1. Extension updates and validation

As mentioned in Section 4.2.1, clients MUST validate that the information in the APQInfo extensions of both T and PQ group match. As the HPQMLSInfo contains the epoch of both groups it MUST be updated in both groups when doing a FULL commit. Consequently, when doing a FULL commit in both commits MUST contain an AppDataUpdate proposal with op set to update. The update payload MUST update the epochs to the new epochs of both groups (note that the epoch of the T group may increment by more than one if one or more T only commits have been performed in the meantime).

```
enum {
  invalid(0),
  t_epoch(1),
  pq_epoch(1),
  (255)
} APQInfoUpdate

struct {
  APQInfoUpdate update;
  select (APQInfoUpdate.update)
    case epoch:
       uint64 epoch;
} APQInfoUpdateData
```

Consequently, when processing a FULL commit, recipients MUST verify that the epoch set by the APQInfoUpdateData matches the actual (new) epoch of both groups.

6.2. Key Schedule

The apq_psk exporter key derived in the PQ session MUST be derived in accordance with the Safe Extensions API guidance (see Exporting Secrets in [I-D.ietf-mls-extensions]). In particular, it SHALL NOT use the extension_secret and MUST be derived using the SafeExportSecret function as defined in Section 4.4 Pre-Shared Keys of [I-D.ietf-mls-extensions]. This is to ensure forward secrecy guarantees (see Section 9).

Even though the apq_psk PSK is not sent over the wire, members of the APQ-MLS session must agree on the value of which PSK to use. In alignment with the Safe Extensions API policy for PSKs, APQ-MLS PSKs used SHALL set PSKType = 3 and component_id = XXX (see Section 4.5 Pre-Shared Keys of [I-D.ietf-mls-extensions]).

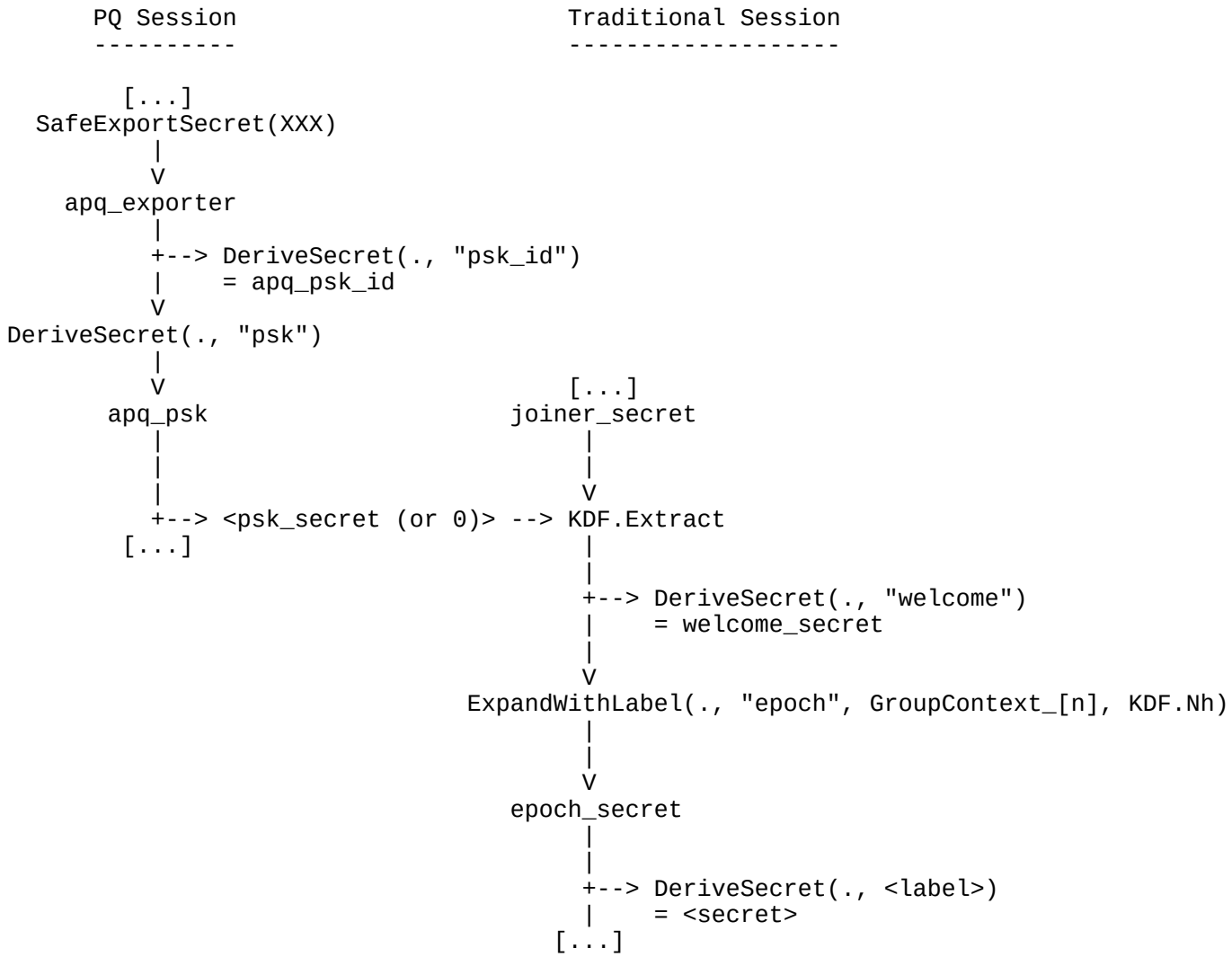

Fig 3: The apq_psk of the PQ session is injected into the key schedule of the traditional session using the safe extensions API DeriveExtensionSecret.

To signal the injection of the PSK derived from the PQ group into the key schedule of the T group, each T group commit that is part of a FULL commit MUST include a PreSharedKey proposal with psk_type = application, component_id = XXX and psk_id = apq_psk_id.

The apq_exporter MUST be deleted after both the apq_psk_id and the apq_psk were derived.

TODO: Replace occurences of XXX with the Component ID of this combiner.

7. Wire formats

Operating two groups in conjunction requires that certain data are sent over the wire in duplictate, for example, two commit messages in the case of a FULL commit. This is made easier through the following wire formats. The GroupContext of both the PQ and the T group MUST include the required_wire_formats extension listing the following wire formats.

```
struct {
  KeyPackage t_key_package;
  KeyPackage pq_key_package;
} APQKeyPackage

struct {
  MLSPublicMessage t_message;
  MLSPublicMessage pq_message;
} APQPublicMessage

struct {
  MLSPrivateMessage t_message;
  MLSPrivateMessage pq_message;
} APQPrivateMessage

struct {
  Welcome t_welcome;
  Welcome pq_welcome;
} APQWelcome

struct {
  GroupInfo t_group_info;
  GroupInfo pq_group_info;
} APQGroupInfo

struct {
  PartialGroupInfo t_group_info;
  PartialGroupInfo pq_group_info;
} APQPartialGroupInfo
```

Where PartialGroupInfo is defined in Section 4 of [I-D.mahy-mls-ratchet-tree-options]. Messages in APQPrivateMessage MUST NOT be of content type application.

TODO: IANA considerations

8. Cryptographic Objects

8.1. Cipher Suites

There are no changes to _how_ cipher suites are used to perform group key computations from RFC9420 (https://www.rfc-editor.org/rfc/rfc9420#name-cipher-suites). However, the choice of _which_ primitives are used by the traditional and PQ subsessions must be explicitly stated by the CipherSuite objects within APQInfo. So long as the traditional session only uses classical primitives and the PQ session uses PQ primitives for KEM, a APQ-MLS session is valid. Specifically, the PQ primitives for APQ-MLS must be 'pure' (fully) PQ: PQ cost is already being amoritized at the protocol level so allowing hybrid PQ cipher suites to be used in the PQ session only adds extra overhead and complexity. Furthermore, the pq_cipher_suite may contain a classical digital signature algorithm used if mode is set to 0 (PQ Confidentiality-Only) but MUST be fully PQ if mode is set to 1 (PQ Confidentiality+Authenticity). These cipher suite combinations and modes MUST not be toggled or modified after a APQ-MLS session has commenced. Clients MUST reject a APQ-MLS session with invalid or duplicate cipher suites (e.g. two traditional cipher suites).

8.1.1. Key Encapsulation Mechanism

For APQ-MLS sessions, the PQ subsession MUST use a Key Encapsulation Mechanism (KEM) that is standardized for post-quantum cryptography. The use of experimental, non-standardized, or hybrid KEMs in the PQ session is NOT RECOMMENDED and MUST be rejected by compliant clients. This requirement ensures interoperability and a consistent security baseline across all APQ-MLS deployments.

8.1.2. Signing

For APQ-MLS sessions, the choice of digital signature algorithm in the PQ subsession depends on the selected mode of operation. If the mode is set to 1 (PQ Confidentiality+Authenticity), the PQ session MUST use a digital signature algorithm that is standardized for post-quantum cryptography, such as ML-DSA as specified in FIPS 204. The use of experimental, non-standardized, or hybrid signature algorithms in the PQ session is NOT RECOMMENDED and MUST be rejected by compliant clients in this mode. If the mode is set to 0 (PQ Confidentiality-Only), the PQ session MAY use a standardized classical digital signature algorithm. These requirements ensure that the authenticity guarantees of APQ-MLS sessions are aligned with the intended security level and provide a consistent baseline for interoperability and security across deployments.

9. Security Considerations

9.1. FULL Commit Frequency

So long as the FULL Commit flow is followed for group administration actions, PQ security is extended to the traditional session. Therefore, FULL Commits can occur as frequently or infrequently as desired by any given security policy. This results in a flexible and efficient use of compute, storage, and bandwidth resources for the host by mainly calling partial updates on the traditional MLS session, given that the group membership is stable. Thus, our protocol provides PQ security and can maintain a tighter PCS window against traditional attackers as well as forward secrecy window against traditional or quantum attackers with lower overhead when compared to running a single MLS session that only uses PQ KEMs or PQ KEM/DSAs. Furthermore, the PQ PCS window against quantum attackers can be selected based on an application and even variable over time, ranging from e.g. a single FULL Commit in PQ/T Confidentiality Only mode followed by PARTIAL Commits from that point onwards (enabling general PQ/traditional confidentiality, traditional update authenticity, traditional PCS, and PQ/traditional forward secrecy) to frequent FULL Commits in the same mode (enabling general PQ/ traditional confidentiality, traditional update authenticity, PQ/ traditional PCS, and PQ/traditional forward secrecy). In PQ/T Confidentiality+Authenticity mode with frequent FULL Commits, the latter case would enable general PQ/traditional confidentiality, PQ/ traditional update authenticity, PQ/traditional PCS, and PQ/ traditional forward secrecy.

9.2. Attacks on Non-Repudiation

While PQ message integrity is provided by the symmetric key used in AEAD, attacks on non-repudiation (e.g., source forgery) on application messages may still be possible by a CRQC adversary since only traditional signatures on used after the AEAD. However, in terms of group key agreement, this is insufficient to mount anything more than a denial-of-service attack (e.g. via group state desynchronization). In terms of application messages, a traditional DSA signature may be forged by an external CRQC adversary, but the content (including sender information) is still protected by AEAD which uses the symmetric group key. Thus, an external CRQC adversary can only conduct a false-framing attack, where group members are assured of the authenticity of a message being sent by a group member for the adversary has changed the signature to imply a different sender; it would require an insider CRQC adversary to actually mount a masquerading or forgery attack, which is beyond the scope of this protocol.

If this is a concern, hybrid PQ DSAs can be used in the traditional session to sign application messages. Since this would negate much of the efficiency gains from using this protocol and enial-of-service attacks can be achieve through more expeditious means, such a option is not considered here.

9.3. Forward Secrecy

Recall that one of the ways MLS achieves forward secrecy is by deleting security sensitive values after they are consumed (e.g. to encrypt or derive other keys/nonces) and the key schedule has entered a new epoch. For example, values such as the init_secret or epoch_secret are deleted at the _start_ of a new epoch. If the MLS exporter_secret or the extension_secret from the PQ session is used directly as a PSK for the traditional session, against the requirements set above, then there is a potential scenario in which an adversary can break forward secrecy because these keys are derived _during_ an epoch and are not deleted. Therefore, the apq_psk MUST be derived from the epoch_secret created at the _start_ of an epoch from the PQ session (see Figure 3) to ensure forward secrecy.

9.4. Transport Security

Recommendations for preventing denial-of-service attacks or restricting transmitted messages are inherited from MLS.

10. IANA Considerations

The MLS sessions combined by this protocol conform to the IANA registries listed for MLS [RFC9420].

11. Normative References

[I-D.hale-pquip-hybrid-signature-spectrums]
Bindel, N., Hale, B., Connolly, D., and F. D, "Hybrid signature spectrums", Work in Progress, Internet-Draft, draft-hale-pquip-hybrid-signature-spectrums-04, 21 March 2024, <https://datatracker.ietf.org/doc/html/draft-hale-pquip-hybrid-signature-spectrums-04>.

[I-D.ietf-mls-extensions]
Robert, R., "The Messaging Layer Security (MLS) Extensions", Work in Progress, Internet-Draft, draft-ietf-mls-extensions-08, 21 July 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-mls-extensions-08>.

[I-D.ietf-pquip-pqt-hybrid-terminology]
D, F., P, M., and B. Hale, "Terminology for Post-Quantum Traditional Hybrid Schemes", Work in Progress, Internet-Draft, draft-ietf-pquip-pqt-hybrid-terminology-06, 10 January 2025, <https://datatracker.ietf.org/doc/html/draft-ietf-pquip-pqt-hybrid-terminology-06>.

[I-D.mahy-mls-ratchet-tree-options]
Mahy, R., "Ways to convey the Ratchet Tree in Messaging Layer Security", Work in Progress, Internet-Draft, draft-mahy-mls-ratchet-tree-options-03, 16 October 2025, <https://datatracker.ietf.org/doc/html/draft-mahy-mls-ratchet-tree-options-03>.

[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>.

[RFC5869] Krawczyk, H. and P. Eronen, "HMAC-based Extract-and-Expand Key Derivation Function (HKDF)", RFC 5869, DOI 10.17487/RFC5869, May 2010, <https://www.rfc-editor.org/rfc/rfc5869>.

[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>.

[RFC9180] 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>.

[RFC9420] Barnes, R., Beurdouche, B., Robert, R., Millican, J., Omara, E., and K. Cohn-Gordon, "The Messaging Layer Security (MLS) Protocol", RFC 9420, DOI 10.17487/RFC9420, July 2023, <https://www.rfc-editor.org/rfc/rfc9420>.

Acknowledgments

Contributors

Konrad Kohbrok Phoenix R&D Email: konrad.kohbrok@datashrine.de

Authors’ Addresses

Xisen Tian
Naval Postgraduate School
Email: xisen.tian1@nps.edu

Britta Hale
Naval Postgraduate School
Email: britta.hale@nps.edu

Marta Mularczyk
AWS
Email: mulmarta@amazon.ch

Jo谷l Alwen
AWS
Email: alwenjo@amazon.com