Security analysis of LAKE PQ signature method 0

This report in another language: 中文.

draft-ietf-lake-pqsuites-02 · ProVerif+ Agent report

Documents	Quantum-Resistant Cipher Suites for LAKE (protocol definition); Ephemeral Diffie-Hellman Over COSE (EDHOC) (supporting)
Properties examined	16: 14 hold, 2 fail (3 attacks)
Findings	None

Contents

1 Protocol and scope
2 Findings
3 Properties examined
4 Limits of this analysis
Appendix A: Reading of each property
Appendix C: Claims and prover answers
Appendix D: How the analysis was run

1 Protocol and scope

LAKE is a lightweight authenticated key exchange; this document registers post-quantum cipher suites for it. Method 0 of RFC 9528 is kept and the ephemeral Diffie-Hellman exchange is replaced by ML-KEM: the encapsulation key travels as G_X in message_1, the KEM ciphertext as G_Y in message_2, and the shared secret K feeds the key schedule. Both parties authenticate with ML-DSA signatures, and each holds a credential (ID_CRED_I, CRED_I and ID_CRED_R, CRED_R) with its long-term signature key; the public keys are known in advance.

The exchange is four messages: the Initiator sends message_1 with its fresh encapsulation key and C_I, the Responder answers with the KEM ciphertext and protected (C_R, ID_CRED_R, Signature_or_MAC_2), the Initiator answers with protected (ID_CRED_I, Signature_or_MAC_3), and the Responder confirms with message_4. Every transcript hash and derived key comes from the KEM secret and the messages exchanged.

When the exchange ends, the Initiator knows that a Responder holding the identity whose key verified Signature_or_MAC_2 ran this session and computed the same PRK_out. The Responder knows the same about the Initiator. Both hold PRK_out, from which application keys are exported.

The parties:

- **Initiator I**: signature method 0, ML-KEM key pair per session. Holds: ID_CRED_I, CRED_I, ML-DSA private key of I, public key of R.
- **Responder R**: signature method 0. Holds: ID_CRED_R, CRED_R, ML-DSA private key of R, public key of I.

Figure 1 shows the exchange as the documents specify it.

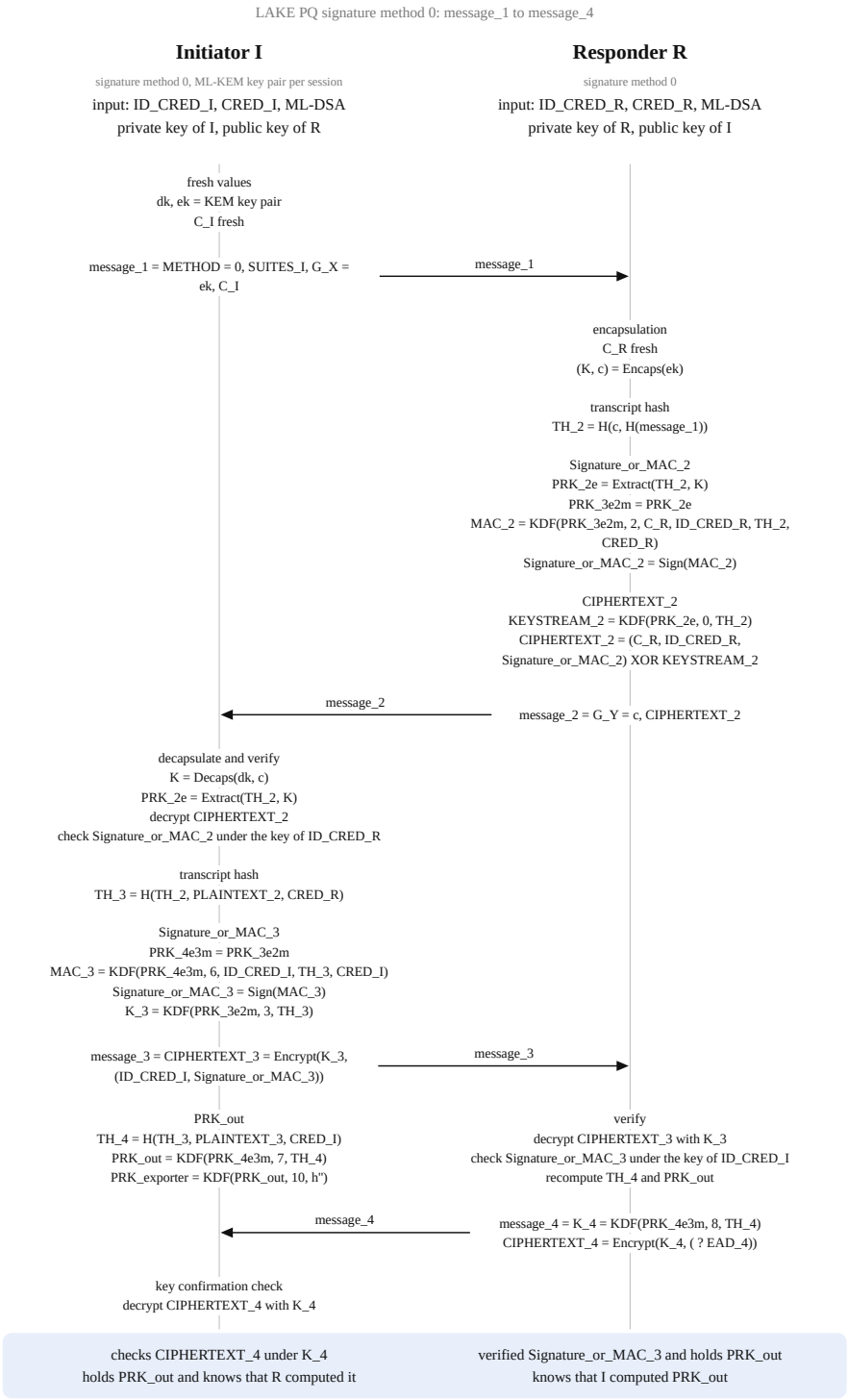


Figure 1. The protocol as specified.

The scope of the analysis is in Appendix D: what was analysed, what was left out, and the sections of the documents it is built from.

2 Findings

No finding. The analysis found nothing that the documents do not already state or set aside. The table in section 3 has the result for every property.

Within the modelled scope the cipher suite gives what the documents say it gives: no new weakness was found. Mutual authentication in both directions, agreement on the session, replay protection, key confirmation, session-key secrecy, key separation, forward secrecy after a long-term key leak and the binding of credentials, identifiers and cipher suite are verified against an attacker on the network, most of them with a control in which the protecting check is dropped or the value is revealed and the property fails. The two attacks found are the ones the documents state: a leaked KEM decapsulation key exposes that session's keys and nothing else, and an active attacker learns the Responder's credential identifier. Two claims are only partly answered: identity protection could not be settled in its indistinguishability form, and key compromise impersonation is tested only with the keys known after the session. The results rest on idealised ML-KEM, ML-DSA, hash and KDF, on one cipher suite with one supported suite per party, and on no error messages, EAD fields or linkability across runs.

3 Properties examined

Each row is one property put to ProVerif. The result is ProVerif's answer on the analyst's model of the protocol; value and whether the documents already state the result are the analyst's judgement, with the reasons in Appendix A.

ID	Property	Compromise assumed	Result	Value	In the documents
A02	**mutual authentication in method 0.** Whenever the Initiator (or the Responder) accepts this session after verifying the signature, there exists a Responder (or Initiator) session running with the peer identity of this party.	no compromise	Holds	medium	yes
A03	**matching session on both sides.** Whenever a party accepts a session, there exists a matching peer run, and both agree on the roles, identities, connection identifiers C_I/C_R, credentials CRED_I/CRED_R, transcript hashes TH_2/TH_3 and the selected cipher suite.	no compromise	Holds	medium	yes
A04	**no replay of a peer run.** Every session a party accepts corresponds to a different peer run; the same peer run cannot be accepted twice as two different sessions.	no compromise	Holds	medium	yes
A06	**key confirmation from message 4.** When the Initiator accepts message_4 (or an application message protected by an exported key), the Responder has already exported the same PRK_out and sent a matching confirmation.	no compromise	Holds	medium	yes
K01	**secrecy of session keys.** The attacker cannot derive the KEM shared secret K, PRK_out, PRK_exporter or the application keys obtained from the export interface.	no compromise	Holds	medium	yes
K04	**key separation between derived keys.** Keys derived with different info_label values, for different directions (K_3 and K_4) and in different sessions are distinct; the same (exporter_label, context) is not used for two different purposes.	no compromise	Holds	medium	yes

ID	Property	Compromise assumed	Result	Value	In the documents
A08	**transcript and credential binding.** The connection identifier, credential identifier, transcript hash and cipher suite accepted by the accepting party match the values the peer actually sent and covered in its signature, and the suite selection follows the SUITES_I ordering rule.	no compromise	Holds	medium	yes
R05	**no cipher suite downgrade.** The suite accepted by the Responder is the first suite in the Initiator's SUITES_I, in order of preference, that both parties support; a forged error message cannot make the Initiator change the order of preference or omit a better suite.	no compromise	Holds	low	yes
C01	**forward secrecy of completed sessions.** After a session completes, leaking both parties' long-term signature private keys still does not let the attacker derive the KEM shared secret, PRK_out or application keys of earlier sessions.	long-term signature private keys of the Initiator and the Responder (after the session completed and the ephemeral KEM private key was erased)	Holds	medium	yes
C04	**ephemeral KEM key leak boundary.** After leaking the Initiator's KEM decapsulation private key dk for one session, check whether that session's KEM shared secret and PRK_out are exposed and whether the keys of other sessions remain secret.	KEM decapsulation private key dk of the Initiator for this session (during or after the session)	Attack holds (stated in the documents)	low	yes
A03.11	**direction and transcript agreement.** Whenever a party accepts a session, there exists a matching peer run, and both agree on the identities, connection identifiers, transcript hashes and direction; the signature in message_2 is valid only for the Responder direction, and the signature in message_3 only for the Initiator direction.	no compromise	Holds	medium	yes
K01.12	**fresh KEM key and exporter secrecy.** Every session uses a freshly generated KEM key pair, and the attacker cannot derive PRK_out, PRK_exporter or the application keys obtained from the export interface.	no compromise	Holds	medium	yes
A08.13	**credential and identifier binding.** The credential, connection identifier and cipher suite accepted by the accepting party match the values the peer actually sent and covered in its signature and in the transcript hash.	no compromise	Holds	medium	yes
X01	**identity protection of the Initiator.** an attacker taking part as a Responder cannot tell which of two initiator credentials (ID_CRED_I, CRED_I) took part in a session in which the Initiator sends message_3	no compromise	Holds	medium	yes
X02	**responder identifier protection against passive attacker only.** an attacker who only observes cannot tell which of two responder credentials took part, but an attacker who answers with its own message_1 learns the responder's credential identifier	no compromise	Attack holds (stated in the documents)	low	yes
X03	**KCI protection under long-term key compromise.** an attacker holding the Initiator's long-term signature key cannot make the Initiator accept a session in which the Responder never ran	the long-term ML-DSA signature private keys of both parties, learned after a session completed	Holds	low	yes

"Attack holds (stated in the documents)": the prover found the attack, and the documents themselves already state it. The property fails; this is a known result, not a finding (Appendix B).

Goals the documents state that the drafted list of properties did not have, added by the analyst (Appendix D): X01, X02, X03.

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: mutual authentication of both parties and signature binding to the message direction	Both directions hold and each is bound to its own message: accepting message_2 forces a Responder run and accepting message_3 forces an Initiator run, with the same C_I, C_R and identity, and a signature of one direction cannot be used in the other. The control in which the Initiator drops the signature check accepts a forged message_2 and fails.	A02, A03.11, A03
Q2: per-session freshness of the ephemeral KEM key and exporter secrecy	Every session uses a fresh KEM key pair in the model, and K, PRK_out, PRK_exporter and the exporter-derived application key stay secret from an attacker on the network, with controls in which a leaked decapsulation key makes the same query fail.	K01.12, K01, C04
Q3: transcript binding of credentials, connection identifiers and cipher suite	Credential identifiers, connection identifiers and transcript hashes are bound to the peer's signature and to the transcript hashes, so a party accepts only values a peer run signed; the cipher suite is covered the same way, but the model fixes one selected suite and has no SUITES_I negotiation or error messages, so the ordering rule is not checked.	A08.13, A08, R05

4 Limits of this analysis

A result here is a result on a model of the protocol: it says what the specified messages and checks give or fail to give, with cryptography taken as ideal, and nothing about an implementation.

What this analysis does not cover, as the analyst states it:

- ML-KEM, ML-DSA, the hashes, the KDF, the message-2 stream cipher and the AEAD of messages 3 and 4 are ideal primitives: no computational statement, and in particular no post-quantum security, is proved.
- One cipher suite with one selected algorithm set is modelled; SUITES_I is a fixed two-element list, each party supports only the selected suite, and error messages are not modelled, so the rule against reordering SUITES_I after forged errors is not checked.
- EAD_1 to EAD_4, connection identifier update, the key update of RFC 9528 Appendix H and message deduplication at the transport layer are outside the model.
- A leaked long-term key is given to the attacker only after the session it is asked about; key compromise impersonation during a session is therefore only partly covered (X03).
- Identity protection is asked as a secrecy question about the credential identifier, not as indistinguishability between two candidate identities; the non-interference form was not proved by ProVerif.
- Linkability across runs through the clear connection identifiers C_I and C_R, which RFC 9528 9.5 notes, was not modelled.
- Side channels, timing, resource exhaustion and anything that needs counting effort rather than events are not covered.

What the host records:

- Proved without a control query (Appendix C): K04: sep_m3, K04: sep_m4, K04: sep_prk, R05: suite_agree, A08.13: suite_agree. A control is a variant of the property that must fail; without one, nothing shows that the property as written can fail in this model.

What stays open for each property, and what would settle it, is in Appendix A.

Appendix A: Reading of each property

The analyst's own reading of the prover's answers, not an answer of the prover: what the answers mean for the protocol, what the reading rests on, whether it can be used as it stands, and what it is worth.

A02: mutual authentication in method 0

Whenever the Initiator (or the Responder) accepts this session after completing signature verification, there exists a Responder (or Initiator) run that actually runs under the peer identity of this party.

Both directions of authentication hold. An Initiator that accepts message_2 has a Responder run behind it with the same C_I, C_R and the same identity, and a Responder that accepts message_3 has an Initiator run behind it. The signature check is what carries it: with the check left out of the Initiator, a forged message_2 is accepted and the property fails.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A proof of a central guarantee, but of the one the documents already claim, resting on idealised primitives.
- Stated in the documents: yes. Where the documents say it: s0036, s0002
- Rests on: ML-KEM and ML-DSA are ideal: no attacker computes the encapsulated secret from the ciphertext, nor forges a signature.; A leaked long-term key is given to the attacker only after a session, so key compromise impersonation during a session is not covered.; One cipher suite and no credential revocation.

A03: matching session on both sides

Whenever one party accepts a session, there exists a matching peer run, and both sides agree on the roles, the identities, the connection identifiers C_I/C_R, the credentials CRED_I/CRED_R, the transcript hashes TH_2/TH_3 and the selected cipher suite.

The two sides of an accepted session agree on the connection identifiers, on the credential that signed, and on the transcript they hashed. The Initiator's acceptance names the responder identity whose key verified Signature_or_MAC_2, and the Responder's acceptance names the initiator identity whose key verified Signature_or_MAC_3.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Consistency of the matched session is shown for a strong notion, including the identity, but it is a property the RFC already states.
- Stated in the documents: yes. Where the documents say it: s0190, s0223
- Rests on: Credentials are public and looked up from the identifier carried in the message; a credential cannot be misbound to another key in the model.; Idealised signature, KEM, hash and KDF.

A04: no replay of a peer run

Whenever one party accepts a session, it corresponds to a distinct peer run, and the same peer run cannot be accepted twice as two different sessions.

No recorded flow is accepted twice. A replayed message_2 fails because the keystream comes from the ephemeral KEM secret of that very session, and a replayed message_3 fails because the responder's key K_3 and transcript belong to one run. The injective form of both agreements is proved.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms with exact conditions what RFC 9528 9.1 claims about replay and cross-session injection.
- Stated in the documents: yes. Where the documents say it: s0022, s0189, s0222
- Rests on: The Initiator generates a fresh KEM key pair per session, as the draft requires.; Transport layer deduplication is not modelled and is not needed here, because the keys differ per session.; Idealised primitives.

A06: key confirmation from message 4

When the Initiator accepts message_4 (or an application message protected by the derived key), the Responder has derived the same PRK_out and sent a matching confirmation.

Accepting message_4 means the Responder derived the same PRK_out and produced CIPHERTEXT_4 under K_4, so the Initiator has key confirmation. An Initiator variant that accepts any message_4 breaks the property, so the check is what carries it.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Key confirmation is a stated guarantee; the result adds the exact role of the message_4 check.
- Stated in the documents: yes. Where the documents say it: s0281, s0266
- Rests on: K_4 and the AEAD are ideal, so a valid CIPHERTEXT_4 can only come from someone holding PRK_out.; message_4 is optional in the protocol; the model always sends it.; EAD_4 is a constant.

K01: secrecy of session keys

The attacker cannot derive the KEM shared secret K, PRK_out, PRK_exporter or the application keys obtained through the exporter interface.

The KEM shared secret, PRK_out, PRK_exporter and the application key derived through the exporter stay secret from an attacker who has only the network. The KEM secret counts as a session secret only once the peer's signature has been verified, which is where it becomes the key of a real session.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A proof of confidentiality and key security for the new cipher suite, on idealised primitives.
- Stated in the documents: yes. Where the documents say it: s0019, s0123, s0127
- Rests on: The KEM is ideal: the encapsulated secret is only obtainable by decapsulation.; A KDF break would not be seen.; An attacker that forces an aborted session does learn the secret it encapsulated to, but that session is never accepted.

K04: key separation between derived keys

Keys derived under different info_labels, in different directions (K_3 and K_4) and in different sessions are all distinct; the same (exporter_label, context) is not used for two different purposes.

Keys derived under different info_labels are independent: an attacker holding KEYSTREAM_2 of a session, and K_3 as well, does not obtain K_4 or PRK_out of that session, although it can read PLAINTEXT_2. This is the label separation Figure 6 relies on.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms the derivation discipline the RFC states, with the message-2 keystream made explicit as the weakest portion.
- Stated in the documents: yes. Where the documents say it: s0123
- Rests on: The KDF is ideal and the labels are distinct constants; a real weakness in KMAC256 output separation would not appear.; The leak is a reveal after the session, not a compromise during it.

A08: transcript and credential binding

The connection identifiers, credential identifiers, transcript hashes and cipher suite that the accepting party accepts are consistent with the values the peer actually sent and covered in the signature, and the suite selection follows the SUITES_I ordering rule.

A party accepts only the credential identifier whose key signed the peer's MAC, and the MAC covers the transcript the party holds, so credential, identifiers and transcript hashes are bound together. Changing any of them in transit makes the check fail instead of binding a wrong credential.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Identity misbinding protection is what RFC 9528 9.1 claims; the proof makes the covered fields explicit.
- Stated in the documents: yes. Where the documents say it: s0190, s0223, s0239
- Rests on: Credentials are public and the identifier-to-key lookup is assumed unambiguous.; Idealised signature and KDF.

R05: no cipher suite downgrade

The suite the Responder accepts is the first suite in the Initiator's SUITES_I, in preference order, that both sides support; forged error messages cannot make the Initiator change the preference order or omit a better suite.

Within the single selected suite, the suite is carried in message_1 and covered by TH_2, and both sides of an accepted session use the same one, so a modified SUITES_I leads to a failed session rather than to a weaker one. The negotiation itself, with its array of suites and the error messages that may be forged, is outside the model.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value low: The binding part follows structurally from the transcript hash; the part that makes downgrade protection interesting is not analysed.
- Stated in the documents: yes. Where the documents say it: s0167, s0169, s0075
- Rests on: One cipher suite and one supported suite per party.; No error messages, no SUITES_R, no retry.
- Open: SUITES_I ordering and the handling of forged error messages with SUITES_R are not modelled.; Each party supports one suite in the model, so a genuine downgrade cannot be expressed.

C01: forward secrecy of completed sessions

After a session completes, leaking both parties' long-term signature private keys still does not let the attacker derive the KEM shared secret, PRK_out or application keys of that session.

After both long-term signature keys leak, the KEM shared secret, PRK_out and PRK_exporter of completed sessions stay secret. Forward secrecy holds because the session key comes from the ephemeral KEM secret and not from the long-term keys; the control in which a session's decapsulation key leaks makes the same queries fail.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Forward secrecy is what the draft's store-now-decrypt-later argument rests on; the proof confirms the structural reason.
- Stated in the documents: yes. Where the documents say it: s0086, s0048
- Rests on: The keys are revealed only after the session, which is the question forward secrecy asks.; The ephemeral private key is assumed erased; a session whose decapsulation key also leaks is C04.; Idealised KEM: a break of ML-KEM would break this.

C04: ephemeral KEM key leak boundary

After the Initiator's KEM decapsulation private key dk for a session leaks, check whether that session's KEM shared secret and PRK_out are exposed with it, and whether the keys of other sessions stay secret.

A leaked decapsulation key of one session gives that session's KEM secret and, with the messages on the wire, its PRK_out: the attacker decapsulates, reads PLAINTEXT_2 and recomputes TH_4. It gives nothing of other sessions, whose PRK_out stays secret because each session has its own fresh key pair.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable
- Value low: Both halves follow from K = Decaps(dk, c) and the fresh-key requirement; the value is in fixing the boundary exactly.
- Stated in the documents: yes. Where the documents say it: s0018, s0022, s0123
- Rests on: The leak is per session and after it, as the fresh-key requirement implies.; The attacker is not given the long-term keys at the same time in that claim.

A03.11: direction and transcript agreement

Whenever a party accepts a session, there is a matching peer run, and both sides agree on the identities, connection identifiers, transcript hash and direction; the signature of message_2 is valid only for the Responder direction and the signature of message_3 only for the Initiator direction.

Each acceptance is matched by a peer run with the same identities, identifiers and transcript, and each signature is computed over a context of its own direction, so a signature of message_2 cannot be used as one of message_3.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It answers the direction-binding question directly and is what makes the two signatures non-transferable.
- Stated in the documents: yes. Where the documents say it: s0190, s0223, s0231
- Rests on: Idealised signature and KDF.; One suite and one credential per role.

K01.12: fresh KEM key and exporter secrecy

Each session uses a newly generated KEM key pair, and the attacker cannot derive PRK_out, PRK_exporter or the application keys obtained from the exporter interface.

Every session runs with a newly generated KEM key pair, and the resulting K, PRK_out, PRK_exporter and application key stay secret from the network attacker. The exporter key is protected only through PRK_out, so its secrecy is that of PRK_out.

- Result: Holds; answers what is asked: yes; usable with caveats

- Value medium: The fresh-key requirement is a MUST of the draft; the model shows the exporter chain inherits its secrecy.
- Stated in the documents: yes. Where the documents say it: s0022, s0057, s0128
- Rests on: Freshness is modelled by generating the key pair inside each session.; Idealised KEM and KDF.

A08.13: credential and identifier binding

The credential, connection identifiers and cipher suite that the accepting party accepts are the same values the peer actually sent and that are covered by the signatures and the transcript hash.

The credential a party accepts, the connection identifiers and the cipher suite are the ones its peer used and signed for: the identifier sits inside the MAC context and the transcript hashes, and the suite inside message_1 and thus inside TH_2. Both sides of an accepted session use the same suite.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: It confirms the field binding for the fields the model has, which is what an implementer must keep.
- Stated in the documents: yes. Where the documents say it: s0190, s0223
- Rests on: Credentials and identifiers are looked up unambiguously from the values carried in the messages.; One suite is modelled.
- Open: The ordering rule of SUITES_I is not analysed (see R05).

X01: identity protection of the Initiator

an attacker taking part as a Responder cannot tell which of two initiator credentials (ID_CRED_I, CRED_I) took part in a session in which the Initiator sends message_3

An attacker taking part as a Responder does not obtain the Initiator's credential identifier during a session: the identifier travels only inside CIPHERTEXT_3, whose key needs the KEM secret, and the Initiator sends message_3 only after the Responder proved its signature. That is the identity protection RFC 9528 9.1 claims for the Initiator against active attacks.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value medium: It reproduces a documented guarantee with the reason it holds and states which form of it was not settled.
- Stated in the documents: yes. Where the documents say it: s0036
- Rests on: The claim is secrecy of the identifier, not indistinguishability between two candidate identities; the latter was not proved.; Idealised KEM and AEAD.
- Open: The indistinguishability form of the question, whether the attacker can tell which of two initiator credentials was used, could not be settled: ProVerif cannot prove the non-interference, and the one derivation it finds uses the long-term keys that the forward-secrecy scenario reveals at the end of the run, where a leaked key identifies its holder.

X02: responder identifier protection against passive attacker only

an attacker who only observes cannot tell which of two responder credentials took part, but an attacker who answers with its own message_1 learns the responder's credential identifier

An active attacker does learn the Responder's credential identifier, so the Responder's identity is protected against passive attackers only. In the protocol the attacker only has to send its own message_1 and read the answer, because it can encapsulate to its own key pair; in the model that route and a message-2 keystream leak both end with the attacker holding ID_CRED_R.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value low: It reproduces a weakness the RFC states itself; the value is that the reason is mechanical.
- Stated in the documents: yes. Where the documents say it: s0036
- Rests on: The trace ProVerif produced reaches the identifier through the keystream reveal of the leak scenario; the protocol route needs no leak and is the one the RFC describes.; Idealised KEM.
- Open: The passive half of the statement, that an observer cannot tell which responder credential took part, was not settled as a non-interference question.

X03: KCI protection under long-term key compromise

an attacker holding the Initiator's long-term signature key cannot make the Initiator accept a session in which the Responder never ran

An attacker holding the long-term signature keys cannot make the Initiator accept a session whose Responder side never ran: the acceptance still needs a signature under the key of the identity the Initiator was given, checked against a fresh C_R and TH_2. That is the direction algorithmic key compromise impersonation protection is about, but the model gives the attacker the keys only after the run.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value low: The result is a weaker reading of the documented KCI claim; saying so is more useful than upgrading it.
- Stated in the documents: yes. Where the documents say it: s0084
- Rests on: The leak happens after the session.; One credential per role, idealised signatures.
- Open: The keys are known to the attacker only after the session in this model, so impersonation while they are live is not tested; a model in which they are known from the start loses the responder-side authentication target, which is lost by definition once the Initiator's key is public.

Appendix C: Claims and prover answers

The prover's record for each target: the level reached, the query that reached it, whether a control was checked, and the attacks with the analyst's verdict on each.

target	level	bound	lemma	control	attacks	claims
A02	proved		auth_i	checked		2 proved
A03	proved		auth_i	checked		2 proved
A04	proved		auth_i_inj	checked		2 proved
A06	proved		conf4	checked		1 proved
K01	proved		k_secret	checked		4 proved
K04	proved		sep_m3			3 proved
A08	proved		auth_i	checked		2 proved
R05	proved		suite_agree			1 proved
C01	proved		fs_prk	checked		3 proved
C04	proved		fs_prk	checked	dk_k (holds_in_source); dk_prk (holds_in_source)	1 proved, 2 attack
A03.11	proved		auth_i	checked		2 proved
K01.12	proved		k_secret	checked		2 proved
A08.13	proved		auth_r	1 of 2		2 proved
X01	proved		id_i_secret	checked		1 proved
X02	none				id_r_reveal (holds_in_source)	1 attack
X03	proved		auth_i	checked		1 proved

Claim by claim, in the analyst's wording of each claim; the answer to each is the prover's:

target	claim	attacker	answer	query	control
A02	If the Initiator accepts a session after verifying Signature_or_MAC_2, a Responder session ran with the same C_I, C_R and the same responder identity; the attacker on the network has nothing beyond the messages.	`network`	proved	auth_i	checked
A02	If the Responder accepts a session after verifying Signature_or_MAC_3, an Initiator session ran with the same C_I, C_R and the same initiator identity; the attacker on the network has nothing beyond the messages.	`network`	proved	auth_r	checked
A03	Initiator side: the C_R, ID_CRED_R and transcript hash the Initiator accepts are those of a Responder session that ran, because Signature_or_MAC_2 is checked over a context containing them.	`network`	proved	auth_i	checked
A03	Responder side: the C_I, C_R, ID_CRED_I and TH_3 the Responder accepts belong to an Initiator session that ran, because Signature_or_MAC_3 is checked over a context containing them.	`network`	proved	auth_r	checked

target	claim	attacker	answer	query	control
A04	Each Responder run is accepted by at most one Initiator acceptance: a recorded message_2 cannot be replayed to a second Initiator session, and no acceptance maps to two Responder runs.	`network`	proved	auth_i_inj	checked
A04	Each Initiator run is accepted by at most one Responder acceptance: a recorded message_3 cannot be replayed into a second Responder session.	`network`	proved	auth_r_inj	checked
A06	If the Initiator accepts message_4, a Responder session ran that derived the same PRK_out and sent the matching confirmation; the check of CIPHERTEXT_4 under K_4 is what makes this hold.	`network`	proved	conf4	checked
K01	The KEM shared secret K (G_XY) of a session the parties accept does not reach the attacker on the network.	`network`	proved	k_secret	checked
K01	PRK_out of an accepted session does not reach the attacker on the network.	`network`	proved	prk_secret	checked
K01	PRK_exporter and the application key derived from PRK_out for an accepted session do not reach the attacker on the network.	`network`	proved	exp_secret	checked
K01	The application key derived through EDHOC_Exporter from PRK_out of an accepted session does not reach the attacker on the network.	`network`	proved	app_secret	checked
K04	An attacker who holds KEYSTREAM_2 of a session does not obtain K_3 of that session, although it can then read PLAINTEXT_2.	`key_leak`	proved	sep_m3	none
K04	An attacker who holds KEYSTREAM_2 and K_3 of a session does not obtain K_4 of that session.	`key_leak`	proved	sep_m4	none
K04	An attacker who holds KEYSTREAM_2 and K_3 of a session does not obtain PRK_out of that session.	`key_leak`	proved	sep_prk	none
A08	The credential identifier the Initiator accepts (ID_CRED_R, and the CRED_R taken for it) is the one whose key signed MAC_2, so a credential cannot be misbound to another identity's key.	`network`	proved	auth_i	checked
A08	The credential identifier the Responder accepts (ID_CRED_I, and the CRED_I taken for it) is the one whose key signed MAC_3, and the transcript hashes it uses are those of the messages it received.	`network`	proved	auth_r	checked
R05	The two sides of an accepted session agree on the selected cipher suite; message_1 with its SUITES_I is covered by TH_2, so a changed suite leaves the sessions disagreeing and no acceptance.	`network`	proved	suite_agree	none
C01	An attacker that learns both long-term ML-DSA signature keys after a session does not obtain PRK_out of that session.	`ltk_leak`	proved	fs_prk	checked
C01	An attacker that learns both long-term signature keys after a session does not obtain PRK_exporter or the application keys of that session.	`ltk_leak`	proved	fs_exp	checked
C01	An attacker that learns both long-term signature keys does not obtain the KEM shared secret K of an accepted session.	`ltk_leak`	proved	fs_k	checked

target	claim	attacker	answer	query	control
C04	Leaking the ephemeral decapsulation key dk of one Initiator session exposes that session's KEM shared secret K, and with it the whole session key chain.	`kem_dk_leak`	attack (holds_in_source)	dk_k (the control of k_secret)	
C04	Leaking dk of one session exposes that session's PRK_out: the attacker decapsulates, reads PLAINTEXT_2 with KEYSTREAM_2, and recomputes TH_4 and PRK_out.	`kem_dk_leak`	attack (holds_in_source)	dk_prk (the control of prk_secret)	
C04	Keys of sessions other than the one whose dk leaked stay secret: the attacker holding one session's dk does not obtain the PRK_out of any other accepted session.	`kem_dk_leak`	proved	fs_prk	checked
A03.11	The identifiers, credentials and transcript hash the Initiator accepts match a Responder run, so the two sides are in the same session.	`network`	proved	auth_i	checked
A03.11	Direction: the Responder's acceptance is bound to the Initiator's own message_3, since MAC_3 is signed over a context holding ID_CRED_I, TH_3 and CRED_I; a signature of one direction cannot be used in the other.	`network`	proved	auth_r	checked
K01.12	Every session uses a freshly generated KEM key pair, so K and PRK_out of a session are secret from the network attacker.	`network`	proved	k_secret	checked
K01.12	PRK_out, PRK_exporter and the exporter-derived application key of an accepted session stay secret from the network attacker.	`network`	proved	prk_secret	checked
A08.13	The credential a party accepts is the one whose key signed its peer's MAC over the transcript that party holds, so credential, identifiers and transcript are bound together.	`network`	proved	auth_r	checked
A08.13	The cipher suite a party accepts is the one carried in message_1 and covered by TH_2, and both sides of an accepted session use the same suite.	`network`	proved	suite_agree	none
X01	An attacker taking part as a Responder does not obtain the Initiator's credential identifier during a session: the identifier travels only inside CIPHERTEXT_3, which needs K_3, and the Initiator sends message_3 only after verifying the Responder's signature.	`network`	proved	id_i_secret	checked
X02	An attacker who takes part as a Responder learns the Responder's credential identifier: it can encapsulate to a message_1 of its own making and read ID_CRED_R from the decrypted PLAINTEXT_2.	`network`	attack (holds_in_source)	id_r_reveal (the control of id_i_secret)	
X03	An attacker holding the long-term signature keys cannot make the Initiator accept a session whose Responder side never ran: the Initiator checks Signature_or_MAC_2 under the key of the identity it was given, and the keys it holds do not let it produce that signature for a fresh C_R and TH_2.	`ltk_leak`	proved	auth_i	checked

Appendix D: How the analysis was run

An LLM analyst builds the model of the protocol from the documents and puts each property to ProVerif; every result in section 3 is an answer of the prover.

ProVerif+ Agent session session-20261114T231222, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 48, prover 76 s, wall 17 min, cost 0.161 USD.

ProVerif: 73 proofs run, 67 decided, 6 without an answer, 56 s, 43 answers recorded.

What was analysed

Protocol	LAKE PQ signature method 0
Version	draft-ietf-lake-pqsuites-02
Analysed	Explicit choice of method 0 mutual signatures and the ML-KEM/ML-DSA cipher suite registered in draft Section 6.4; the ephemeral KEM public key in m1, ct in m2, m3 and optional m4, the full transcript/credential/context and key derivation. Not the authkem draft, so the five-message ambiguity in that document does not need correcting; pre-shared trusted public keys, algorithms idealized.
Not analysed	The symbolic model does not prove computational or post-quantum security of the algorithms; attack traces hold only under explicit assumptions and are not directly equivalent to specification flaws. Each property must have a reachability/positive control.
Source text	281 numbered passages (41464 bytes) from the sections listed below. A reference such as s0001 in this report is one of these passages

The sections of the documents the analysis is built from:

Document	Section	Title
main	2	LAKE with Quantum-Resistant Algorithms
main	3	Using KEMs in the Key Exchange
main	4	Security Considerations
main	4.1	Classical LAKE Security Properties
main	4.2	Post-quantum Security
main	4.2.1	Store Now Decrypt Later
main	4.2.2	PQ/T Hybridization
main	4.2.3	Side-channel Considerations
main	6.4	LAKE Cipher Suites Registry
rfc9528	4	Key Derivation
rfc9528	4.1	Keys for EDHOC Message Processing
rfc9528	4.1.1	EDHOC_Extract
rfc9528	4.1.1.1	PRK_2e
rfc9528	4.1.1.2	PRK_3e2m
rfc9528	4.1.1.3	PRK_4e3m
rfc9528	4.1.2	EDHOC_Expand and EDHOC_KDF

Document	Section	Title
rfc9528	4.1.3	PRK_out
rfc9528	4.2	Keys for EDHOC Applications
rfc9528	4.2.1	EDHOC_Exporter
rfc9528	5	Message Formatting and Processing
rfc9528	5.1	EDHOC Message Processing Outline
rfc9528	5.2	EDHOC Message 1
rfc9528	5.2.1	Formatting of Message 1
rfc9528	5.2.2	Initiator Composition of Message 1
rfc9528	5.2.3	Responder Processing of Message 1
rfc9528	5.3	EDHOC Message 2
rfc9528	5.3.1	Formatting of Message 2
rfc9528	5.3.2	Responder Composition of Message 2
rfc9528	5.3.3	Initiator Processing of Message 2
rfc9528	5.4	EDHOC Message 3
rfc9528	5.4.1	Formatting of Message 3
rfc9528	5.4.2	Initiator Composition of Message 3
rfc9528	5.4.3	Responder Processing of Message 3
rfc9528	5.5	EDHOC Message 4
rfc9528	5.5.1	Formatting of Message 4
rfc9528	5.5.2	Responder Composition of Message 4
rfc9528	5.5.3	Initiator Processing of Message 4

The purpose of the protocol as the source step located it in the documents: Explicit choice of method 0 mutual signatures and the ML-KEM/ML-DSA cipher suite registered in draft Section 6.4; the ephemeral KEM public key in m1, ct in m2, m3 and optional m4, the full transcript/credential/context and key derivation. Not the authkem draft, so the five-message ambiguity in that document does not need correcting; pre-shared trusted public keys, algorithms idealized.

The properties against the documents

Before a query was written, the analyst set the drafted list of properties against what the documents themselves say the protocol is for. The name, what a property is about and what the documents say are the analyst's reading; the places named exist in the documents.

ID	Name	About	What the documents say	Where
A02	mutual authentication in method 0	each side is assured of the peer's identity when it accepts a session after verifying the peer's signature	main 4.1 says the suites preserve the RFC 9528 Section 9 properties for Method 0, including mutual authentication	main:368, rfc9528:2237
A03	matching session on both sides	both parties agree on C_I, C_R, CRED_I, CRED_R and the transcript hashes of the session they complete	RFC 9528 9.1 says EDHOC provides mutual authentication with aliveness, consistency and peer awareness	rfc9528:2238

ID	Name	About	What the documents say	Where
A04	no replay of a peer run	each acceptance corresponds to a different peer session; a recorded flow is not accepted twice	RFC 9528 9.1 says the authentication coverage protects against replay and injection from another session; a fresh KEM key pair is required per session	rfc9528:2268
A06	key confirmation from message 4	the Initiator is assured that the Responder computed the same PRK_out when it accepts message_4	RFC 9528 5.5.3 says after verifying message_4 the Initiator is assured the Responder has calculated PRK_out	rfc9528:1913
K01	secrecy of session keys	K, PRK_out, PRK_exporter and the application keys stay secret from an attacker on the network	main claims confidentiality and keys security for Method 0; RFC 9528 9.1 says one session key does not compromise other session keys	main:369, rfc9528:2309
K04	key separation between derived keys	keys derived under different info_labels from the same PRK are independent	Figure 6 derives each key with a distinct info_label; 9.2 says one portion must not degrade another	rfc9528:2468, rfc9528:1354
A08	transcript and credential binding	the accepted credential identifiers, connection identifiers, transcript hashes and cipher suite match what the peer used and signed	RFC 9528 9.1 says including the credentials in the transcript hash protects against identity misbinding	rfc9528:2241
R05	no cipher suite downgrade	the selected cipher suite cannot be changed: the parties only complete with the suite the Initiator selected	RFC 9528 9.1 says EDHOC adds downgrade-protected negotiation; 5.2.2 forbids reordering SUITES_I because of forged error messages	rfc9528:2272, rfc9528:2268
C01	forward secrecy of completed sessions	session keys of completed sessions stay secret from an attacker who later learns both long-term signature keys	RFC 9528 9.1 says compromise of the private authentication keys does not compromise past session keys; 4.1.1.1 says the ephemeral shared secret gives forward secrecy	rfc9528:2284, rfc9528:2299
C04	ephemeral KEM key leak boundary	a leak of one session's dk gives that session's K and PRK_out but not other sessions' keys	K = Decaps(dk, c) and a fresh KEM key pair per session; RFC 9528 9.1 says compromise of the shared secret compromises the keys derived from it	rfc9528:2306
A03.11	direction and transcript agreement	each acceptance matches a peer run with the same identities, identifiers and transcript hashes, and each signature is valid only for its direction	RFC 9528 9.1: consistency and peer awareness; 5.3.2 and 5.4.2 sign MAC_2 and MAC_3 over different contexts	rfc9528:2238
K01.12	fresh KEM key and exporter secrecy	every session uses a new KEM key pair and PRK_out, PRK_exporter and application keys stay secret from the network attacker	a new KEM key pair is required per session; 9.1 says compromise of PRK_out compromises everything derived with EDHOC_Exporter	rfc9528:2310
A08.13	credential and identifier binding	the credential, identifiers and cipher suite a party accepts are the ones the peer used and covered by the transcript hashes and signatures	RFC 9528 9.1: including the credentials in the transcript hash protects against identity misbinding	rfc9528:2241
X01	identity protection of the Initiator	the Initiator's credential identifier and credential	RFC 9528 9.1 says EDHOC provides identity protection of the Initiator against active attacks; main lists identity protection for Method 0	rfc9528:2248, main:369
X02	responder identifier protection against passive attacker only	the Responder's credential identifier and credential	RFC 9528 9.1 says EDHOC provides identity protection of the Responder against passive attacks, and an active attacker can get the credential identifier by sending its own message_1	rfc9528:2246, rfc9528:2249

ID	Name	About	What the documents say	Where
X03	KCI protection under long-term key compromise	the Responder's authenticity towards the Initiator when the Initiator's long-term key is known	RFC 9528 9.1 (Key Compromise Impersonation) says signature-authenticated EDHOC provides protection against an attacker having access to the long-term key	rfc9528:2366

Added from the documents by the analyst: X01, X02, X03.

The security statements of the documents, paragraph by paragraph

The host listed the 124 paragraphs of the sections in which the documents say what the protocol protects and against what, and the analyst said of each what it made of it: a property, an attacker, or why neither. Taken up: 19. Not taken: 105. Which paragraphs are listed follows from the layout of the standard; where each went is the analyst's judgement.

Paragraphs	Where	Taken up as	Why not
P001 to P003	draft-ietf-jose-pqc-kem:550-571	Outside the scope	COSE PQ-KEM conventions draft, not this exchange
P004	main:355-359	No goal stated	motivation
P005	main:363-370	A02, K01, X01, X02	
P006	main:372-378	The prover cannot be asked	argument from primitive security
P007	main:382-388	The prover cannot be asked	algorithm security references
P008	main:390-393	The prover cannot be asked	random oracle model, computational
P009	main:395-399	The prover cannot be asked	quantum adversary models
P010	main:404-404	No goal stated	page footer
P011	main:408-408	The prover cannot be asked	future work in the Q1 model
P012	main:411-418	No goal stated	page header
P013	main:420-422	The prover cannot be asked	NIST categories
P014	main:424-427	The prover cannot be asked	pre-image resistance
P015	main:431-433	C01	
P016	main:437-443	The prover cannot be asked	hybrid suite composition
P017	main:445-446	No goal stated	future registration
P018	main:450-454	The prover cannot be asked	side channels
P019	main:456-458	The prover cannot be asked	implementation requirements
P020	main:462-462	No goal stated	page footer
P021	main:466-466	No goal stated	page header
P022	main:469-475	A04, C04, K01.12	
P023	main:479-480	X01, X02	
P024 to P027	rfc5116:913-940	Outside the scope	AEAD algorithm specification
P028 to P030	rfc8392:500-521	Outside the scope	CWT specification

Paragraphs	Where	Taken up as	Why not
P031 to P032	rfc8742:257-268	Outside the scope	CWT claims registry
P033 to P046	rfc8949:2406-2533	Outside the scope	CBOR encoding
P047 to P060	rfc9052:2388-2467	Outside the scope	COSE layer
P061 to P069	rfc9360:440-505	Outside the scope	CWT/COSE profile
P070	rfc9528:2231-2235	No goal stated	refers to published analyses
P071	rfc9528:2237-2244	A02, A03, A06, A08, C01, A03.11	
P072	rfc9528:2246-2255	X01, X02, `network`	
P073	rfc9528:2257-2263	A02, A08	
P074	rfc9528:2265-2269	A04, A08.13	
P075	rfc9528:2271-2278	R05, A03	
P076	rfc9528:2280-2290	C01	
P077	rfc9528:2292-2297	No goal stated	advice on credential types
P078	rfc9528:2299-2311	C01, C04	
P079	rfc9528:2313-2323	The prover cannot be asked	bit security
P080	rfc9528:2325-2337	The prover cannot be asked	MAC strength sums
P081	rfc9528:2339-2348	A02, A06, K01	
P082	rfc9528:2350-2354	Outside the scope	EAD_1 and EAD_2
P083	rfc9528:2356-2364	Outside the scope	EAD_3 and EAD_4
P084	rfc9528:2366-2375	C01, C04, X03	
P085	rfc9528:2377-2382	The prover cannot be asked	repudiation
P086	rfc9528:2384-2386	No goal stated	reference to analyses
P087	rfc9528:2390-2396	No goal stated	implementation requirement
P088	rfc9528:2398-2407	A04, C04, K01.12	
P089	rfc9528:2409-2421	X02	
P090	rfc9528:2423-2429	No goal stated	curve key validation, unused here
P091	rfc9528:2431-2441	No goal stated	key reuse advice
P092	rfc9528:2443-2456	A02, X01	
P093	rfc9528:2458-2470	K04	
P094	rfc9528:2474-2479	The prover cannot be asked	key length harmonisation
P095	rfc9528:2481-2494	The prover cannot be asked	algorithm requirements
P096 to P098	rfc9528:2498-2528	The prover cannot be asked	post-quantum considerations
P099	rfc9528:2532-2544	X02	

Paragraphs	Where	Taken up as	Why not
P100	rfc9528:2546-2558	Outside the scope	correlation through clear connection identifiers; no linkability model
P101	rfc9528:2562-2572	Outside the scope	threat model and tracking
P102	rfc9528:2574-2582	Outside the scope	length side channel
P103 to P104	rfc9528:2586-2609	The prover cannot be asked	denial of service
P105 to P117	rfc9528:2613-2745	No goal stated	implementation guidance
P118 to P119	rfc9964:283-290	Outside the scope	ML-DSA COSE profile
P120	rfc9964:294-299	C01, `ltk_leak`	
P121	rfc9964:303-309	Outside the scope	HashML-DSA rationale
P122 to P123	rfc9964:313-327	No goal stated	key validation requirement
P124	rfc9964:331-335	Outside the scope	key parameter encoding

The catalogue of properties, entry by entry

The properties were drafted from a catalogue of security and privacy properties. The draft left out 32 of its entries, and the analyst went through each of them: examined here as a property, or left out with the reason. Examined: 11. Left out: 21. That the documents do not speak of a property is not a reason the analyst may give.

Entry	Catalogue property	Examined as	Why not
K02	Payload and end-to-end confidentiality	Not applicable	no payload; the protocol only establishes keys
K03	Strong confidentiality / indistinguishability of secret values	K01	
D01	Message integrity and data origin authenticity	A03, A08	
A01	Peer participation (aliveness)	A02	
A05	Implicit key authentication	A03, A08	
A07	Identity–credential–key binding	A08.13	
R01	Authentication / challenge freshness	A04	
R02	Message-level replay resistance	A04	
R03	Business operation at most once	Not applicable	no business operation
R04	Message acceptance order / window security	Not applicable	no window or ordering rule
C02	Resistance to impersonation after key compromise (KCI)	X03	
C03	Known-key security / compromise isolation	K04, C04	
C05	Post-compromise security recovery (PCS)	Not applicable	no ratchet or rekeying in the modelled flow
C06	Strengthened authentication after compromise of enrolment credentials	Not applicable	no password record
C07	Resistance to offline dictionary attacks	Not applicable	no low-entropy secret

Entry	Catalogue property	Examined as	Why not
C08	Goal preservation under failure of a specified primitive	Not applicable	the registered suites claim nothing under primitive failure
Z01	Correct enforcement of authorization	Not applicable	no authorization
Z02	Delegated authority does not expand	Not applicable	no delegation
Z03	Authorization bound to identity / session / operation	Not applicable	no tokens
Z04	Revocation and expiry take effect	Not applicable	no revocation
Z05	Secure isolation across tenants / across services	Not applicable	no tenants
S01	Encryption nonce uniqueness	Not applicable	per-session keys used for one message
S02	Safety of retransmission / key reinstallation	Not applicable	no key installation
S03	Rollback resistance and state monotonicity	Not applicable	no state to roll back
S04	Consistency of the security context across phases	Not applicable	single handshake phase
S05	Termination and invalidation of old state	A04	
P01	Identity anonymity	X01, X02	
P02	Session unlinkability	Outside the scope	RFC 9528 9.5 notes correlation via clear connection identifiers; no linkability model was built
P03	Hiding of user / credential existence	Not applicable	no directory to enumerate
V01	Conditional progress / error recovery	The prover cannot be asked	liveness
V02	No unintended deadlock	The prover cannot be asked	liveness
V03	Abuse of constrained resources / amplification resistance	The prover cannot be asked	resource exhaustion

The parties and the protected values, one by one

The host listed the 28 parties and protected values that the drafted properties name, and the analyst said of each which attacker controls the party or holds the value, or why no such attacker was set. With an attacker: 14. Without: 14. The list follows from the properties; what was made of each entry is the analyst's judgement.

ID	Party or value	Kind	Attacker	Why none
S01	Initiator I	party	`ltk_leak`, `kem_dk_leak`	
S02	Responder R	party	`ltk_leak`	
S03	Attacker	party	Nothing gained	this is the adversary itself; the network attacker is assumed throughout
S04	Authenticity of the peer's identity	value	`ltk_leak`	
S05	Connection identifiers C_I, C_R	value	Nothing gained	public on the wire
S06	Credentials CRED_I, CRED_R	value	Nothing gained	credentials are public
S07	transcript hashes TH_2, TH_3	value	Nothing gained	intermediate hashes, no key without prk_2e

ID	Party or value	Kind	Attacker	Why none
S08	Selected cipher suite	value	Nothing gained	sent in clear in message_1
S09	Session uniqueness	value	Nothing gained	not a holdable value; replay is A04
S10	Ephemeral KEM key pair	value	`kem_dk_leak`	
S11	PRK_out	value	`kem_dk_leak`, `ltk_leak`, `key_leak`	
S12	K_4	value	`key_leak`	
S13	KEM shared secret K	value	`kem_dk_leak`	
S14	PRK_exporter	value	`kem_dk_leak`, `ltk_leak`, `key_leak`	
S15	Application export key	value	`kem_dk_leak`, `ltk_leak`	
S16	K_3	value	`key_leak`	
S17	KEYSTREAM_2	value	`key_leak`	
S18	Export key	value	`ltk_leak`, `kem_dk_leak`	
S19	C_I, C_R	value	Nothing gained	public on the wire
S20	ID_CRED_I, ID_CRED_R	value	Nothing gained	identifier values; which one a session used is asked by the two identity goals
S21	CRED_I, CRED_R	value	Nothing gained	credentials are public
S22	TH_2, TH_3, TH_4	value	Nothing gained	intermediate hashes
S23	SUITES_I	value	Nothing gained	sent in clear
S24	KEM shared secret K of a past session	value	`ltk_leak`	
S25	Keys of other sessions	value	`kem_dk_leak`	
S26	Identity	value	Nothing gained	identities are public; which one a session used is asked by the identity goals
S27	TH_2, TH_3	value	Nothing gained	intermediate hashes
S28	Direction	value	Nothing gained	fixed by the message that carries the signature

The properties under each attacker

The attackers are the analyst's, set before a query was written; each answer was recorded under one of them. A cell is the prover's answer for the property under that attacker; "No answer" is an attacker set for the property under which the prover left a claim undecided or nothing was recorded; an empty cell is an attacker the analyst did not set for the property.

Attacker	Has	Basis	Where
`network`	the network only: it reads, drops, changes and sends messages, and holds no secret of a party		
`ltk_leak`	the long-term ML-DSA signature private keys of both parties, learned after a session completed	RFC 9528 9.1 states what compromise of the private authentication keys does and does not give	rfc9528:2299
`kem_dk_leak`	the ML-KEM decapsulation key of one Initiator session, learned after that session, plus the network	K = Decaps(dk, c) and the draft requires a fresh KEM key pair per session, so a leaked dk belongs to one session	rfc9528:2306

Attacker	Has	Basis	Where
key_leak	KEYSTREAM_2 and K_3 of one session, learned after that session, plus the network	Figure 6 derives every key under its own info_label and 9.2 says one portion must not degrade another	rfc9528:2468

ID	network	ltk_leak	kem_dk_leak	key_leak
A02	Holds			
A03	Holds			
A04	Holds			
A06	Holds			
K01	Holds			
K04				Holds
A08	Holds			
R05	Holds			
C01		Holds		
C04			Fails	
A03.11	Holds			
K01.12	Holds			
A08.13	Holds			
X01	Holds			
X02	Fails			
X03		Holds		

Source and targets

Source: 281 units, 41464 bytes, frozen in source-delivery-01.json. Targets: 10 approved (A02, A03, A04, A06, K01, K04, A08, R05, C01, C04); the questions asked: 3. The list of targets was approved as drafted, beforehand, with --approve-targets.

Decisions the host took by its documented default (task `decisions: auto`), for you to check:

- approve_package: approved the source package protocol-3d24772d27239387d0704487813b

This step as it reports itself: 1 requests (1 failed), 0.000 USD, 36304 tokens in and 5066 out, 31 s.

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
42	A02, A03, A04, A06, K01, K04, A08, R05, C01, C04, A03.11, K01.12, A08.13, X01, X02, X03	plain_primitives, many_sessions	Every answer rests on ML-KEM as an ideal encapsulation (decapsulate with dk gives back exactly the encapsulated secret), ML-DSA signatures, and hash/KDF/stream-cipher/AEAD as ideal one-way primitives; no algebraic laws beyond that are used. The properties are asked over any number of sessions of both roles. No target rests on a counter, sequence number or key that is replaced across steps, so there is no bounded unrolling and no bounded.pv.

The analyst's closing note

Model of LAKE PQ signature method 0 (ML-KEM replacing the ephemeral DH of RFC 9528 method 0, ML-DSA signatures, one selected PQ cipher suite, message_1 to message_4 with the full transcript and key schedule) built in ProVerif; the normal run exists and all 16 targets have an answer.

Established (all on model.pv, attacker on the network unless named): mutual authentication in both directions (A02, A03, A03.11) and its injective form, i.e. no replay of message_2 or message_3 (A04), each with a control party that drops the signature check and fails; key confirmation from message_4 (A06, control without the check); secrecy of the KEM shared secret, PRK_out, PRK_exporter and the exporter-derived application key (K01, K01.12); key separation, so a leaked KEYSTREAM_2 and K_3 give neither K_4 nor PRK_out (K04, under an attacker holding those keys); credential, identifier and transcript binding (A08, A08.13); forward secrecy, i.e. both long-term signature keys leaking after the session gives no past session key (C01, with controls in which a decapsulation key leaks); and that the Initiator's credential identifier does not reach an attacker taking part as a Responder (X01).

Attacks found, both already stated by the documents and submitted as holds_in_source: a leaked KEM decapsulation key exposes that session's K and PRK_out and nothing of other sessions (C04), and an active attacker learns the Responder's credential identifier, so the Responder's identity is protected against passive attackers only (X02). No finding beyond the documents.

Partial: R05 (cipher suite downgrade) is answered only for the binding of the selected suite, because SUITES_I negotiation and forged error messages are not modelled; X03 (key compromise impersonation) is answered with the long-term keys known only after the session, so impersonation while the keys are live is not tested, and the identity-protection questions could not be settled in their non-interference form (ProVerif cannot prove them; the only derivation it gives uses the revealed long-term keys, which identify their holder by nature). Blocks: a Tamarin model with a long-term key known during a session would settle X03, and one without the phase-1 key reveal would settle the identity indistinguishability question. All answers rest on idealised ML-KEM, ML-DSA, hash and KDF, recorded with `fit` (plain_primitives, many_sessions); no bounded.pv was needed because no target rests on a value that changes across steps.