Security analysis of TLS hybrid HelloRetryRequest profile

This report in another language: 中文.

rfc10024 · ProVerif+ Agent report

Documents	Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3 (protocol definition); Hybrid Key Exchange in TLS 1.3 (supporting); The Transport Layer Security (TLS) Protocol Version 1.3 (supporting)
Properties examined	16: 15 hold, 0 fail, 1 without an answer
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

The protocol is the TLS 1.3 profile with a single HelloRetryRequest and the fixed hybrid group X25519MLKEM768. A client and a server take part: the client holds the server's certificate public key, while the server holds its certificate signing key and a cookie integrity key. The server first answers ClientHello1 with a HelloRetryRequest that selects the group and carries a cookie bound to the hash of ClientHello1; the client then sends ClientHello2 with the hybrid key share and the cookie, and the server authenticates with Certificate, CertificateVerify and Finished over the transcript reconstructed from message_hash(ClientHello1).

When the exchange ends the client is sure it talked to the server named by its certificate and that the group and transcript are the ones it offered and the server echoed. Both sides hold the same hybrid shared secret and the same handshake and application traffic keys, and neither the shared secret nor the server signing key is known to an attacker on the network.

The parties:

- **Client**: TLS client, offers X25519MLKEM768. Holds: server certificate public key.
- **Server**: TLS server, sends one HelloRetryRequest. Holds: certificate signing key, cookie integrity key.

Figure 1 shows the exchange as the documents specify it.

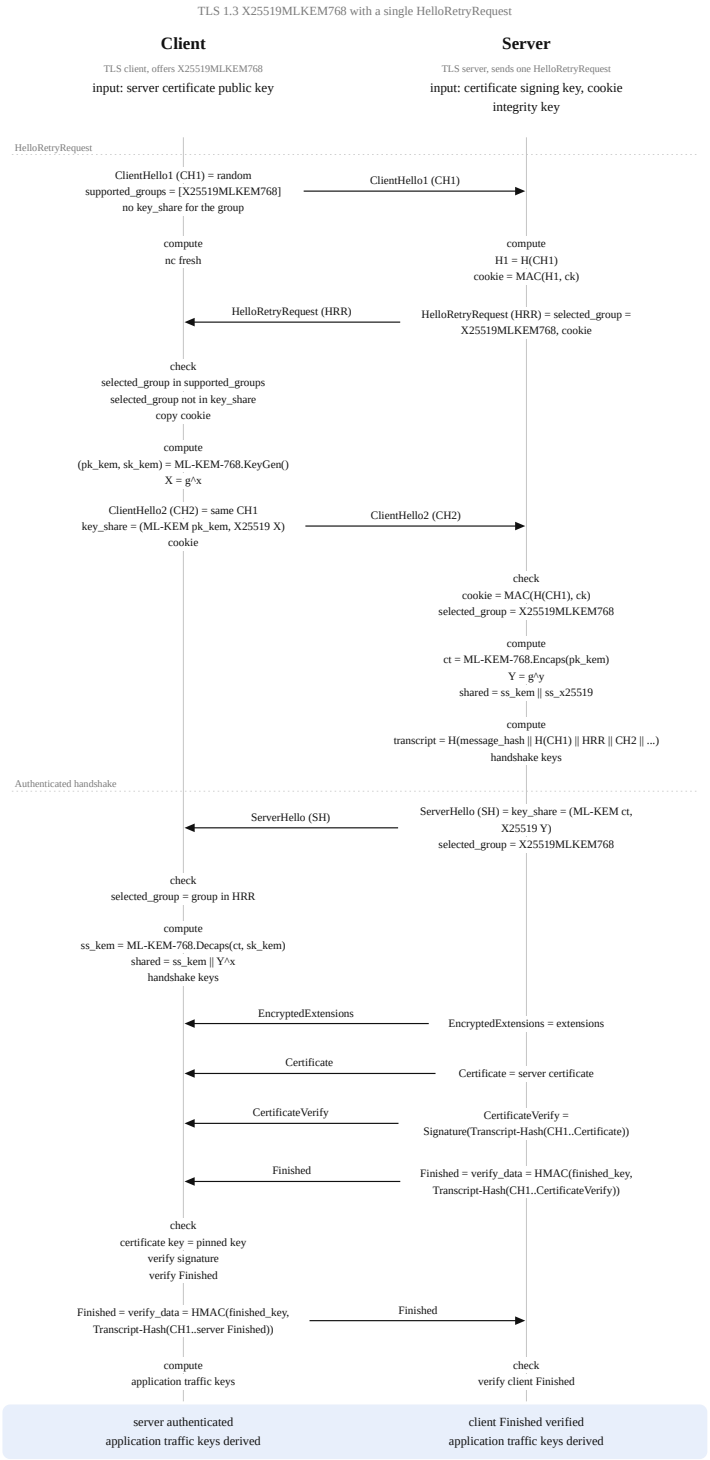


Figure 1. The protocol as specified.

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 symbolic model the profile gives what it is meant to give: server authentication after the retry, binding of the HRR, cookie and transcript to ClientHello1, rejection of repeated or spliced retries, secrecy of the shared secret and keys, and the hybrid and forward-secrecy properties. No finding is new: every guarantee proved is already stated in the documents for the attacker it is proved against, and every named attack is excluded. One property is open: strong secrecy (indistinguishability) of the shared secret was not decided, because the symbolic KEM with a single global chosen secret lets an attacker test a candidate; that needs Tamarin+ Agent with a per-session secret. The results hold for the plain-primitive abstraction (KEM, Diffie-Hellman with g^xy=g^yx, hash, MAC, signature) and for any number of sessions, and they do not cover the record layer, the computational strength of the primitives, or the all-zero/small-order X25519 check.

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
A03	**server authentication after HRR.** Whenever the client accepts completion of the handshake after the HRR flow, there is a real server run whose identity, negotiated group X25519MLKEM768, both parties' key shares and transcript match the values accepted by the client.	no compromise	Holds	medium	yes
A04	**no double acceptance of a server run.** Every time the client successfully completes the handshake after the HRR flow, this corresponds to a distinct real server run; the same server run cannot be accepted twice by the client.	no compromise	Holds	medium	yes
A08	**HRR parameters bound to transcript.** The selected_group, cipher suite and selected_version accepted by the client are bound to the transcript, all corresponding to its CH1 offer, the HRR contents and the reconstruction of message_hash(CH1); an attacker cannot substitute unauthenticated values.	no compromise	Holds	low	yes
K01	**secrecy of shared secret and keys.** The attacker cannot derive the X25519MLKEM768 shared secret, the handshake traffic keys, the application traffic keys, or the server's long-term signing private key.	no compromise	Holds	medium	yes
K04	**key separation and freshness.** The handshake/application traffic keys derived by the client and the server are separated by direction and purpose, and keys of different sessions are not unexpectedly the same.	no compromise	Holds	medium	yes
R01	**cookie bound to current CH1.** The HRR and its cookie accepted by the client must correspond to the current CH1; an old HRR/cookie cannot be replayed as valid authentication for this run.	no compromise	Holds	medium	yes
R02	**no replay of HRR or ClientHello.** The same HRR or CH message instance is accepted at most once within a single connection; an HRR spliced across connections cannot be accepted as a valid handshake.	no compromise	Holds	medium	yes
C01	**forward secrecy of session keys.** After a session completes, leaking the server's long-term signing private key still does not let the attacker obtain the handshake or application traffic keys of previously completed sessions.	server long-term signing private key (after the session completes)	Holds	medium	yes

ID	Property	Compromise assumed	Result	Value	In the documents
C04	**hybrid security under one bad component.** Leaking either the X25519 ephemeral private key share or the ML-KEM private key share still does not let the attacker obtain the session shared secret and traffic keys.	X25519 ephemeral private key share or ML-KEM private key share (during or after key exchange)	Holds	medium	yes
C08	**ML-KEM protects when X25519 breaks.** When the X25519 primitive fails (the attacker obtains the X25519 shared secret), the ML-KEM component still protects the session shared secret and traffic keys.	X25519 shared secret (during key exchange)	Holds	medium	yes
A08.11	**selected_group and transcript binding.** The selected_group and transcript (message_hash(CH1), HRR, CH2) accepted by the client are bound, corresponding to its CH1 offer and the HRR contents.	no compromise	Holds	low	yes
R02.12	**replay of HRR or ClientHello.** The same HRR or CH message instance is accepted at most once within a single connection; an HRR spliced across connections cannot be accepted as a valid handshake.	no compromise	Holds	medium	yes
A03.13	**server run and application key agreement.** When the client accepts completion of the handshake, there is a real server run whose identity, negotiated group and key shares match; both sides derive the same application traffic keys.	no compromise	Holds	medium	yes
X01	**no completion after a second HelloRetryRequest.** A client never completes a handshake after it has received a second HelloRetryRequest in the same connection; receiving one forces an abort.	no compromise	Holds	low	yes
X02	**downgrade-free parameter agreement.** The negotiated group, cipher suite and TLS version accepted by the client are the same as if no attacker were present, and match what the peers agreed.	no compromise	Holds	low	yes
X03	**strong secrecy of the session shared secret.** The attacker cannot tell which of two allowed values the session shared secret takes: the shared secret is indistinguishable, not merely unrecoverable.	no compromise	Reachable only	none	no

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

Properties of the catalogue that the documents do not speak of, added by the analyst (Appendix D): X03 (K03).

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: HRR selected group bound to the CH1/CH2 transcript	In the model the client completes only when the HRR selects X25519MLKEM768 and its cookie and transcript are bound to its own ClientHello1, so an attacker cannot swap the selected group or splice a foreign HRR into the run.	A08.11, A08, R01
Q2: a replayed/cross-connection spliced HRR cannot lead to wrong authentication	A repeated or cross-connection HRR is not accepted: client completions map injectively to server HRRs and only to an HRR of the same session, so replay does not produce a second accepted handshake.	R02.12, R02, A04
Q3: server authentication and agreement of both parties' application keys	When the client accepts completion, a real server ran with matching key shares, and both sides derived the same application traffic key; the server signing key is not derivable by the network attacker.	A03.13, A03, K01

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:

- The model is symbolic: it does not prove the computational security of ML-KEM-768 or X25519, and a symbolic proof is a proof about the abstraction.
- X25519 is modelled only with the law g^xy=g^yx; low-order points and the all-zero shared-secret check required by main s0018 and described in rfc7748 P10 cannot be represented and are not analysed.
- The record layer is not modelled, so payload confidentiality, AEAD nonce uniqueness, server-certificate confidentiality and endpoint identity protection are outside the analysis.
- The profile fixes a single group, cipher suite and version, so the group/cipher/version binding and downgrade checks are vacuous; the model has no second group for an attacker to substitute.
- 0-RTT, PSK, resumption and KeyUpdate are not modelled; the RNG-quality and variable-length timing issues in the security considerations are not trace properties.
- Certificate-chain validation is out of scope (rfc9846 s0394); the client is pinned to one server public key.
- Strong secrecy of the shared secret (X03) is undecided: the symbolic KEM and one global chosen secret make an attacker-as-client equality test possible, which is a model artifact.
- No value kept and changed across steps (counter, sequence number, replaced key) is relied on by a target, so no unrolled bounded model and no state review were produced.

What the host records:

- No answer for: X03.
- The answers changed after the reading was written for: X03.
- Set by the analyst and without a proof or an attack (Appendix D): X03 under `network`.

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.

A03: server authentication after HRR

Whenever a client accepts handshake completion after the HRR flow, there exists a real server run whose identity, negotiated group X25519MLKEM768, both key shares, and transcript match the values the client accepted.

The client's completion is tied to a server that sent the Finished over the same session, and the accepted group is the offered X25519MLKEM768. A network attacker cannot make the client accept without such a server run.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state peer authentication as a TLS goal.
- Stated in the documents: yes. Where the documents say it: rfc9846:6490
- Rests on: The KEM and X25519 are symbolic primitives; only g^xy=g^yx is modelled.; The transcript is a nested hash chain and message_hash(ClientHello1) is represented by hashing ClientHello1.; There is one server identity, fixed by the public key the client pins.
- Next step: None for this target; the computational strength of the primitives is out of scope.

A04: no double acceptance of a server run

Every time a client successfully completes the handshake after the HRR flow, it corresponds to a distinct real server run; the same server run cannot be accepted twice by the client.

Client completions and server HRRs are in injective correspondence, so one server run cannot be accepted twice and a replayed HRR does not create a second session. This rests on the cookie binding the hash of ClientHello1.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents require the cookie to bind ClientHello1 and forbid a second HRR.
- Stated in the documents: yes. Where the documents say it: s0126, s0167, s0168
- Rests on: The cookie is modelled as a MAC over the ClientHello1 hash with a server key.; The unicity of the transcript hash chain makes distinct sessions distinct.
- Next step: None; the concrete cookie integrity algorithm is unspecified in the source.

A08: HRR parameters bound to transcript

The selected_group, cipher suite, and selected_version accepted by the client are bound to the transcript and all correspond to its CH1 proposal, the HRR contents, and the message_hash(CH1) reconstruction; an attacker cannot substitute unauthenticated values.

The client accepts only selected_group X25519MLKEM768 and only an HRR of its own session; cipher suite and selected_version are fixed constants in this profile. The binding therefore holds, though vacuously for the fixed single-group profile.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: The documents state the selected_group and version checks explicitly.

- Stated in the documents: yes. Where the documents say it: s0057, s0058, s0247, s0251
- Rests on: Only one group exists in the model, so no alternative selected_group can be tested.; Cipher suite and selected_version are constants, not negotiated values.
- Next step: Model a second group if a non-vacuous downgrade result is wanted.

K01: secrecy of shared secret and keys

The attacker cannot derive the X25519MLKEM768 shared secret, the handshake traffic keys, the application traffic keys, or the server's long-term signing private key.

Neither the hybrid shared secret and the keys derived from it nor the server signing key can be learned by the network attacker during the handshake. The controls show that revealing both ephemeral components (or the signing key after the run) does reveal them, so the proof is not vacuous.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state that the hybrid shared secret is secure while at least one component is unbroken.
- Stated in the documents: yes. Where the documents say it: rfc9954:550
- Rests on: The KEM is modelled with decaps(encaps(pk(sk),r),sk)=r.; The key schedule is collapsed to a one-way KDF and a Finished key function.; The signing key is a private free name and is revealed only in phase 2.
- Next step: A computational argument is needed for the primitive strength, which the symbolic model cannot give.

K04: key separation and freshness

The handshake/application traffic keys derived by the client and the server are separated by direction and purpose, and keys from different sessions are not unexpectedly identical.

Handshake secrets are unique per session: the session identifier uniquely determines the handshake secret, and holding one session's keys does not make another session's secret shared. This follows from the transcript-bound KDF.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state uniqueness of session keys.
- Stated in the documents: yes. Where the documents say it: rfc9846:6495
- Rests on: The KDF is one-way and injective on its inputs in the symbolic model.; Distinct sessions use fresh nonces and fresh ephemeral shares.
- Next step: None.

R01: cookie bound to current CH1

The HRR and its cookie accepted by the client must correspond to the current CH1; an old HRR/cookie cannot be replayed as valid authentication for this run.

The client accepts an HRR only when it belongs to the same server session as its ClientHello1, so a stale or foreign cookie does not authenticate a run. Cookie freshness is enforced by binding the ClientHello1 hash.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents describe the cookie as binding the ClientHello1 hash for a stateless HRR.
- Stated in the documents: yes. Where the documents say it: s0167, s0168

- Rests on: The cookie is a MAC over the ClientHello1 hash.; The server is modelled statelessly, as the documents describe.
- Next step: None; the concrete cookie encoding is unspecified.

R02: no replay of HRR or ClientHello

The same HRR or CH message instance is accepted at most once within a single connection; an HRR spliced across connections cannot be accepted as a valid handshake.

Repeated HRRs do not yield a second accepted run, and a spliced cross-connection HRR is rejected because the client completes only with an HRR of its own session. Completions injectively match server HRRs.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents require abort on a second HRR and bind the retry to ClientHello1.
- Stated in the documents: yes. Where the documents say it: s0080, s0126
- Rests on: The client reads a single HRR; there is no second-HRR acceptance path.; The transcript binds the HRR to ClientHello1 through message_hash.
- Next step: None.

C01: forward secrecy of session keys

If the server's long-term signing private key leaks after a session completes, the attacker still cannot obtain the handshake or application traffic keys of previously completed sessions.

Leaking the server signing key after the handshake does not reveal past session keys, because the shared secret is built from ephemeral ML-KEM and X25519 material. The control shows that revealing both ephemeral components does reveal the secret.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state forward secrecy with respect to long-term keys.
- Stated in the documents: yes. Where the documents say it: rfc9846:6504
- Rests on: Erased ephemeral material is modelled by phase separation.; The KEM and DH are symbolic; the DH law is only g^xy=g^yx.
- Next step: Computational forward secrecy is not proved by the symbolic model.

C04: hybrid security under one bad component

If either the X25519 ephemeral private key share or the ML-KEM private key share leaks, the attacker still cannot obtain the session shared secret and traffic keys.

Leakage of either ephemeral component alone leaves the shared secret protected by the other component, and only leaking both reveals it (the control). This is the hybrid property for the two private-key compromises.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state the hybrid secret is secure while at least one component is unbroken.
- Stated in the documents: yes. Where the documents say it: rfc9954:550
- Rests on: The KEM shared secret equals the encapsulation randomness in the model.; The two components are combined with a one-way concatenation KDF.
- Next step: Computational hybrid security needs a computational argument.

C08: ML-KEM protects when X25519 breaks

When the X25519 primitive fails (the attacker obtains the X25519 shared secret), the ML-KEM component still protects the session shared secret and traffic keys.

With the X25519 component broken (its shared secret known to the attacker), ML-KEM-768 still protects the session secret and traffic keys. The model uses only the single Diffie-Hellman law, so this is a lower bound on the hybrid protection.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state the hybrid secret remains secure while one component is unbroken.
- Stated in the documents: yes. Where the documents say it: rfc9954:550
- Rests on: X25519 is broken by giving the attacker the client scalar, which lets it compute the X25519 shared secret.; The KEM remains one-way in the symbolic model.
- Next step: Computational ML-KEM security is out of scope.

A08.11: selected_group and transcript binding

The selected_group accepted by the client is bound to the transcript (message_hash(CH1), HRR, CH2) and corresponds to its CH1 proposal and the HRR contents.

The accepted selected_group is X25519MLKEM768, matching the ClientHello1 offer, and the transcript is bound to the HRR of the same session. Because the profile has one group, the group part is vacuous but the transcript binding is real.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: The documents state the selected_group and transcript reconstruction requirements.
- Stated in the documents: yes. Where the documents say it: s0057, s0058, s0247, s0251
- Rests on: Single fixed group.; Transcript as a nested hash chain with message_hash(ClientHello1).
- Next step: Add a second group for a non-vacuous test.

R02.12: replay of HRR or ClientHello

The same HRR or CH message instance is accepted at most once within a single connection; an HRR spliced across connections must not be accepted as a valid handshake.

A repeated HRR or ClientHello instance yields at most one accepted run, and a spliced cross-connection HRR is not accepted. This is the replay requirement on the retry.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents require abort on a second HRR and forbid post-handshake ClientHello reuse.
- Stated in the documents: yes. Where the documents say it: s0080, s0087, s0126
- Rests on: Single-read client.; Cookie binds ClientHello1 hash.
- Next step: None.

A03.13: server run and application key agreement

When the client accepts completion, a real server run exists whose identity, negotiated group and key shares match; the application traffic keys derived by both sides are the same.

When the client accepts completion, a real server ran with matching key shares, and both sides derive the same application traffic key. The application key is tied to the shared transcript-bound handshake secret.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents state peer authentication and show that both sides derive the application secrets from the shared secret.
- Stated in the documents: yes. Where the documents say it: rfc9846:6490
- Rests on: Finished and CertificateVerify are symbolic MAC and signature over the transcript hash.; One fixed server identity.
- Next step: None.

X01: no completion after a second HelloRetryRequest

A client never completes a handshake after it has received a second HelloRetryRequest in the same connection; receiving one forces an abort.

The client processes exactly one HRR and only completes after it, so there is no path that accepts a second HRR. In this single-retry profile the property holds structurally, because the client never reads a second HRR, rather than by a counter.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: The documents require the client to abort on a second HRR.
- Stated in the documents: yes. Where the documents say it: s0126
- Rests on: The client is written as a single-read process; a second-HRR path is not constructed.; Group is fixed.
- Next step: Model an explicit retry counter if a mechanical second-HRR test is wanted.

X02: downgrade-free parameter agreement

The negotiated group, cipher suite and TLS version accepted by the client are the same as if no attacker were present, and match what the peers agreed.

The accepted group is X25519MLKEM768 and cannot be replaced by an attacker-chosen HRR, and the client completes only with an HRR matching its own session. Cipher suite and version are fixed constants, so downgrade is vacuous in this profile.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: The documents state downgrade protection.
- Stated in the documents: yes. Where the documents say it: rfc9846:6499
- Rests on: Single fixed group, cipher suite and version.
- Next step: Model multiple suites/versions for a non-vacuous downgrade test.

X03: strong secrecy of the session shared secret

The attacker cannot tell which of two allowed values the session shared secret takes: the shared secret is indistinguishable, not merely unrecoverable.

The answers of this target changed after this reading was written.

Strong secrecy (indistinguishability) of the shared secret was not decided. ProVerif's non-interference fails only because the dedicated copy uses one global chosen secret and a public KEM encapsulation, so an attacker acting as a client can decapsulate it and test a candidate value. That is a model artifact, not a protocol weakness, and nothing can be concluded about per-session strong secrecy from this run.

- Result: Reachable only; answers what is asked: undetermined; not usable
- Value none: The documents state secrecy and the hybrid property but not indistinguishability of the shared secret; the run decides nothing either way.
- Stated in the documents: no
- Rests on: A single global chosen secret instead of a per-session one.; The symbolic KEM lets the encapsulation key holder recover the chosen secret.
- Open: Whether the per-session shared secret is indistinguishable from another allowed value.
- Next step: Tamarin+ Agent with per-session secrets, or a non-interference model that hides the candidate values under a private test key.

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
A03	proved		auth	checked		2 proved
A04	proved		auth_inj	checked		2 proved
A08	proved		group	checked		2 proved
K01	proved		ss_secret	checked		2 proved
K04	proved		key_sep	checked		2 proved
R01	proved		hrr_bound	checked		1 proved
R02	proved		hrr_inj	checked		2 proved
C01	proved		fs	checked		1 proved
C04	proved		x25519_hybrid	checked		2 proved
C08	proved		x25519_hybrid	checked		1 proved
A08.11	proved		group	checked		2 proved
R02.12	proved		hrr_inj	checked		2 proved
A03.13	proved		auth	checked		2 proved
X01	proved		x01	checked		1 proved
X02	proved		group	checked		2 proved
X03	reachable		ss_reach		strong (model_artifact)	1 attack

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
A03	When the client accepts the handshake after the HRR, a real server run exists that sent the Finished for the same session; the attacker has the network only.	`network`	proved	auth	checked
A03	The negotiated group accepted is X25519MLKEM768 as offered in CH1, so the server identity, group and shares match; the attacker has the network only.	`network`	proved	group	checked
A04	Each client completion maps to a distinct real server run: a server Finished cannot be accepted twice; the attacker has the network only.	`network`	proved	auth_inj	checked
A04	A given HRR of a server run yields at most one accepted client completion (injective agreement on the HRR); the attacker has the network only.	`network`	proved	hrr_inj	checked

target	claim	attacker	answer	query	control
A08	The accepted selected_group is X25519MLKEM768 (the cipher suite and selected_version are fixed constants in this profile), so the attacker cannot substitute a group that was not offered; the attacker has the network only.	`network`	proved	group	checked
A08	The client completes only with an HRR of the same session, which binds message_hash(CH1), the HRR and CH2 into the transcript; the attacker has the network only.	`network`	proved	hrr_bound	checked
K01	The X25519MLKEM768 shared secret and the handshake and application traffic keys derived from it cannot be learned by the network attacker during the handshake; the attacker has the network only.	`network`	proved	ss_secret	checked
K01	The server long-term signing private key cannot be learned by the network attacker; the attacker has the network only.	`network`	proved	sk_secret	checked
K04	Distinct sessions produce distinct handshake secrets: the session identifier uniquely determines the handshake secret, so different sessions do not share keys; the attacker has the network only.	`network`	proved	key_sep	checked
K04	Even when the attacker holds the shared secret and traffic keys of one completed session, another session's handshake secret is distinct and not shared; the attacker has one session's keys beyond the network.	`one_session_key_leak`	proved	key_sep	checked
R01	An HRR or cookie accepted by the client corresponds to the current CH1: the client accepts only with an HRR from the server session bound to its own ClientHello; the attacker has the network only.	`network`	proved	hrr_bound	checked
R02	A repeated HRR does not yield a second accepted run: client completions map injectively to server HRRs; the attacker has the network only.	`network`	proved	hrr_inj	checked
R02	A cross-connection spliced HRR is not accepted: the client completes only with an HRR of its own session; the attacker has the network only.	`network`	proved	hrr_bound	checked
C01	After the handshake completes, leaking the server long-term signing private key does not reveal the past session shared secret or the handshake and application traffic keys; the attacker has that signing key beyond the network.	`server_sig_key_leak`	proved	fs	checked
C04	Leakage of the X25519 ephemeral private scalar alone does not reveal the session shared secret or traffic keys; ML-KEM still protects them; the attacker has that scalar beyond the network.	`x25519_sk_leak`	proved	x25519_hybrid	checked
C04	Leakage of the ML-KEM-768 decapsulation private key alone does not reveal the session shared secret or traffic keys; X25519 still protects them; the attacker has that key beyond the network.	`mlkem_sk_leak`	proved	mlkem_hybrid	checked

target	claim	attacker	answer	query	control
C08	When the X25519 component is broken (the attacker obtains the X25519 shared secret), ML-KEM-768 still protects the session shared secret and traffic keys; the attacker has the X25519 shared secret beyond the network.	`x25519_broken`	proved	x25519_hybrid	checked
A08.11	The accepted selected_group is X25519MLKEM768, matching the CH1 offer; the attacker has the network only.	`network`	proved	group	checked
A08.11	The transcript is bound: the client completes only with an HRR of the same session, so message_hash(CH1), the HRR and CH2 cannot be replaced; the attacker has the network only.	`network`	proved	hrr_bound	checked
R02.12	A repeated HRR/CH message instance yields at most one accepted run (injective agreement on the HRR); the attacker has the network only.	`network`	proved	hrr_inj	checked
R02.12	A spliced cross-connection HRR is not accepted: the client completes only with an HRR bound to its own session; the attacker has the network only.	`network`	proved	hrr_bound	checked
A03.13	When the client accepts completion, a real server ran with matching identity and key shares (server Finished for the same session); the attacker has the network only.	`network`	proved	auth	checked
A03.13	Both sides derive the same application traffic key: whatever application key the client holds, the server has derived the same one; the attacker has the network only.	`network`	proved	appkey	checked
X01	A client that completes has processed only the single HRR with the agreed group; there is no completion after a second HRR in the modelled single-retry flow; the attacker has the network only.	`network`	proved	x01	checked
X02	The accepted group and parameters are X25519MLKEM768 and cannot be downgraded by an active attacker; the attacker has the network only.	`network`	proved	group	checked
X02	The client completes only with an HRR matching its own session, so the parameters cannot be replaced by an attacker-chosen HRR; the attacker has the network only.	`network`	proved	hrr_bound	checked
X03	The attacker cannot tell which of two allowed values the session shared secret takes (strong secrecy/indistinguishability); the symbolic KEM model does not decide this.	`network`	attack (model_artifact)	strong	

The prover never says that a property of this kind (the attacker cannot tell which of two values is used) is false. It gave no proof, along with a trace on which the attacker tells the values apart, and the attack is recorded on that trace: strong.

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-20261115T080740, 2026-11-15. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 50, prover 47 s, wall 13 min, cost 0.118 USD.

ProVerif: 109 proofs run, 107 decided, 2 without an answer, 31 s, 48 answers recorded.

What was analysed

Protocol	TLS hybrid HelloRetryRequest profile
Version	rfc10024
Analysed	Fixed X25519MLKEM768, full server authentication after a single HelloRetryRequest; transcript reconstruction preserving message_hash(CH1), HRR and CH2, cookie/selected_group checks and rejection of a second HRR are retained. The HRR process is not treated as a successful unauthenticated public-key substitution.
Not analysed	The symbolic model does not prove the computational or post-quantum security of the algorithms; attack traces hold only under explicit assumptions and are not directly equivalent to specification vulnerabilities. Each property must have a reachability/positive control.
Source text	541 numbered passages (142415 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	4	Negotiated Groups
main	4.1	Client Share
main	4.2	Server Share
main	4.3	Shared Secret
rfc9954	3	Construction for Hybrid Key Exchange
rfc9954	3.1	Negotiation
rfc9954	3.2	Transmitting Public Keys and Ciphertexts
rfc9954	3.3	Shared Secret Calculation
rfc9846	4	Handshake Protocol
rfc9846	4.1	The Transcript Hash
rfc9846	4.2	Key Exchange Messages
rfc9846	4.2.1	Cryptographic Negotiation
rfc9846	4.2.2	Client Hello
rfc9846	4.2.3	Server Hello
rfc9846	4.2.4	Hello Retry Request
rfc9846	4.3	Extensions

Document	Section	Title
rfc9846	4.3.1	Supported Versions
rfc9846	4.3.2	Cookie
rfc9846	4.3.3	Signature Algorithms
rfc9846	4.3.4	Certificate Authorities
rfc9846	4.3.5	OID Filters
rfc9846	4.3.6	Post-Handshake Certificate-Based Client Authentication
rfc9846	4.3.7	Supported Groups
rfc9846	4.3.8	Key Share
rfc9846	4.3.8.1	Diffie-Hellman Parameters
rfc9846	4.3.8.2	ECDHE Parameters
rfc9846	4.3.9	Pre-Shared Key Exchange Modes
rfc9846	4.3.10	Early Data Indication
rfc9846	4.3.11	Pre-Shared Key Extension
rfc9846	4.3.11.1	Ticket Age
rfc9846	4.3.11.2	PSK Binder
rfc9846	4.3.11.3	Processing Order
rfc9846	4.4	Server Parameters
rfc9846	4.4.1	Encrypted Extensions
rfc9846	4.4.2	Certificate Request
rfc9846	4.5	Authentication Messages
rfc9846	4.5.1	Certificate
rfc9846	4.5.1.1	OCSP Status and SCT Extensions
rfc9846	4.5.1.2	Certificate Selection
rfc9846	4.5.1.3	Receiving a Certificate Message
rfc9846	4.5.2	Certificate Verify
rfc9846	4.5.3	Finished
rfc9846	4.6	End of Early Data
rfc9846	4.7	Post-Handshake Messages
rfc9846	4.7.1	New Session Ticket Message
rfc9846	4.7.2	Post-Handshake Authentication
rfc9846	4.7.3	Key and Initialization Vector Update
rfc9846	7	Cryptographic Computations
rfc9846	7.1	Key Schedule
rfc9846	7.2	Updating Traffic Secrets
rfc9846	7.3	Traffic Key Calculation
rfc9846	7.4	Asymmetric Shared Secret Calculation

Document	Section	Title
rfc9846	7.4.1	Finite Field Diffie-Hellman
rfc9846	7.4.2	Elliptic Curve Diffie-Hellman
rfc9846	7.5	Exporters

The purpose of the protocol as the source step located it in the documents: fix X25519MLKEM768, complete server authentication after a single HelloRetryRequest; preserve the transcript reconstruction of message_hash(CH1), HRR, CH2, the cookie/selected_group checks, and rejection of a second HRR. Do not treat the HRR process as a successful unauthenticated public key replacement.

The properties against the documents

Before writing a query, 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
A03	server authentication after HRR	The client is assured that, if it completes the handshake after a HelloRetryRequest, a real server ran with matching identity, negotiated group X25519MLKEM768, matching key shares and matching transcript.	RFC 9846 states peer authentication: the client's view of the peer identity should reflect the server's identity, and the signature-based modes authenticate the server over the handshake transcript.	rfc9846:6490, s0345, s0401, s0427, s0437
A04	no double acceptance of a server run	Each client completion corresponds to a distinct real server run; one server run cannot be accepted twice (HRR/cookie/CH1 binding).	RFC 9846 requires the HRR/cookie to bind the retried ClientHello to ClientHello1 and the client to abort on a second HRR; the cookie binds the hash of ClientHello1.	s0080, s0126, s0167, s0168, s0247, s0251
A08	HRR parameters bound to transcript	The client's accepted selected_group, cipher suite, selected_version and transcript (message_hash(CH1), HRR, CH2) match its CH1 offer and the HRR, and cannot be replaced by unauthenticated values.	RFC 9846 requires checks of selected_group against supported_groups and key_share, retention of selected_version and cipher suite, and reconstructs the transcript from message_hash(CH1).	rfc9846:1074, s0057, s0058, s0126, s0131, s0133, s0247, s0251
K01	secrecy of shared secret and keys	The X25519MLKEM768 shared secret, handshake and application traffic keys, and the server long-term signing private key are kept from the network attacker.	RFC 9954 states the hybrid shared secret is secure as long as at least one component is unbroken; RFC 9846 key schedule derives traffic keys from it.	rfc9954:550, s0015, s0044, s0045, s0522
K04	key separation and freshness	Client and server handshake/application traffic secrets are separated by direction and use, and distinct sessions do not share keys.	RFC 9846 states uniqueness of session keys: distinct handshakes should produce distinct, unrelated session keys and individual keys should be distinct and independent.	rfc9846:6495, s0503, s0522
R01	cookie bound to current CH1	An HRR and its cookie accepted by the client must correspond to the current ClientHello1; an old HRR/cookie is not accepted as this run's authentication.	RFC 9846 says the cookie lets the server bind the hash of ClientHello1 for a stateless HRR.	s0166, s0167, s0168

ID	Name	About	What the documents say	Where
R02	no replay of HRR or ClientHello	Within a connection a given HRR or ClientHello instance is accepted at most once; a cross-connection spliced HRR is not accepted as a valid handshake.	RFC 9846 requires the client to abort on a second HRR and forbids reuse of a ClientHello after the handshake; replay of 1-RTT handshake is limited by the server Random.	s0080, s0087, s0126
C01	forward secrecy of session keys	After the handshake completes, leakage of the server long-term signing private key does not reveal the session handshake or application traffic keys.	RFC 9846 states forward secrecy with respect to long-term keys: compromise of the signature keys after the handshake does not compromise the session key, as long as ephemeral material is erased.	rfc9846:6504, s0015, s0045
C04	hybrid security under one bad component	Leakage of the X25519 ephemeral private share, or of the ML-KEM private share, still leaves the session shared secret and traffic keys secret.	RFC 9954 states the hybrid secret is secure as long as at least one component key exchange algorithm is unbroken.	rfc9954:550, s0015, s0019, s0041
C08	ML-KEM protects when X25519 breaks	When the X25519 primitive is broken (the attacker obtains the X25519 shared secret), the ML-KEM component still protects the session shared secret and traffic keys.	RFC 9954 states the hybrid secret remains secure as long as at least one component is unbroken; RFC 10024 fixes X25519MLKEM768 as ML-KEM then X25519.	rfc9954:550, s0019, s0041, s0047
A08.11	selected_group and transcript binding	The client's accepted selected_group and transcript bind to its CH1 offer and the HRR content.	RFC 9846 requires the client to verify selected_group came from its supported_groups and not from its key_share, and to reconstruct the transcript with message_hash(CH1).	rfc9846:1074, s0057, s0058, s0247, s0251
R02.12	replay of HRR or ClientHello	A repeated HRR or a spliced cross-connection HRR is not accepted as a valid handshake; a message instance is accepted at most once.	RFC 9846 requires abort on a second HRR and forbids post-handshake ClientHello reuse.	s0080, s0087, s0126
A03.13	server run and application key agreement	When the client accepts completion, a real server ran with matching identity, group and key shares, and both sides derive the same application traffic keys.	RFC 9846 peer authentication and authentication messages: CertificateVerify signs the transcript and Finished MACs it; both sides derive the same application secrets from the shared secret.	rfc9846:6490, s0345, s0401, s0427, s0437, s0503
X01	no completion after a second HelloRetryRequest	The client and the connection it is setting up; the second HRR is a message the attacker might replay or splice.	RFC 9846 requires the client to abort with unexpected_message if it receives a second HelloRetryRequest in the same connection.	s0126
X02	downgrade-free parameter agreement	The client and server cryptographic parameters; protected from an active network attacker.	RFC 9846 states downgrade protection: the cryptographic parameters should be the same on both sides and the same as without an attack.	rfc9846:6499
X03	strong secrecy of the session shared secret	The X25519MLKEM768 shared secret and the traffic keys derived from it, hidden from the network attacker.	Catalogue property K03 (strong secrecy / indistinguishability); the state is a goal the prover is asked by non-interference.	

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

The security statements of the documents, paragraph by paragraph

The host listed the 20 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: 3. Not taken: 17. 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
P01	main:268-271	A03, A03.13, A08, A08.11, `network`	
P02	main:273-276	The prover cannot be asked	Side-channel guidance to implementers; not a trace property.
P03	main:278-280	The prover cannot be asked	Fixed-length assumption about the primitives, not a protocol trace property.
P04	main:282-287	The prover cannot be asked	Property of ML-KEM encapsulation randomness, outside the symbolic message flow.
P05	main:289-297	The prover cannot be asked	Attack on an insecure RNG; randomness quality is outside the symbolic model.
P06	main:299-306	C08, `x25519_broken`	
P07	main:308-310	The prover cannot be asked	Insecure-RNG interaction between the components; outside the symbolic model.
P08	rfc7748:783-793	The prover cannot be asked	Strength of curve25519 (about 128-bit level); primitive strength cannot be asked.
P09	rfc7748:802-808	The prover cannot be asked	curve448 strength trade-off; primitive strength cannot be asked.
P10	rfc7748:810-818	The prover cannot be asked	Contributory behaviour / small-order points: the symbolic DH law g^xy=g^yx cannot represent low-order points, so the all-zero shared-secret check (main s0018) cannot be asked.
P11	rfc7748:820-825	No goal stated	Advice to designers about using public keys as identifiers; no protocol goal stated.
P12	rfc7748:827-837	The prover cannot be asked	Implementation distinguishability and timing; outside the symbolic model.
P13	rfc9846:4676-4677	No goal stated	Pointer to Appendix C, E, F; states no goal itself.
P14	rfc9954:550-558	C04, C08, `x25519_broken`, `x25519_sk_leak`, `mlkem_sk_leak`	
P15	rfc9954:560-569	The prover cannot be asked	Requirement that the KEM be IND-CCA2/FO; a property of the primitive, not of the handshake traces.
P16	rfc9954:571-574	The prover cannot be asked	Constant-length assumption about keys, ciphertexts and secrets.
P17	rfc9954:576-582	The prover cannot be asked	Variable-length timing side channel; not a trace property.
P18	rfc9954:584-586	The prover cannot be asked	Variable-length secret with non-collision-resistant hash; cryptographic assumption.
P19	rfc9954:588-592	No goal stated	Design guidance: if not fixed length, a length prefix would be needed; no goal for this fixed-length profile.
P20	rfc9954:594-597	The prover cannot be asked	Requirement to use only fixed-length shared secrets; an assumption on the primitive.

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: 13. Left out: 19. That the documents do not speak of a property is not a reason the analyst may give.

Entry	Property of the catalogue	Examined as	Why not
K02	Payload and end-to-end confidentiality	Outside the scope	The task models the handshake; TLS record-layer payload confidentiality is not modelled.
K03	Strong secrecy / indistinguishability of secret values	X03	
D01	Message integrity and data-origin authenticity	A03, A08	
A01	Peer participation (Aliveness)	A03, A03.13	
A02	Weak consistency authentication	A03	
A05	Implicit key authentication	A03, K01	
A06	Explicit key confirmation	A03.13	
A07	Identity–credential–key binding	A03	
R03	Business operation at most once	Not applicable	No business operation, billing or consumable token in the handshake.
R04	Message acceptance order / window security	R02	
R05	Resistance to downgrade / security policy preservation	X02	
C02	Resistance to key-compromise impersonation (KCI)	Not applicable	This profile authenticates only the server; the client has no long-term credential, so there is no client key-compromise impersonation goal. Server key compromise is the server's identity itself, not KCI.
C03	Known-key security / compromise isolation	K04	
C05	Post-compromise security (PCS)	Outside the scope	No ratchet or rekey in the main handshake; KeyUpdate (s0473) is outside the modelled flow.
C06	Enhanced authentication after compromise of enrolment credentials	Not applicable	No PAKE or registration record.
C07	Resistance to offline dictionary attacks	Not applicable	No password or guessable secret.
Z01	Correctness of authorization enforcement	Not applicable	No authorization decision in the handshake.
Z02	No expansion of delegated authority	Not applicable	No delegation or proxy.
Z03	Binding of authorization to identity / session / operation	Not applicable	No authorization token or service session.
Z04	Revocation and expiry take effect	Not applicable	No revocation or expiry mechanism.
Z05	Security isolation across tenants / services	Not applicable	No tenants or shared services.
S01	Uniqueness of encryption nonces	Outside the scope	AEAD nonces are a record-layer concern; the task models handshake messages.

Entry	Property of the catalogue	Examined as	Why not
S02	Security of retransmission / key reinstallation	R02	
S03	Resistance to rollback and state monotonicity	R02, X02	
S04	Consistency of security context across phases	A08	
S05	Termination and invalidation of old state	Outside the scope	No teardown, deauthentication or logout in the modelled handshake.
P01	Identity anonymity	Outside the scope	Identity confidentiality rests on record-layer encryption of the Certificate message, which the handshake model does not include.
P02	Session unlinkability	Outside the scope	Session unlinkability rests on record-layer behaviour not modelled.
P03	Hiding of user / credential existence	Not applicable	No login, discovery or registration query.
V01	Conditional progress / error recovery	The prover cannot be asked	Conditional progress/error recovery cannot be asked of a protocol model (terminality).
V02	Absence of unexpected deadlock	The prover cannot be asked	Absence of deadlock cannot be asked (terminality).
V03	Abuse of limited resources / resistance to amplification	The prover cannot be asked	Resource cost and amplification bounds cannot be asked beyond counting events.

The parties and the protected values, one by one

The host listed the 27 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: 12. Without: 15. 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	Client	party	Nothing gained	The client holds only its own ephemeral material and no long-term credential; the network attacker can already act as a client.
S02	Server	party	`server_sig_key_leak`	
S03	Server identity	value	Nothing gained	The server certificate and identity are public.
S04	Negotiated group and key shares	value	Nothing gained	The negotiated group and key shares are sent on the wire.
S05	Handshake transcript	value	Nothing gained	The handshake transcript is public.
S06	Binding of HRR to CH1/CH2	value	Nothing gained	The HRR/CH binding values are public.
S07	cookie	value	Nothing gained	The cookie value is sent in the clear; only its integrity is protected.
S08	selected_group	value	Nothing gained	selected_group is public.
S09	cipher suite	value	Nothing gained	The cipher suite is public.
S10	selected_version	value	Nothing gained	selected_version is public.
S11	transcript (message_hash(CH1), HRR, CH2)	value	Nothing gained	The transcript is public.

ID	Party or value	Kind	Attacker	Why none
S12	X25519MLKEM768 shared secret	value	`one_session_key_leak`	
S13	Handshake traffic key	value	`one_session_key_leak`	
S14	Application traffic key	value	`one_session_key_leak`	
S15	Server long-term signing private key	value	`server_sig_key_leak`	
S16	client_handshake_traffic_secret	value	`one_session_key_leak`	
S17	server_handshake_traffic_secret	value	`one_session_key_leak`	
S18	client_application_traffic_secret	value	`one_session_key_leak`	
S19	server_application_traffic_secret	value	`one_session_key_leak`	
S20	CH1 hash	value	Nothing gained	Hash(ClientHello1) is public.
S21	HRR	value	Nothing gained	The HRR is public.
S22	CH1	value	Nothing gained	ClientHello1 is public.
S23	CH2	value	Nothing gained	ClientHello2 is public.
S24	Handshake/application traffic keys of past sessions	value	`one_session_key_leak`	
S25	Traffic key	value	`one_session_key_leak`	
S26	Session shared secret	value	`one_session_key_leak`	
S27	transcript	value	Nothing gained	The transcript is public.

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, modifies and sends messages, and holds no secret of a party		
`server_sig_key_leak`	the server's long-term certificate signing private key, obtained after the handshake is complete	RFC 9846 forward secrecy is stated against compromise of the long-term signing keys after the handshake.	rfc9846:6504
`x25519_sk_leak`	the client's X25519 ephemeral private scalar of one session, and the peer's public share, but not the ML-KEM private key	The RFC 9846 forward-secrecy discussion says the private keys corresponding to the key shares must be erased; RFC 9954 requires security if one component is unbroken.	rfc9954:550
`mlkem_sk_leak`	the client's ML-KEM-768 decapsulation private key of one session, but not the X25519 private scalar	RFC 9954 hybrid property: secure as long as at least one component is unbroken.	rfc9954:550
`x25519_broken`	the X25519 shared secret of the session (the X25519 component is broken), while ML-KEM remains secure	RFC 9954 hybrid property: the concatenated secret must stay secure when one component fails; RFC 7748 notes that a CRQC recovers the X25519 scalar.	rfc9954:550, rfc7748:783

Attacker	Has	Basis	Where
`one_session_key_leak`	the shared secret and the handshake/application traffic keys of one completed session, read from a party's store after the handshake	RFC 9846 states that session keys are unique and unrelated across distinct handshakes, so a key compromised in one session must not expose another.	rfc9846:6495

ID	`network`	`server_sig_key_leak`	`x25519_sk_leak`	`mlkem_sk_leak`	`x25519_broken`	`one_session_key_leak`
A03	Holds					
A04	Holds					
A08	Holds					
K01	Holds					
K04	Holds					Holds
R01	Holds					
R02	Holds					
C01		Holds				
C04			Holds	Holds		
C08					Holds	
A08.11	Holds					
R02.12	Holds					
A03.13	Holds					
X01	Holds					
X02	Holds					
X03	No answer					

Source and targets

Source: 541 units, 142415 bytes, frozen in source-delivery-01.json. Targets: 10 approved (A03, A04, A08, K01, K04, R01, R02, C01, C04, C08); 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 source package protocol-901b7b236d9f45b5e4ddfc180692

This step, as reported by itself: 1 requests (1 failed), 0.000 USD, 66836 tokens in and 4526 out, 28 s.

Notes on the prover and the model

What the analyst named for the targets:

step	targets	features	reason
34	K01, C01, C04, C08, X03, A03, A03.13	plain_primitives	The hybrid exchange is modelled with hash, MAC, KDF, signature and a symbolic KEM plus Diffie-Hellman under its single law g^xy=g^yx; no further algebraic law of ML-KEM or X25519 is modelled, so the secrecy and forward-secrecy answers rest on this plain-primitive abstraction.

step	targets	features	reason
34	A03, A04, A08, A08.11, R01, R02, R02.12, K01, K04, C01, C04, C08, X01, X02, A03.13	many_sessions	The client, server and compromise copies are replicated, so the authentication, binding, replay and key-separation answers are claimed for any number of sessions and parties.

The analyst's closing note

All 16 targets have answers. Fifteen are proved on model.pv with a control that falsifies a stronger or converse variant: server authentication after the HRR (A03, A03.13), injective agreement and no double acceptance (A04), HRR/selected_group/transcript binding (A08, A08.11, R01, R02, R02.12), single-retry/no-second-HRR and downgrade (X01, X02), key secrecy (K01), key separation under the network and a one-session-key leak (K04), forward secrecy under the server signing-key leak (C01), and the hybrid property under either ephemeral-key leak or a broken X25519 (C04, C08). X03 (strong secrecy/indistinguishability of the shared secret) is undecided: the symbolic KEM with one global chosen secret lets an attacker acting as a client decapsulate and test it, which is a model artifact; a per-session strong-secrecy check is a case for Tamarin+ Agent. No finding is new: every proved guarantee is stated in the documents (RFC 9846 peer authentication, uniqueness, forward secrecy, downgrade; RFC 9954 hybrid property), and every attack the documents name is excluded under the attacker they give. The analysis is symbolic (KEM, DH with only g^xy=g^yx, hash, MAC, signature), covers any number of sessions, and does not cover the record layer, primitive strength, the all-zero/small-order X25519 check, RNG quality, or the fixed single group making group/cipher/version binding vacuous. No changing-state target arose, so no bounded.pv and no state review were needed.