Security analysis of TLS hybrid client certificate profile

rfc10024 · ProVerif+ Agent report

Documents	Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3 (protocol definition); The Transport Layer Security (TLS) Protocol Version 1.3 (supporting)
Properties examined	14: 14 hold, 0 fail
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 profile fixes one key agreement, X25519MLKEM768, and certificate authentication in both directions for the TLS 1.3 handshake. A client and a server each hold a certificate and its signing key, and both trust one certificate authority. The client sends its ML-KEM encapsulation key and an X25519 share; the server answers with an ML-KEM ciphertext and its own X25519 share, and each side derives one secret from the two parts.

The server then proves its identity with a CertificateVerify signature over the transcript and with a Finished check value; the client checks both and replies with its own Certificate, CertificateVerify and Finished, and the server accepts the client's identity only after verifying the client Finished. When the exchange ends the client is sure of the server's identity and of the transcript, and the server is sure of the client's identity, the negotiated parameters and the shared session keys.

The parties:

- **Client**: holds its certificate and signing key; trusts the CA. Holds: CertC, skC, CA public key.
- **Server**: holds its certificate and signing key; trusts the CA. Holds: CertS, skS, CA public key.

Figure 1 shows the exchange as the documents specify it.

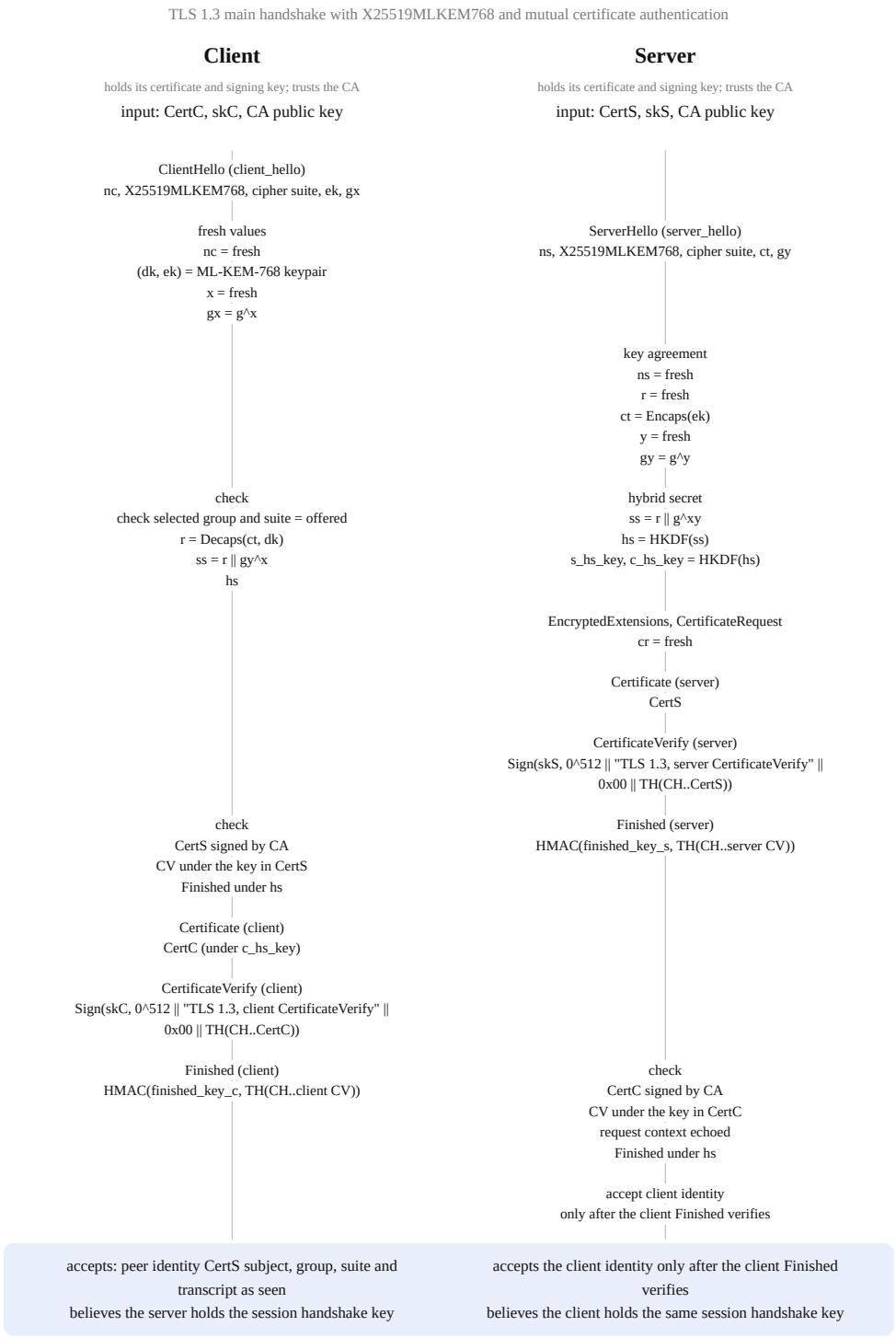


Figure 1. The protocol as specified.

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 modelled scope the profile does what it is meant to do: mutual authentication, agreement on identity, group, suite and transcript, injective agreement, key confirmation, role-tag separation, secrecy of the long-term and session keys, forward secrecy and hybrid security against one broken key-agreement component were all proved for any number of sessions. No finding is reported; every property examined is one the documents themselves require, and the analysis found no attack the documents do not already state. The proofs rest on idealised primitives (an opaque ML-KEM, X25519 with only g^xy = g^yx) and on a certificate path validation the model abstracts away. Two caveats limit the reach: several authentication proofs have no control, and in the leaked-server-key world the attacker was not also given the certificate that binds the key, because that world does not terminate in ProVerif.

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 of the peer.** The client did run against the client when accepting the service certificate and Finished; The client did run against the client when it accepted the client 's identity after verifying the client 's Finished.	no compromise	Holds	medium	yes
A03	**agreement on transcript and parameters.** Client acceptance is consistent with service-run on identity, consultative group, package and handshake The service accepts the client ' s identity at the same value as the client.	no compromise	Holds	medium	yes
A04	**uniqueness of each accepted handshake.** Each time a service provider accepts a client ' s identity and every time a client receives a service provider, it functions against a different peer protocol, and the same handshake evidence cannot be repeated.	no compromise	Holds	low	yes
A06	**Finished confirms the handshake keys.** The service does hold the key to the current session when the client validates the client Finished; The client does hold a handshake key for the same session when it validates the client Finished.	no compromise	Holds	low	yes
A07	**accepted identity matches the signing key holder.** The peer identity identified by the recipient is consistent with that of the subject of the peer certificate and the entity that actually holds the signature private key.	no compromise	Holds	medium	yes
A08	**negotiated parameters not substituted.** Groupes, packages, CertificateVerify role labels and transcripts accepted by the recipients correspond to this offer and the results of the consultations, and are not replaced or staggered.	no compromise	Holds	medium	yes
K01	**secrecy of long-term and traffic keys.** The long-term private signature key of the client and the service, as well as the handshake and application flow key derived from this session, may not be extrapolated except by the legitimate holder.	no compromise	Holds	medium	yes

ID	Property	Compromise assumed	Result	Value	In the documents
C01	**forward secrecy of past session keys.** Following completion of the session and removal of the secret of handshake, the leaking of the parties ' long-standing private signing keys does not reveal the flow keys of previous sessions.	Long-term private key signature between client and service (after session is completed and handshake secret removed)	Holds	low	yes
C02	**key compromise impersonation resistance.** Following the disclosure of the client ' s long-term private signature key, the attacker cannot pretend to be a service provider for the client ' s acceptance; After a long-term release of the private key signed by the service, the assailant cannot impersonate the client to make the service accept the client identity.	Long-term signature of the private key by the injured party (subject to the continued honest execution of the injured party)	Holds	medium	yes
C08	**hybrid security if one component fails.** Upon the expiration of part X25519 or part ML-KEM, where the shared secret was obtained by the person who was attacked, the confidentiality of both parties ' authentication and flow keys remains valid.	X25519orML-KEMSharing secrets (in handshake)	Holds	medium	yes
U01	**CertificateVerify role separation.** When the client and the service side accept the peer CertificateVerify each, the signature is generated by the peer to the current transcript with the peer role tag and cannot be reflected or mixed in direction.	no compromise	Holds	medium	yes
A06.12	**server accept implies client held the keys.** The client does hold the handshake key for this session when it validates the client Finished and accepts the client ' s identity.	no compromise	Holds	low	yes
C02.13	**one-sided long-term key leak.** When the client or service sign a long-term private key separately, the aggressor cannot pretend to be the undisclosed counterparty to be acceptable to the other party.	Long-term private key signed by client or service (single-end leak, other undiscovered)	Holds	medium	yes
X01	**client certificate concealed from the network.** The value of the client certificate does not appear in the data available to the assailant of the attacker who holds the network (readable and writes all handshake records); The client ' s identity is also protected in the presence of the active aggressor.	no compromise	Holds	low	yes

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

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: Non-reflective/mixed signature orientation for two-way identity authentication	The two directions cannot be mixed: each side signs a different context string over the transcript, and each verifies with the peer certificate's key, so no signature made in one role is ever accepted in the other; ProVerif proves this for any number of sessions.	U01, A02
Q2: Client certifier completion condition key binding	The server accepts the client identity only after the client Finished verifies, and that check value is an HMAC under the session handshake key over the transcript, so acceptance implies the client held the same handshake key; this is proved.	A06.12, A06
Q3: Check the impact of leaking only one end of a long-term authentication key separately	Leaking the client's long-term signing key does not let the attacker impersonate the server to that client, and leaking the server's key does not let it impersonate a client to the server; both are proved, the second for an attacker that holds the key but not the certificate that binds it.	C02, C02.13, 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 ML-KEM part is an opaque encapsulation: the model says nothing about computational or post-quantum security, ciphertext lengths, the encapsulation-key check or decapsulation failures.
- X25519 is modelled with the single law g^xy = g^yx; the all-zero and low-order checks are not modelled.
- Certificates are CA signatures over an identity and a key; path validation, revocation, name constraints and key usage are abstracted away, and both sides trust one CA.
- Only the client flight is protected by the record layer in the model; the server flight is plaintext, so endpoint-identity protection is asked for the client certificate only.
- One group and one cipher suite are modelled, as the profile fixes them; a downgrade to another group or suite cannot occur in the model.
- Post-handshake authentication, HelloRetryRequest, PSK, 0-RTT, KeyUpdate, session tickets and alerts are outside the modelled scope, so distinguishable failure behaviour is not analysed.
- In the compromise roles the ephemeral key-agreement values are shared by the sessions of one role instance, which makes the attacker stronger than in the protocol.
- In the leaked-server-key world the attacker holds the signing key but not the certificate that binds it to the server identity; giving it the certificate makes ProVerif's search not terminate, so that answer is for a slightly weaker attacker.
- Timing, availability and resource consumption are not askable of this model.

What the host records:

- Proved without a control query (Appendix C): A02: auth_client, A02: auth_server, A03: auth_client, A03: auth_server, A04: inj_client, A04: inj_server, A06: auth_client, A06: auth_server, A07: auth_client, A07: auth_server, A08: auth_client, A08: auth_server, K01: sess_secret, C01: sess_secret, C02: auth_client_x, C02: auth_server_leakedsrv, C08: kem_broken, C08: dh_broken, C08: auth_client, C08: auth_server_any, A06.12: auth_server, C02.13: auth_client_x, C02.13: auth_server_leakedsrv. 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 of the peer

The client did run against the client when accepting the service-end certificate and Finished; The client did run against the client when it accepted the client 's identity after verifying the client 's Finished.

Both directions hold. A client that accepts a server has talked to a server that really ran the handshake with the same transcript, and the server accepts a client identity only after a client with that certificate ran it. The client cannot learn that the server accepted its certificate, because TLS 1.3 sends no confirmation.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: a central guarantee of the profile, machine-checked for any number of sessions, but on idealised primitives and an abstracted PKI
- Stated in the documents: yes. Where the documents say it: rfc9846:6490
- Rests on: the client flight is treated as encrypted and the server flight as plaintext in the model; all certificates are CA-issued and path validation is abstracted; the KEM secret is recovered only with the decapsulation key; X25519 with only g^xy = g^yx
- Next step: none

A03: agreement on transcript and parameters

Client acceptance is consistent with service-run identities, consultative groups, packages and handshakes; The service accepts the client ' s identity at the same value as the client.

The client and the server end a run on the same identities, group, suite and transcript. Because both signatures and both Finished values cover the transcript hash, a party that changed any handshake message would have to produce a value that verifies under the peer's certificate key.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: this is the binding the profile relies on, proved mechanically; the fixed group means a downgrade to another group cannot arise in the model
- Stated in the documents: yes. Where the documents say it: rfc9846:6499
- Rests on: transcript modelled as a hash chain over the plaintext handshake messages; one fixed group and suite in the model
- Next step: none

A04: uniqueness of each accepted handshake

Each time a service provider accepts a client ' s identity and each time a client receives a service, it operates against a different peer protocol, and the same handshake evidence cannot be repeated.

Each acceptance corresponds to a distinct run of the peer: no handshake evidence can be accepted twice, in either direction. This rests on the fresh nonces both sides put into the transcript.

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

- Value low: uniqueness follows directly from the fresh nonces and the transcript binding; it confirms rather than adds
- Stated in the documents: yes. Where the documents say it: rfc9846:6495
- Rests on: fresh nonces are modelled as bound names
- Next step: none

A06: Finished confirms the handshake keys

The client does hold the key to this session when certifying service provider Finished; The client does hold a handshake key for the same session when it validates the client Finished.

Verifying the peer's Finished proves that the peer computed the same session handshake key, in both directions. The check value is an HMAC under a key derived from the session secret over the transcript.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: a direct consequence of the Finished construction, proved to make the property explicit
- Stated in the documents: yes. Where the documents say it: s0398, s0408
- Rests on: key schedule collapsed to one HKDF step from the hybrid secret; the hybrid secret is opaque
- Next step: none

A07: accepted identity matches the signing key holder

The recipient determines that the end-to-end identity is consistent with the subject of the peer certificate and the entity that actually holds the signature private key.

The identity a party accepts is the subject of the certificate it verified, and the signature it checked was made with that certificate's key, so the accepted identity belongs to the holder of the private key. This holds in both directions.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: identity binding is what certificate authentication is for; proved, with the PKI abstracted
- Stated in the documents: yes. Where the documents say it: rfc9846:6490
- Rests on: certificates are CA signatures over identity and key; key usage and name constraints are not checked
- Next step: none

A08: negotiated parameters not substituted

The groups, packages, CertificateVerify role labels and transcripts accepted by the recipients correspond to this proposal and the results of the consultations, and are not replaced or staggered.

The group, the suite and the role context string each party accepts are the ones that party put into the transcript, and the transcript hash is inside every signature and Finished value. A party that replaced them would need the peer's signing key or the session key.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: the profile's promise that the parameters cannot be substituted, proved for the parameters it actually negotiates
- Stated in the documents: yes. Where the documents say it: rfc9846:6499
- Rests on: one group and one suite, as the profile fixes them; the role context strings are distinct constants
- Open: a downgrade to another group or suite cannot be tested because the profile fixes both

- Next step: model a client offering an alternative group if the profile ever allows one

K01: secrecy of long-term and traffic keys

The long-term private signature key of the client and the service, as well as the handshake and application flow key derived from this session, may not be extrapolated except by the legitimate holder.

The attacker on the network never obtains either long-term signing key or the session handshake key, so the derived traffic keys stay secret. The control shows that the query can fail once a key is revealed.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: key secrecy is central; the proof rests on the abstracted KEM and the single-law Diffie-Hellman
- Stated in the documents: yes. Where the documents say it: rfc9846:6483
- Rests on: the hybrid secret is opaque and derived from both parts; traffic keys are derived from the session secret with no destructor
- Next step: none

C01: forward secrecy of past session keys

After the session has been completed and the secret of handshakes has been removed, the flow key of the previous session is not disclosed by the disclosure of the parties ' long-standing private signatures.

Revealing both long-term signing keys after the handshake does not expose the session handshake key of a completed session. The session secret comes only from the two ephemeral key-agreement parts, which the model discards with the session.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: forward secrecy follows from the fresh ephemeral parts and nothing surprising was shown
- Stated in the documents: yes. Where the documents say it: rfc9846:6504
- Rests on: the phase-1 reveal models the post-session key leak; the ephemeral secrets are not retained anywhere after the session
- Open: no control shows the query can fail, because no role publishes both key-agreement secrets
- Next step: add a role publishing both component secrets to witness the control

C02: key compromise impersonation resistance

When the client's long-term private key signature is leaked, the aggressor cannot impersonate the service to the client's acceptance; After a long-term release of the private key signed by the service, the assailant cannot impersonate the client to make the service accept the client identity.

Neither half of the property fails. With the client's signing key known, that client still accepts only the genuine server; with a server's signing key known, the server's acceptance of a client still corresponds to a real client run. The key that leaks is not the one the peer's authentication rests on.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: both directions of a stated property are proved, with the server-key answer caveated by the missing certificate
- Stated in the documents: yes. Where the documents say it: rfc9846:6522
- Rests on: the leaked client key is modelled by publishing the key of a second client identity that keeps running; the leaked server key is modelled by publishing a key without the certificate that binds it

- Open: in the server-key direction the attacker was not given the certificate that binds the key to the server identity, so that answer is for a slightly weaker attacker
- Next step: a model in which the attacker completes a server handshake, or Tamarin+ Agent, would settle the missing certificate case

C08: hybrid security if one component fails

At the time of the expiry of part X25519 or one part of part ML-KEM (whose shared secret was obtained by the person attacked), the authentication and flow key confidentiality of both parties remained valid.

When only the ML-KEM part fails, and when only the X25519 part fails, the session handshake key stays secret and both authentication directions still hold. The attacker needs both component secrets to reach the session key, which is the hybrid property the profile is built for.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: the central promise of a hybrid group, shown mechanically; the combiner is a free function, so only the structure is checked
- Stated in the documents: yes. Where the documents say it: rfc9954:550, main:268
- Rests on: the component failures are modelled by publishing the decapsulation key of one client identity and the X25519 exponent of another; the combiner is an opaque function, so nothing is said about the real combiner's strength
- Open: no control shows the session-key queries can fail, because no role publishes both component secrets
- Next step: add a role publishing both component secrets as the control

U01: CertificateVerify role separation

When the client and the service accept the peer of CertificateVerify each, the signature is generated by the peer using the peer role tag for this transcript, which cannot be re-reflected or mixed across directions.

No CertificateVerify value ever satisfies both the client-role and the server-role check: the two context strings differ and the two keys are distinct. ProVerif shows that a client never verifies a signature it made in the client role, and a server never verifies one it made in the server role.

- Result: Holds; answers what is asked: yes; usable
- Value medium: the profile's role-tag requirement checked mechanically, and the answer does not depend on a simplified primitive
- Stated in the documents: yes. Where the documents say it: s0383
- Rests on: the role context strings are distinct constants, as the profile specifies
- Next step: none

A06.12: server accept implies client held the keys

The client does hold the key to this session when it validates the client Finished and accepts the client ' s identity.

When the server accepts a client identity, the client must have held the same session handshake key, because the client Finished verifies only under that key over the transcript. This is the profile's rule that the client identity is accepted only after the client Finished.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: a direct consequence of the Finished construction and of where the accept event sits

- Stated in the documents: yes. Where the documents say it: s0398, s0402, s0408
- Rests on: key schedule collapsed to one HKDF step
- Next step: none

C02.13: one-sided long-term key leak

When the private key of the client or service is disclosed separately, the aggressor cannot pretend to be the undisclosed counterparty to be acceptable to the other party.

The content is the same as for C02: with the client's signing key known the client still accepts only the genuine server, and with a server signing key known the server still accepts only real client runs.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: same content as C02, both directions proved with the server-key caveat
- Stated in the documents: yes. Where the documents say it: rfc9846:6522
- Rests on: the leaked client key is modelled by publishing the key of a second client identity; the leaked server key is modelled without its certificate
- Open: the server-key half was answered without giving the attacker the matching certificate
- Next step: Tamarin+ Agent, or a model with the server handshake abstracted, for the missing certificate case

X01: client certificate concealed from the network

The value of the client certificate does not appear in the data available to the assailants who hold the network (readable and write all handshake records); The client ' s identity is also protected in the presence of the active aggressor.

The client's certificate never appears in data the network attacker can obtain: the client Certificate message is under the client handshake traffic key, which the attacker cannot derive. The server's certificate is treated as public data in this model.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: the property holds because the client flight is encrypted; the model cannot show what a passive observer could link over time
- Stated in the documents: yes. Where the documents say it: rfc9846:6530
- Rests on: only the client flight is encrypted in the model, so this speaks of the client certificate; certificate values are not otherwise published
- Open: unlinkability of the client identity across sessions is not asked
- Next step: a non-interference query over two client identities would address linkability

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_client			2 proved
A03	proved		auth_client			2 proved
A04	proved		inj_client			2 proved
A06	proved		auth_client			2 proved
A07	proved		auth_client			2 proved
A08	proved		auth_client			2 proved
K01	proved		secret_skC	2 of 3		3 proved
C01	proved		sess_secret			1 proved
C02	proved		auth_client_x			2 proved
C08	proved		kem_broken			4 proved
U01	proved		role_client	checked		2 proved
A06.12	proved		auth_server			1 proved
C02.13	proved		auth_client_x			2 proved
X01	proved		cert_hidden	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	The client did run the session when accepting the service certificate and the Finished (reverse identity corresponds to transcript)	`network`	proved	auth_client	none
A02	The client did run the session when the client validated the client Finished and accepted the client 's identity (reciprocal identity aligned with transcript)	`network`	proved	auth_server	none
A03	Client accepted service end runs on identity, consultative group, package and transcript	`network`	proved	auth_client	none
A03	Service accepts client identity in line with client running on identity, consultative group, package and transcript	`network`	proved	auth_server	none
A04	Each client receives a different service end and the old flight cannot be accepted again	`network`	proved	inj_client	none
A04	Each service-end acceptance of the client's identity is run against a different client, and handshake evidence cannot be duplicated	`network`	proved	inj_server	none
A06	Client Validation Service User Finished does hold the key to this session	`network`	proved	auth_client	none
A06	The client does hold a handshake key for the same session while the service validates the client 's Finished	`network`	proved	auth_server	none

target	claim	attacker	answer	query	control
A07	Client-identified service-end identity is the same as the service-end certificate subject and the signature is generated by the private key holder of the certificate	network	proved	auth_client	none
A07	The client identity identified by the service is the same as that of the client certificate subject and the signature is generated by the private key holder of the certificate	network	proved	auth_server	none
A08	The group and package that the client accepts is the value it proposes in ClientHello and bound to the certified transcript	network	proved	auth_client	none
A08	The group, package and transcript received by the service side is running in line with the client and is not replaced	network	proved	auth_server	none
K01	The key to this session's handshake cannot be extrapolated by the cyber attacker.	network	proved	sess_secret	none
K01	Client's long-term signature private key cannot be extrapolated by a cyber attacker	network	proved	secret_skC	checked
K01	The service long-term signature private key cannot be extrapolated by a cyber attacker	network	proved	secret_skS	checked
C01	After the session has been completed and the secret of handshakes has been removed, both parties will be given a long-term private signature key and the flow key of previous sessions will not be disclosed	ltk_after_session	proved	sess_secret	none
C02	After the client has permanently signed the private key leak (the assailant holds the private key), the assailant cannot impersonate the service to the client ' s acceptance	leaked_client_key	proved	auth_client_x	none
C02	(a) After the service has permanently signed the private key (the aggressor holds the private key); The model is not shared with the certificate, so its capacity is weaker than that described in the document) and the assailant cannot impersonate the client to make the client accept the client identity	leaked_server_key	proved	auth_server_leakedsrv	none
C08	ML-KEMThe flow key of this session is still not deduced when partial failure (the aggressor gets the share of the secret)	kem_broken	proved	kem_broken	none
C08	X25519The flow key of this session is still not deduced when partial failure (the aggressor gets the share of the secret)	dh_broken	proved	dh_broken	none
C08	The service-end identity and running accepted by the client remains the real service-end when the single key exchange component is disabled	kem_broken	proved	auth_client	none
C08	When a single key exchange component is disabled, the client's identity accepted by the service is still running against the real client	kem_broken	proved	auth_server_any	none
U01	Client accepted service end CertificateVerify generated by service-end role and client 's own signature is not accepted as service-end signature	network	proved	role_client	checked
U01	Client accepted by the service side CertificateVerify generated by the client 's role, and the service 's own signature will not be accepted as the client 's signature	network	proved	role_server	checked
A06.12	The client does hold the key to this session when it validates the client Finished and accepts the client 's identity	network	proved	auth_server	none
C02.13	After the client has permanently signed the private key leak (the assailant holds the private key), the assailant cannot impersonate the service to the client ' s acceptance	leaked_client_key	proved	auth_client_x	none

target	claim	attacker	answer	query	control
C02.13	(a) After the service has permanently signed the private key (the aggressor holds the private key); The model is not shared with the certificate, so its capacity is weaker than that described in the document) and the assailant cannot impersonate the client to make the client accept the client identity	`leaked_server_key`	proved	auth_server_leakedsrv	none
X01	Client certificates do not appear in the data available to cyberattackers (client identity is also protected for active attackers)	`network`	proved	cert_hidden	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-20261115T080743, 2026-11-15. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 71, prover 405 s, wall 24 min, cost 0.177 USD.

ProVerif: 163 proofs run, 154 decided, 9 without an answer, 200 s, 33 answers recorded.

What was analysed

Protocol	TLS hybrid client certificate profile
Version	rfc10024
Analysed	FixedX25519MLKEM768and the certification of the parties, the partiesCertificateVerifyRole tags and complete transcript, which only accepts client identity after authenticating client Finished; Exclude post-handshankauth/HRR/PSK。
Not analysed	Symbolic models do not support the calculation of algorithms as safe or as resistant to quantum security; The trajectory of the attack is established only under clear assumptions and is not directly equivalent to a normative loophole. Each nature must be subject to accessibility/proportional controls.
Source text	512 numbered passages (135022 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
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
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

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

The promise of the program as the source step listed it in the documents:X25519MLKEM768and the certification of the parties, the partiesCertificateVerifyRole tags and complete transcript, which only accepts client identity after authenticating client Finished; Exclude post-handshankauth/HRR/PSK。

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 authenticati on of the peer	Client accepted service-end identity from service-end certificate andCertificateVerify、Finished； Client identity accepted by the service is derived from the client certificate andCertificateVerify, Finished, and accepted only after checking the client 's Finisheds0402）。	rfc9846Appendix E List of Peer Identification: The client ' s peer identity view should reflect the client ' s identity and, if the client is certified, the client ' s peer identity view should correspond to the client ' s identity. （rfc9846:6490）	rfc9846:6490 , s0372, s0392, s0398, s0402
A03	agreement on transcript and parameters	The service-end that the client accepts runs on the identity, group, package and transcript at the time the client accepts.	rfc9846List Downglade Production: Password parameters should be the same on both sides and should be the same as in the absence of an attack (rfc9846:6499）； CertificateVerifyOverwrite Transcript-Hash()s0376), Finished Overwrite Transcript()s0408）。	rfc9846:6499 , s0376, s0408, s0026, s0031
A04	uniqueness of each accepted handshake	Each client accepts the client ' s identity, and the service accepts the client ' s identity, which corresponds to a different peer protocol; The same handshake evidence cannot be accepted for a second time.	rfc9846Unique of the session keys: any two different handshakes should produce a different and irrelevant session key (rfc9846:6495）; Finished has to be verified.s0399）。	rfc9846:6495 , s0399, s0408
A06	Finished confirms the handshake keys	The client does hold a handshake key for the same session when it authenticates its client, Finished; The service does hold the key when the client certifies the service provider Finished.	Finished is the last message of the authentication block to authenticate the handshake with the calculated key (s0398); verify_data = HMAC(finished_key, transcript)（s0408）。	s0398, s0399, s0404, s0408
A07	accepted identity matches the signing key holder	The recipient determines that the end-to-end identity is the same as the subject of the peer certificate and is the same as the entity that actually holds the private key to the signature of the certificate.	CertificateVerify is used to prove that the endpoint holds the private key corresponding to its certificate, and the recipient must authenticate the signature with the public key of the terminal entity certificate in Certificate (s0372, s0392..s0396); Peer Audit requires a peer identity view (rfc9846:6490).	rfc9846:6490 , s0372, s0392, s0393, s0394, s0395, s0396
A08	negotiated parameters not substituted	The groups, packages, CertificateVerify role labels and transcripts accepted by the recipient correspond to the results of this consultation and are not replaced or staggered.	Downgrade Protection（rfc9846:6499）； Role context string used to distinguish signatures from different contextss0383）； Transcript Overwrite All Handshake Messagess0026、s0031）。	rfc9846:6499 , s0383, s0026, s0031, s0408
K01	secrecy of long-term and traffic keys	The long-term private signature key of the client and the service, as well as the handshake key derived from this session, may not be extrapolated except by the legitimate holder.	rfc9846Focus of the session keys: shared session key should be known only by both parties to the communicationrfc9846:6483）； I'll see you at the key.s0455、s0474。	rfc9846:6483 , s0455, s0474
C01	forward secrecy of past session keys	After the session has been completed and the secret of handshakes has been removed, the flow key of the previous session is not disclosed by the disclosure of the parties ' long-standing private signatures.	rfc9846Forward secret with restoration to long-term keys: the release of the long-term key after handshake does not endanger the session key as long as the session key and the materials that can be rebuilt have been erased (rfc9846:6504. .6514； s0479）。	rfc9846:6504 , s0479, s0484, s0474

ID	Name	About	What the documents say	Where
C02	key compromise impersonation resistance	The disclosure of a long-term private signature key from one end to the other will not undermine the authentication of the other end: the aggressor cannot pretend to be the opposite to be accepted by the leaking party or to be the leaking party to be accepted by the other.	rfc9846 Key Compromise Impersonation (KCI) release: in a two-way certificate authentication connection, leaking a party's long-term secret should not undermine the party's peer's authentication in that connection (rfc9846:6522. . 6528）。	rfc9846:6522, s0372, s0383
C08	hybrid security if one component fails	X25519Part or partML-KEMThe confidentiality of the conversational flow key and the authentication of both parties remain valid when one part of the message lapses (the secret of its sharing falls into the hands of the assailant).	RFC9954hybrid nature: the result is safe as long as at least one component key exchange algorithm is not broken (rfc9954:550. .553）; RFC10024Inheritance to the same security concerns (main:268）； Sharing secrets is a fusion of two parts.s0015）。	rfc9954:550, main:268, s0001, s0015, s0018
U01	Certificate Verify role separation	When the client and the service accept the peer of CertificateVerify each, the signature will create this transcript with the peer role tag, which cannot be reflected or mixed across directions.	The service provider context string is "TLS1.3, serverCertificateVerify" , client is "TLS1.3, clientCertificateVerify" , which is used to distinguish signatures from different contexts in order to prevent cross-agreements/cross-directional attacks " .s0383）。	s0372, s0383, s0376
A06.12	server accept implies client held the keys	The client does hold the key to this session when it validates the client Finished and accepts the client ' s identity.	Constructing and Verifying Requirements for Finisheds0398、s0399、s0408）； The service does not have a reciprocal identity and activity guarantee before verifying the client 's Finished (see box).s0402）。	s0398, s0399, s0402, s0408
C02.13	one-sided long-term key leak	When the private key of the client or service is disclosed separately, the aggressor cannot pretend to be the undisclosed counterparty to be acceptable to the other party.	KCI resistance（rfc9846:6522..6528）。	rfc9846:6522, s0372, s0383
X01	client certificate concealed from the network	Client certificate (end entity certificate); The service-end certificate is the same, but the document requires only passive attacker protection.	rfc9846 List of endpoint comments: Service-end certificates should be protected against passive and active attackers, as client Certificate sends them encrypted under handshake key (rfc9846:6530. . 6535、s0413）。	rfc9846:6530, s0413, s0402

Added from the documents by the analyst: X01.

The security statements of the documents, paragraph by paragraph

The host listed the 40 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: 4. Not taken: 36. 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	C08, A03	
P02	main:273-276	The prover cannot be asked	NISTSP800-227Guidance for implementation and side channel protection recommendations, non-indicative models to determine the nature of the agreement.
P03	main:278-280	The prover cannot be asked	The fixed length of the public key/cipher/shared secret is of a coding and length nature, and the symbol model does not contain the length.

Paragraphs	Where	Taken up as	Why not
P04	main:282-287	The prover cannot be asked	ML-KEMEncapsulated random number m restored by the client and for itRNGImpact of other outputs,RNGQuality and computational layer nature.
P05	main:289-297	The prover cannot be asked	Not safe.RNGThe output is used to destroy.RNGStatusDUALECTLS) is the materiality of random number generation.
P06	main:299-306	`dh_broken`	
P07	main:308-310	The prover cannot be asked	The interaction of the two algorithms with the same unsafe RNG is a random layer of realization.
P08	rfc5280:5226-5231	No goal stated	A description of the certificate and CRL are not confidential and are accessible anonymously; This does not constitute a requirement under this Agreement.
P09	rfc5280:5233-5236	Outside the scope	The certificate management factor that affects Asurance outside of the norm, only goes to the master handshake stream.
P10	rfc5280:5238-5240	Outside the scope	CA/RAChecking process for subject identity and public key bindingPKI(a) Practice; Models.CAThe issued certificate is deemed to be lawful.
P11	rfc5280:5246-5248	Outside the scope	Ibid., accessed by the relying partyCADeclaration of practice on authentication,PKIPractice.
P12	rfc5280:5250-5259	Outside the scope	The key-use separation proposal for the key holder is not the nature of the information stream of this protocol.
P13	rfc5280:5261-5274	A07, C02, `leaked_client_key`, `leaked_server_key`	
P14	rfc5280:5276-5280	Outside the scope	CAPrivate key loss and backup,PKIOperation.
P15	rfc5280:5282-5302	Outside the scope	Resume the freshness of the information and the processing of CRL, the task does not include the revocation of the inspection.
P16	rfc5280:5304-5311	Outside the scope	The distribution of the anchor of trust is an outside process and is not included in the mandate.
P17	rfc5280:5313-5318	Outside the scope	TrustedCAUpdate of confidence information after key leaksPKIOperation.
P18	rfc5280:5320-5326	Outside the scope	Certificate processing achieves quality and path validation, which is PKI.
P19	rfc5280:5328-5334	The prover cannot be asked	The limitation on binding of key length and hash strength is the calculation of the security nature.
P20	rfc5280:5336-5339	Outside the scope	The name comparison rule, which is PKIX path validation details.
P21	rfc5280:5347-5349	Outside the scope	DN Coding Consistency is the specifics of the PKIX path verification.
P22	rfc5280:5351-5356	Outside the scope	nameConstraints encoded for PKIX path validation details.
P23	rfc5280:5358-5365	Outside the scope	The binding covers between different name forms are the details of the PKIX path verification.
P24	rfc5280:5367-5369	Outside the scope	The issuer ' s name conflicts and is operated by PKIX.

Paragraphs	Where	Taken up as	Why not
P25	rfc5280:5371-5381	Outside the scope	emailAddressCase Matches, underPKIXName processing.
P26	rfc5280:5383-5391	Outside the scope	CRLMalicious link in distribution point/extensionPKIXClient achieved.
P27	rfc5280:5393-5397	Outside the scope	httpsURI♪ Caused circulatory dependence ♪PKIXAchieved.
P28	rfc5280:5399-5406	Outside the scope	CAhttpss is not to be used in the extensionURIof whichPKIXOperation.
P29	rfc5280:5408-5416	Outside the scope	From the purpose for which the certificate was issued, it was operated by PKIX.
P30	rfc5280:5418-5428	Outside the scope	TeletexString/Latin1StringEncoding Difference,PKIXName processing.
P31	rfc5280:5430-5437	Outside the scope	String visual means confusion, which is handled by interface/PKIX name.
P32	rfc5280:5439-5449	Outside the scope	Idem. Visual confusion of names.
P33	rfc9846:4676-4677	No goal stated	The paragraph refers only to appendix C/E/F; The targets actually listed have been identified byA02、A03、A04、A06、A08、K01、C01、C02、U01is covered by the new target.
P34	rfc9954:550-558	C08, `kem_broken`, `dh_broken`	
P35	rfc9954:560-569	The prover cannot be asked	KEMIt's...IND-CCA2/FOThe security of a change with the public key is the calculation layer assumption.
P36	rfc9954:571-574	The prover cannot be asked	Long public key/secret/secret is coded and long.
P37	rfc9954:576-582	The prover cannot be asked	Longer secrets lead to a time-side channel for the processing of a hash block, which is time-series of realization.
P38	rfc9954:584-586	The prover cannot be asked	A hash is no longer resistant to the risk of long collages when collisions occur and is a computational layer assumption.
P39	rfc9954:588-592	The prover cannot be asked	The adjoining requires a single-gauge code, which is layered in nature.
P40	rfc9954:594-597	The prover cannot be asked	Only those algorithms that share secrets with fixed durations may be used, as required by the code/longness.

The parties and the protected values, one by one

The host listed the 22 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: 6. Without: 16. 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	`leaked_client_key`	
S02	Service	party	`leaked_server_key`	
S03	Certificant	party	Everything lost by definition	The issuer of the aggressor control certificate may issue a certificate for an arbitrary identity, and the full authentication nature of the question here is no longer, by definition, valid; Document HypothesisPKINormal working (%)rfc5280:5233、5311）。
S04	Attacker.	party	Nothing gained	It's the attacker himself. It doesn't give any more information.

ID	Party or value	Kind	Attacker	Why none
S05	Peer Identity	value	Nothing gained	The end-to-end identity is a public name and the certificate itself is open data (rfc5280:5226); The objective of the new “customer certificate is not known to the network” is that the certificate value does not appear online, rather than that the name is not known.
S06	Parameters for consultation	value	Nothing gained	Consultation parameters (groups, packages) are explicitly exchanged in hand and are open values.
S07	Handshake Transcript	value	Nothing gained	Transcript, made up of handshakes, is open to observation, knowing that it does not leak secrets.
S08	Handshake key	value	Everything lost by definition	Holds the flow key for this handshake, which by definition loses the confidentiality of the session and the reliability of Finished. The document does not require any nature under it.
S09	CertificateVerify Signature	value	Nothing gained	CertificateVerify Signs sent in hand are open values; Its value is that it cannot be forged, not kept secret.
S10	Finished Validation	value	Nothing gained	Finished Validation is sent in hand, is public.
S11	Terminal Entity Certificate	value	Nothing gained	The certificate is publicly verifiable data.rfc5280:5226）； Yes.TLS1.3It's just not available online.
S12	Sign Private Key	value	`leaked_client_key`, `leaked_server_key`, `ltk_after_session`	
S13	Consultative Group	value	Nothing gained	The name of the Consultative Group was expressly exchanged in ClientHello/ServerHello.
S14	Package	value	Nothing gained	The package name is explicitly exchanged in hand.
S15	Role Tabs	value	Nothing gained	The role label is an open fixed string constant (s0383).
S16	transcript	value	Nothing gained	Also S07, transcript is made up of public information.
S17	Client Sign Private Key	value	`leaked_client_key`, `ltk_after_session`	
S18	Service Sign Private Key	value	`leaked_server_key`, `ltk_after_session`	
S19	Apply flow keys	value	Everything lost by definition	The application flow key is derived from a secret handshake, which means that the confidentiality of the connection is by definition lost and the document does not require other properties under it.
S20	History Session Flow Key	value	Everything lost by definition	The flow key that holds a historical session is the loss of the confidentiality of the session by definition; The confidential question is whether it is available after a long-term key has been leaked, rather than what is established when it is held.
S21	End-to-end authentication	value	`leaked_client_key`, `leaked_server_key`	
S22	Session flow key	value	Everything lost by definition	Same as S08/S19, with a session flow key, that is, the session is by definition exposed.

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		
`leaked_client_key`	The client ' s certificate corresponds to a long-term signature private key that can be used in hand (the session is already in progress).	KCIThere's been a discussion about the situation where a party's long-standing secret has been leaked.rfc9846:6522. .6528）； If the private key is not protected, the attacker can impersonate (rfc5280:5261）。	rfc9846:6522, rfc5280:5261, s0372
`leaked_server_key`	The service-end certificate corresponds to a long-term signature private key that can be used in hand.	Ibid.rfc9846:6522、rfc5280:5261）。	rfc9846:6522, rfc5280:5261, s0372
`kem_broken`	SessionML-KEMPart of the shared secret.ML-KEMIt's broken, it can be counted during handshake.	RFC9954Require that at least one of the components remain secure until it is broken, so that the individual component is separately inspected for break-up (para.rfc9954:550. .553）。	rfc9954:550, s0015, s0008
`dh_broken`	SessionX25519Part of the shared secret.X25519is broken, e.g. by a password-related quantum computer restoring the mark from a temporary public key and usable in a session).	RFC9954_Other Organiserrfc9954:550. .553）; RFC10024It says passive observers can borrow it.CRQCRestoreECDHScale (main:299. .302）。	rfc9954:550, main:299, s0015, s0003, s0018
`ltk_after_session`	Both parties signed the private key for a long time and obtained it after the session had been completed and the secret handshake had been erased.	Forward secret with respect to long-term keys （rfc9846:6504. . 6514) with an indication of the median value to be erased (s0479).	rfc9846:6504, s0479

ID	`network`	`leaked_client_key`	`leaked_server_key`	`kem_broken`	`dh_broken`	`ltk_after_session`
A02	Holds					
A03	Holds					
A04	Holds					
A06	Holds					
A07	Holds					
A08	Holds					
K01	Holds					
C01						Holds
C02		Holds	Holds			
C08				Holds	Holds	
U01	Holds					
A06.12	Holds					
C02.13		Holds	Holds			
X01	Holds					

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
54	A02, A03, A04, A06, A07, A08, K01, C01, C02, C08, U01, A06.12, C02.13, X01	plain_primitives, many_sessions	All answers come from a model with hashing, HMAC/HKDF, public-key encryption of the client flight, signatures with certificate checks, and X25519 modelled as a plain Diffie-Hellman key agreement with the single law g^xy = g^yx; the ML-KEM part is an opaque encapsulation with an attacker-inaccessible shared secret, and the hybrid combiner is a free function. Session keys are derived from both parts, so no algebraic law beyond g^xy = g^yx is used, and no answer depends on a value that changes across messages (no counter, sequence number or replaced key is modelled: post-handshake auth, HRR, PSK and resumption are out of scope). The properties are asked for any number of sessions of each role.

The analyst's closing note

The model covers the main TLS 1.3 handshake of this profile — ClientHello, ServerHello with the X25519MLKEM768 shares, EncryptedExtensions, CertificateRequest, both Certificate/CertificateVerify/Finished flights — with the transcript hash inside every signature and Finished value, the two role context strings, and the client flight under the client handshake traffic key. It runs and the normal run is found; 28 answers are recorded on the final text and all still hold (checkpoint verified).

Every claim of every target is answered except one kind of caveat: 13 of 14 targets are answered yes and C02/C02.13 too, with the leaked-server-key half proved for an attacker that holds the key but not the certificate that binds it (giving it the certificate makes ProVerif's search exceed the 10 s limit). Proved for any number of sessions: mutual authentication (both directions), agreement on identity/group/suite/transcript, injective agreement, key confirmation both ways, the identity-to-signing-key binding, no substitution of the negotiated parameters, secrecy of both long-term signing keys and of the session handshake key, forward secrecy after a post-session long-term key leak, hybrid security when only the ML-KEM part or only the X25519 part fails (secrecy and authentication), role-tag separation between the two CertificateVerify directions, the server accepting a client identity only after the client Finished, and concealment of the client certificate from the network attacker.

No finding: no property failed, so nothing new relative to the documents, which state all of these goals (peer authentication, uniqueness, downgrade protection, session-key secrecy, forward secrecy, KCI resistance, endpoint-identity protection, hybrid security). Remaining gaps, all recorded in the report: several authentication proofs have no control because the world in which the attacker completes a server handshake does not terminate in ProVerif; the C01/C08 session-key proofs have no control either (a role publishing both key-agreement secrets would be needed); and every reading rests on an opaque ML-KEM, X25519 with only g^xy = g^yx, an abstracted PKI without revocation, one fixed group and suite, and the exclusion of PSK, 0-RTT, HRR, post-handshake auth and alerts. Tamarin+ Agent would settle the server-certificate case and give the missing controls.