Security analysis of PQC EAP-TLS full authentication

This report in another language: 中文.

draft-ietf-emu-pqc-eap-tls-04 · ProVerif+ Agent report

Documents	Post-Quantum Enhancements to TLS-Based EAP Methods (protocol definition); EAP-TLS 1.3: Using the Extensible Authentication Protocol with TLS 1.3 (supporting); The Transport Layer Security (TLS) Protocol Version 1.3 (supporting); Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3 (supporting)
Properties examined	18: 3 hold, 2 fail (3 attacks), 13 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 a full EAP-TLS 1.3 authentication for EAP deployments that must resist a future quantum adversary. Two parties take part: the EAP peer, which holds its own certificate and ML-DSA-65 signing key, and the EAP server, which holds its certificate and signing key. Both trust one root CA, and the peer is configured with the server name it accepts. The authenticator in between only relays EAP packets; the key it receives from the EAP server is outside the analysis. The exchange is the EAP identity request and response, the TLS start, then a TLS 1.3 full handshake over the X25519MLKEM768 hybrid group with mutual ML-DSA-65 certificate authentication. The server then sends the encrypted TLS application data 0x00 as the protected success indication, the peer answers with an empty EAP response, and the server sends EAP-Success. Both sides export the Key_Material from the TLS exporter, take the MSK from its first 64 octets and the EMSK from the next 64. When the exchange ends the peer is sure that the party on the other side signed the handshake transcript with the key of the certificate it trusts, and the server is sure of the same about the peer's certificate. Neither is made sure of anything by the final EAP-Success, which carries no protection of its own.

The parties:

- **EAP peer**: holds its certificate, its ML-DSA-65 key, the root CA key and the name it expects for the server. Holds: certP, skP, skCA, nameS.
- **EAP server**: holds its certificate, its ML-DSA-65 key and the root CA key. Holds: certS, skS, skCA.

Figure 1 shows the exchange as the documents specify it.

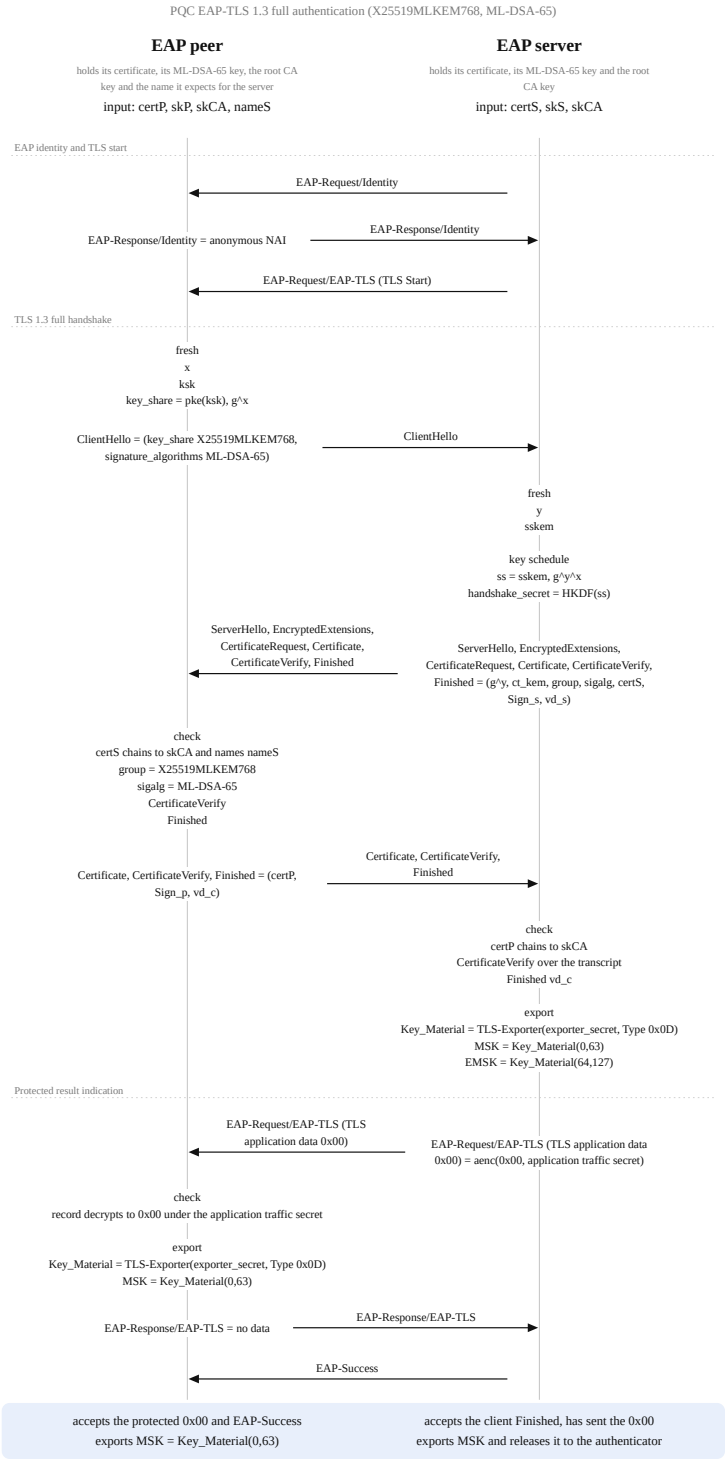


Figure 1. The protocol as specified.

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.

The handshake as specified behaves as intended wherever the prover could decide the question, and no new weakness was found. Two weaknesses the documents already state are reproduced: an attacker that recovers the server's ML-DSA-65 signing key impersonates the EAP server to a peer, and EAP-Success carries no protection, so a peer can be told the run succeeded by the attacker while the protected 0x00 record is what it may really rely on. The ordering rules around the success indication hold for any number of sessions, the certificate signing keys are not obtainable from the network, and the identity in the client certificate stays hidden from a passive observer. Agreement on a single run, injectivity, freshness, the negotiable parameters, the nonce rule and the hybrid component claims have no answer: the shared secret is modelled with only the law g^xy = g^yx and ProVerif does not identify the two spellings of it, so six target and attacker pairs stay open and several recorded answers are marked as artifacts of the model. Of the 18 targets, 5 rest on a proved query or a verified secrecy result, 2 are attacks the documents state, 5 are undecided and 6 pairs are open; the arithmetic of the hybrid group is what Tamarin+ Agent would settle.

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 agreement on a real peer run.** When the EAP peer accepts EAP-Success, there is a real run of an EAP server with that peer as its counterpart; when the EAP server accepts client Finished, there is a real run of an EAP peer with that server as its counterpart.	no compromise	Attack holds (stated in the documents)	medium	yes
A03	**agreement on the exported key material.** When the peer and the server each accept, the two sides agree on the identities, the negotiated group, the signature algorithm, the transcript and the Key_Material exported from exporter_secret.	no compromise	No answer	none	unclear
A04	**injective agreement on keying.** Every EAP-Success acceptance by the peer and every client Finished acceptance by the server correspond to distinct counterpart runs (injective matching).	no compromise	No answer	none	unclear
A06	**success indication after client Finished.** When the peer accepts the protected 0x00 success indication, the server has already exported the same session key and has already processed client Finished.	no compromise	No answer	medium	yes
A08	**transcript and negotiation binding.** The negotiated group, the signature algorithm, the transcript and the exporter context accepted by the peer and the server all come from the real proposals and selections of this handshake.	no compromise	Reachable only	none	unclear
K01	**secrecy of MSK and long-term keys.** The attacker cannot export the MSK/Key_Material or exporter_secret, and cannot obtain the long-term signing private keys of the peer or the server.	no compromise	Holds	medium	no
C01	**forward secrecy of the MSK.** After the session completes and the long-term signing private keys of both sides leak, the attacker still cannot export that session's Key_Material/MSK or decrypt its payload.	the long-term signing private keys of the peer and the server (leaked after the session completes)	No answer	none	unclear

ID	Property	Compromise assumed	Result	Value	In the documents
C08	**hybrid security with one part broken.** Under the assumption that either X25519 or ML-KEM-768 is broken, the session key and the payload stay secret.	one of the X25519 ephemeral private key or the ML-KEM-768 private key (during or after the handshake)	No answer	none	unclear
R01	**freshness of the success indication.** The 0x00 success indication accepted by the peer and the client Finished accepted by the server both come from the random numbers and the transcript of this handshake, not from a replay of an old session.	no compromise	No answer	none	unclear
R02	**one acceptance per message.** Within the same session, the same 0x00 success indication or client Finished is accepted at most once.	no compromise	No answer	none	unclear
R05	**only PQ parameters accepted.** The version, the key exchange group and the signature algorithm accepted by the peer and the server meet the minimum policy of TLS 1.3 and PQ/hybrid.	no compromise	Reachable only	none	unclear
S01	**unique nonce per traffic key.** Under the same session key, the nonce of every AEAD encryption call is unique.	no compromise	No answer	none	unclear
S04	**MSK bound to the TLS identity.** The exported MSK is correctly bound to the authenticated identity and the transcript of this TLS session; the outer EAP identity is not used for authorization.	no compromise	No answer	medium	yes
A03.14	**peer and server agree on key material.** When the peer and the server each accept, the two sides agree on the identities, the transcript and the Key_Material exported from exporter_secret.	no compromise	No answer	none	unclear
U01	**MSK secrecy and ordered EAP-Success.** The MSK is known only to the legitimate two parties; EAP-Success is sent only after a legitimate client Finished has been processed and the protected 0x00 success indication has been sent.	no compromise	Holds	medium	yes
U02	**outer identity cannot detach the MSK.** Substituting or replaying the outer EAP identity does not change the binding of the MSK to this TLS authenticated identity and transcript, and the attacker cannot export the MSK.	no compromise	No answer	medium	yes
X01	**server impersonation after signing-key recovery.** an attacker holding the server certificate signing key can make an EAP peer complete the handshake and derive an MSK although no EAP server ran; without that key no peer acceptance without a server run can be produced	the ML-DSA-65 private keys of the peer certificate and of the server certificate, recovered before or during a session	Attack holds (stated in the documents)	medium	yes
X02	**peer identity confidentiality in the handshake.** a passive attacker that records the handshake does not learn the identity carried in the peer certificate unless it breaks a component of the hybrid key agreement	no compromise	Holds	medium	yes

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

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

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: EAP peer/server authentication and binding to the TLS session	The server side holds and the peer side does not: with the server's signing key an attacker makes a peer complete the run with no server, and on the network alone a peer accepts an EAP-Success the server never sent. Agreement between the two sides on the exported key material could not be decided.	A02, A03, X01
Q2: MSK secrecy and EAP-Success not before legitimate TLS success	The MSK is not obtainable by an attacker on the network, and the ordering is proved: EAP-Success follows the client Finished and the protected 0x00 for any number of sessions. The MSK secrecy result holds in the model only because the two component disclosures are needed to break it.	K01, U01
Q3: effect of outer identity confusion/replay on the exporter session	Substituting or replaying the outer identity does not move the MSK and does not reveal it, because the MSK follows the certificate identity authenticated inside TLS, and that identity stays hidden from a passive observer unless a component of the hybrid group is broken.	U02, S04, X02

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 hybrid group is modelled as a pair of a KEM secret and a Diffie-Hellman term with the single law g^xy = g^yx; the two sides write that term in different orders and ProVerif does not identify them, which is the cause of the inconclusive results and of every answer marked as a model artifact.
- The two component disclosures are modelled as requests any attacker may make at any time, so no target could be answered for an attacker that does not take them; that is why no usable answer exists for the forward-secrecy and hybrid targets.
- Certificates are checked against one root CA key and one expected server name; revocation, OCSP stapling, intermediate chains and certificate compression are left out.
- No value that changes: no NewSessionTicket, no resumption, no KeyUpdate and no record sequence numbers, so a counter, a rekeying and the repeat of a record cannot be examined and a bounded model would be needed for them.
- The EAP server to authenticator key transport, the AAA protocol, the lower-layer four-way handshake and the access-network channel properties are the boundary the task sets, so nothing is said about them.
- The symbolic model proves nothing about the computational strength of ML-KEM or ML-DSA, nor about side channels, random number generation or implementation faults.
- Timing, availability and denial of service are not properties the prover answers, so the use of a forged Success packet as a denial of service is not examined.
- For the injective and record-matching queries, ProVerif hit its time limit, so neither injectivity nor the freshness questions have an answer here.
- Six target and attacker pairs remain open: the peer-side checks that a run happened, the forward secrecy of the MSK and the two component cases of the hybrid group, and the binding of the MSK to the certificate identity. Their traces needed the assumed disclosures, so no answer is recorded for them rather than an attack that the model produced for its own reasons.

What the host records:

- No answer for: A03, A04, A06, A08, C01, C08, R01, R02, R05, S01, S04, A03.14, U02.
- The answers changed after the reading was written for: A02, A03, K01.
- The answers of A02, A03, A08, K01, C01, C08, R05, S01, S04, U01, U02, X01, X02 rest on an operation the model simplifies (Appendix D).
- The prover gave no answer (Appendix C): A03: a03_km, A04: a04_inj_peer, A06: a06_zero, R01: r01_zero, R02: r02_once, S01: s01_once, A03.14: a03_km. The property was asked and the prover neither proved it nor found an attack, so the property is shown without an answer.
- Set by the analyst and without a proof or an attack (Appendix D): A08 under `network`; C01 under `ltk_leak`; C08 under `x25519_broken`; C08 under `mlkem_broken`; R05 under `network`; S04 under `network`; U02 under `network`.
- Proved without a control query (Appendix C): K01: k01_ltk_p, K01: k01_ltk_s, U01: u01_order, U01: u01_zero, X02: id_ctl. 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 remains 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 agreement on a real peer run

When the EAP peer accepts EAP-Success, there exists a real run of an EAP server with that peer as its peer; when the EAP server accepts the client Finished, there exists a real run of an EAP peer with that server as its server.

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

The server side holds: no trace makes a server accept a client Finished that no peer produced. On the peer side the two claims come apart. With the certificate keys an attacker makes a peer accept and export an MSK with no server run, and even on the network alone the peer accepts an EAP-Success the server did not send, because EAP-Success carries no protection; what the peer may rely on is the protected 0x00 record.

- Result: attack; answers what is asked: partly; usable with caveats
- Value medium: The peer-side failure is the weakness RFC 9190 section 5.5 states for EAP-Success and the impersonation is the one main section 4 states for a recovered certificate key; what the analysis adds is the exact messages and the exact trace.
- Stated in the documents: yes. Where the documents say it: rfc9190:1095, main:306, s0155, s0018
- Rests on: the X25519 shared secret is modelled with the single law g^xy = g^yx and the ML-KEM secret as a one-way function, and ProVerif does not identify the two spellings of that secret; certificates are checked against one root CA key and one expected server name; revocation and OCSP are left out; the authenticator is a pass-through and its key transport is the excluded boundary
- Open: the server-side direction as a proved correspondence; agreement on one run rather than on the transcript value

A03: agreement on the exported key material

When the peer and the server each accept, the two sides agree on the identity, the negotiated group, the signature algorithm, the transcript and the Key_Material exported from the exporter_secret.

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

The two sides run the same key schedule over the same transcript, and the prover could neither prove nor refute that their exported Key_Material agrees. Its derivation pairs two different sessions and hashes terms that differ only in the order of the Diffie-Hellman exponents, so the answer says nothing either way about the protocol.

- Result: undecided; answers what is asked: partly; not usable
- Value none: No answer came back.
- Stated in the documents: unclear
- Rests on: the one-law Diffie-Hellman equation, applied in the derivation but not when two events are matched; the exporter as an uninterpreted one-way function
- Open: whether the exported Key_Material agrees between the two sides

A04: injective agreement on keying

Every acceptance of EAP-Success by the peer and every acceptance of client Finished by the server correspond to distinct peer runs (injective matching).

No answer: the injective queries did not finish, so it is not established whether one acceptance can stand for two runs.

- Result: undecided; answers what is asked: undetermined; not usable
- Value none: The prover gave nothing to stand on.
- Stated in the documents: unclear
- Rests on: no result at all; the per-run identifiers sit inside hashed transcripts, which is what makes the injective query expensive
- Open: injectivity in either direction

A06: success indication after client Finished

When the peer accepts the protected 0x00 success indication, the server has already exported the same session key and has processed the client Finished.

The ordering the target really asks for is established: the server sends the protected 0x00 only after it accepted the client Finished, and EAP-Success only after the 0x00. Whether the peer's acceptance of that record is matched by a server run that produced it could not be decided; the record is sealed under keys that only come out of the same session.

- Result: undecided; answers what is asked: partly; usable with caveats
- Value medium: RFC 9190 section 2.5 states the ordering rules and the analysis confirms the model keeps them; the correspondence half has no answer.
- Stated in the documents: yes. Where the documents say it: rfc9190:1094, s0138
- Rests on: the record-matching query hit the time limit; one AEAD record for the 0x00, with no record numbering or padding
- Open: the peer's acceptance of the 0x00 matched to a server run

A08: transcript and negotiation binding

The negotiated group, the signature algorithm, the transcript and the exporter context that the peer and the server accept all come from the real offer and selection of this handshake.

The model can only show that a run completes under the post-quantum group and signature scheme, because it has no second group or scheme to fall back to. A policy failure therefore cannot be exhibited, and the answer says nothing about what a peer configured with traditional parameters as well would do.

- Result: reachable; answers what is asked: no; not usable
- Value none: The only recorded answer is that the situation is reachable, which is not the property asked.
- Stated in the documents: unclear
- Rests on: one group and one signature scheme in the model; the policy check is a comparison with a constant in the peer
- Open: whether a peer can be made to accept a traditional group or a traditional signature scheme

K01: secrecy of MSK and long-term keys

The attacker cannot derive the MSK/Key_Material or the exporter_secret, nor obtain the long-term signing private key of the peer or the server.

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

Neither certificate signing key is obtainable by an attacker on the network. The MSK itself is obtainable in the model, but only after an attacker has taken both modelled component disclosures, which is the compromise the other targets speak about and not a network attack.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value medium: The result follows from the model's disclosure mechanism; the documents state that the MSK is exported from the TLS session without discussing this.
- Stated in the documents: no
- Rests on: the component disclosures are modelled as requests any attacker may make, so the model has no way to forbid them; the MSK is an uninterpreted function of the exporter output
- Open: MSK secrecy for an attacker restricted to the network

C01: forward secrecy of the MSK

If both parties' long-term signing private keys leak after the session completes, the attacker still cannot derive that session's Key_Material/MSK or decrypt its payload.

Nothing is shown about forward secrecy: the falsifying trace needed the two component disclosures on top of the certificate keys, so it does not demonstrate what the target asks. The exclusivity that would make the answer mean forward secrecy could not be proved either.

- Result: No answer; answers what is asked: no; not usable
- Value none: The answer is an artifact of the disclosure mechanism.
- Stated in the documents: unclear
- Rests on: the component disclosures are always available, so the assumed compromise cannot be isolated; the one-law Diffie-Hellman equation and the two spellings of the shared secret
- Open: whether the MSK survives the loss of both certificate keys

C08: hybrid security with one part broken

Under the assumption that either X25519 or ML-KEM-768 is broken, the session keys and payload remain confidential.

Neither component claim is answered: both falsifying traces use the other disclosures as well, so they say nothing about the hybrid group keeping the secret when one component breaks. That the hybrid construction answers the harvest-now-decrypt-later threat therefore rests on the documents, not on this analysis.

- Result: No answer; answers what is asked: no; not usable
- Value none: Both answers are artifacts of the disclosure mechanism.
- Stated in the documents: unclear
- Rests on: the component disclosures are always available; the two spellings of the X25519 shared secret are not identified by the prover
- Open: the component-wise hybrid argument for either component

R01: freshness of the success indication

Both the 0x00 success indication accepted by the peer and the client Finished accepted by the server come from the random values and transcript of this handshake, not from a replay of an old session.

No answer: the query did not finish. The transcript hash and the per-run traffic keys do bind the peer's acceptance to its own key share, and an old record cannot be decrypted under a new session's keys, but the model did not establish it.

- Result: No answer; answers what is asked: undetermined; not usable
- Value none: The prover gave nothing to stand on.
- Stated in the documents: unclear
- Rests on: no result; no record numbers or sequence numbers, so the only freshness is the key share and the traffic keys
- Open: freshness of the 0x00 acceptance

R02: one acceptance per message

Within one session, the same 0x00 success indication or client Finished is accepted at most once.

No answer: the injective query hit the time limit. The Finished check and the 0x00 check are per session and the traffic keys are per run, but whether a record can be accepted twice was not decided.

- Result: No answer; answers what is asked: undetermined; not usable
- Value none: The prover gave nothing to stand on.
- Stated in the documents: unclear
- Rests on: no result; no sequence numbers, so a repeated record is a repeated term
- Open: at most one acceptance per message

R05: only PQ parameters accepted

The versions, key exchange groups and signature algorithms accepted by the peer and the server meet the minimum policy of TLS 1.3 and PQ/hybrid.

The model has one group and one signature scheme, so it can only show a run under them; it cannot show what a peer does when an attacker offers a traditional group instead. The requirement that deployments negotiate a post-quantum group and signature is therefore modelled as satisfied rather than tested.

- Result: Reachable only; answers what is asked: no; not usable
- Value none: The recorded answer is reachability, which is not the property asked.
- Stated in the documents: unclear
- Rests on: one group and one signature scheme; the policy check compares with a constant
- Open: a downgrade attempt against a peer that also supports a traditional group

S01: unique nonce per traffic key

Under the same session key, the nonce of each AEAD encryption call is unique.

The prover could neither prove nor refute that no two records are sealed under one traffic key with the same nonce; its derivation pairs two runs that share a key only in the over-approximation. In the plain run each record has its own nonce, but that is not a proof.

- Result: No answer; answers what is asked: undetermined; not usable
- Value none: No answer came back.
- Stated in the documents: unclear
- Rests on: one AEAD invocation per flight and no sequence numbers; the derivation is an over-approximation that ProVerif could not turn into a trace
- Open: whether one traffic key can seal two records with the same nonce

S04: MSK bound to the TLS identity

The exported MSK is correctly associated with the authenticated identity and transcript of this TLS session; the EAP outer identity is not used for authorization.

The server authenticates the certificate identity, not the outer NAI, and derives the MSK from that same TLS session; the trace shows it, because the client flight and the transcript that the MSK is derived from carry the certificate identity. The correspondence query was reported false only because the peer's own keying event comes later in the run.

- Result: No answer; answers what is asked: partly; usable with caveats
- Value medium: RFC 9190 sections 2.2 and 5.6 state that the outer identity is unauthenticated and must not be used for authorization; the analysis shows the MSK following the certificate identity.
- Stated in the documents: yes. Where the documents say it: rfc9190:1136, s0115
- Rests on: the outer identity is a free value the attacker rewrites, which is the documented behaviour; the MSK is released to the authenticator, whose transport is the excluded boundary
- Open: the binding as a proved correspondence

A03.14: peer and server agree on key material

When the peer and the server each accept, both agree on the identity, the transcript and the Key_Material derived from exporter_secret.

Same answer as A03: the same key schedule over the same transcript, with no verdict from the prover, whose derivation pairs different sessions.

- Result: No answer; answers what is asked: partly; not usable
- Value none: No answer came back.
- Stated in the documents: unclear
- Rests on: the one-law Diffie-Hellman equation as before; the exporter as a one-way function
- Open: agreement on the exported key material

U01: MSK secrecy and ordered EAP-Success

The MSK is known only to the legitimate parties; EAP-Success is sent only after a legitimate client Finished has been processed and the protected 0x00 success indication has been sent.

EAP-Success is only sent after the server has accepted the client Finished, and only after it has sent the protected 0x00 record. Both hold for any number of sessions, which is what the user asked about the ordering of the success indication.

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

- Value medium: RFC 9190 section 2.5 states both rules as MUSTs and the analysis confirms the model keeps them.
- Stated in the documents: yes. Where the documents say it: rfc9190:1094, s0138
- Rests on: the server's program order is the specification's order; one server role with no branches; no post-handshake messages and no resumption
- Open: the MSK secrecy half of the target, answered under K01

U02: outer identity cannot detach the MSK

Replacing or replaying the outer EAP identity does not change the MSK's binding to this TLS authenticated identity and transcript, and the attacker cannot derive the MSK.

The trace shows a peer completing normally while a different server session recorded a different outer identifier, which is expected: the outer identity is free for the attacker to choose and is bound to nothing. What matters is that the MSK stays tied to the TLS session and its certificate identity, which the trace shows but no correspondence query proves.

- Result: No answer; answers what is asked: partly; usable with caveats
- Value medium: RFC 9190 section 2.2 states that the outer identity is not authenticated and is trivial to forge, modify or replay; the analysis reproduces that this does not move the MSK.
- Stated in the documents: yes. Where the documents say it: rfc9190:1136, s0115
- Rests on: the conclusion asks for an exact event match between two different sessions of the outer identity exchange; the outer identity is never authenticated, by design
- Open: the binding as a proved correspondence

X01: server impersonation after signing-key recovery

an attacker holding the server certificate signing key can make an EAP peer complete the handshake and derive an MSK although no EAP server ran; without that key no peer acceptance without a server run can be produced

With the server's ML-DSA-65 signing key an attacker makes a peer complete the handshake and export an MSK although no EAP server ran: it signs its own CertificateVerify and finished flight, and the peer checks only that the certificate chains to the trusted root and names the expected server. This is the impersonation the threat section describes for a recovered certificate key.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable
- Value medium: main section 4 states this impersonation; the analysis gives the message-by-message trace.
- Stated in the documents: yes. Where the documents say it: main:306, s0018
- Rests on: the peer checks against a single root key and one expected name; revocation and OCSP are left out; the borrowed key is available before the run
- Open: whether revocation checking shrinks the window, which the model leaves out

X02: peer identity confidentiality in the handshake

a passive attacker that records the handshake does not learn the identity carried in the peer certificate unless it breaks a component of the hybrid key agreement

An attacker that records the handshake does not obtain the identity in the client certificate: the certificate travels inside records sealed under handshake traffic keys that come from the ephemeral shared secret. This holds although the outer NAI is in the clear, because the NAI is the anonymous form.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: RFC 9190 sections 2.1.8 and 5.8 state that with TLS 1.3 the certificates are encrypted and the username is not revealed, and main section 3 names the exposure; the analysis confirms it holds in the model.
- Stated in the documents: yes. Where the documents say it: rfc9190:1259, s0105, s0108
- Rests on: identity confidentiality is asked as a secrecy query on the certificate identity, not as a test over two peer identities; the model has one peer identity, so unlinkability across runs is not asked; with both component disclosures the identity would be given away, which the query states as its exception
- Open: unlinkability across two runs of the same peer

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	none				a02_p2s (holds_in_source); a02_s2p (model_artifact); a02_p2s_ltk (holds_in_source); a02_s2p_ltk (model_artifact)	4 attack
A03	none				a03_km_rev (model_artifact)	1 attack, 1 undecided
A04	none					1 undecided
A06	none					1 undecided
A08	reachable		a08_policy			1 reachable
K01	proved		k01_ltk_p		k01_msk (model_artifact); k01_km (model_artifact)	2 proved, 2 attack
C01	none				c01_msk (model_artifact)	1 attack
C08	none				c08_dh (model_artifact); c08_kem (model_artifact)	2 attack
R01	none					1 undecided
R02	none					1 undecided
R05	reachable		a08_policy			1 reachable
S01	reachable		s01_aead			1 reachable, 1 undecided
S04	none				s04_id (model_artifact)	1 attack
A03.14	none					1 undecided
U01	proved		u01_order			2 proved
U02	none				u02_bind (model_artifact)	1 attack
X01	none				a02_p2s_ltk (holds_in_source)	1 attack
X02	proved		id_ctl			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 peer's acceptance of EAP-Success implies a server run that sent it, against the attacker on the network alone	`network`	attack (holds_in_source)	a02_p2s	
A02	the server's acceptance of the client Finished implies a peer run that sent it, against the attacker on the network alone	`network`	attack (model_artifact)	a02_s2p	
A02	the peer's acceptance of EAP-Success implies a server run that sent it, against an attacker that holds the certificate signing keys	`ltk_leak`	attack (holds_in_source)	a02_p2s_ltk	
A02	the server's acceptance of the client Finished implies a peer run that sent it, against an attacker that holds the certificate signing keys	`ltk_leak`	attack (model_artifact)	a02_s2p_ltk	

target	claim	attacker	answer	query	control
A03	the Key_Material the peer exports equals the one the server exports in one run, against the attacker on the network alone	`network`	undecided	a03_km	
A03	the Key_Material the server exports equals the one the peer exports in the same run, read from the server's side	`network`	attack (model_artifact)	a03_km_rev	
A04	each peer acceptance corresponds to a distinct server run (injective agreement), against the network attacker	`network`	undecided	a04_inj_peer	
A06	the peer accepts the protected 0x00 only for a server run that processed its client Finished and sent that record	`network`	undecided	a06_zero	
A08	the peer accepts only a group and signature algorithm from the post-quantum set its deployment requires, against the network attacker	`network`	reachable	a08_policy	
K01	the peer certificate signing key is not obtainable by the attacker on the network alone	`network`	proved	k01_ltk_p	none
K01	the server certificate signing key is not obtainable by the attacker on the network alone	`network`	proved	k01_ltk_s	none
K01	the MSK and Key_Material of a session are not obtainable by the attacker on the network alone	`network`	attack (model_artifact)	k01_msk	
K01	the Key_Material itself is not obtainable by the attacker on the network alone	`network`	attack (model_artifact)	k01_km	
C01	the Key_Material/MSK of a completed session stays secret from an attacker that holds both certificate signing keys	`ltk_leak`	attack (model_artifact)	c01_msk	
C08	the session keys stay secret when the X25519 component of the hybrid group is broken	`x25519_broken`	attack (model_artifact)	c08_dh	
C08	the session keys stay secret when the ML-KEM-768 component of the hybrid group is broken	`mlkem_broken`	attack (model_artifact)	c08_kem	
R01	the 0x00 indication the peer accepts was produced in this handshake over the peer's own key share, against the network attacker	`network`	undecided	r01_zero	
R02	the client Finished and the 0x00 indication are accepted at most once per session, against the network attacker	`network`	undecided	r02_once	
R05	only the post-quantum hybrid group and ML-DSA are negotiated, against the network attacker	`network`	reachable	a08_policy	
S01	each record is sealed under a traffic key with its own nonce, against the network attacker	`network`	reachable	s01_aead	
S01	no two records are sealed under one traffic key with the same nonce, against the network attacker	`network`	undecided	s01_once	
S04	the identity the server authenticates is the one in the client certificate authenticated inside TLS, and the MSK comes from that same TLS session, against the network attacker	`network`	attack (model_artifact)	s04_id	
A03.14	peer and server derive the same Key_Material for one run (user question Q1)	`network`	undecided	a03_km	
U01	EAP-Success is sent only after the client Finished was accepted, against the network attacker	`network`	proved	u01_order	none
U01	EAP-Success is sent only after the protected 0x00 indication, against the network attacker	`network`	proved	u01_zero	none

target	claim	attacker	answer	query	control
U02	substituting or replaying the outer EAP identity does not let the attacker obtain the MSK or tie it to another identity, against the network attacker	`network`	attack (model_artifact)	u02_bind	
X01	an attacker holding the server certificate signing key makes an EAP peer complete the authentication with no EAP server run	`ltk_leak`	attack (holds_in_source)	a02_p2s_ltk	
X02	the identity in the client certificate is not obtainable by an attacker that records the handshake, unless a component of the hybrid key agreement is broken	`network`	proved	id_ctl	none

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-20261114T224408, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 39, prover 176 s, wall 16 min, cost 0.204 USD.

What was analysed

Protocol	PQC EAP-TLS full authentication
Version	draft-ietf-emu-pqc-eap-tls-04
Analysed	One full EAP-TLS 1.3 mutual authentication; X25519MLKEM768 and ML-DSA-65 fixed; models the EAP Identity/Success conditions, MSK export via the TLS exporter, TLS Finished, and the handshake transcript. Trusted key transport between the EAP server and the authenticator is a boundary; EST, TEAP, and resumption are excluded.
Not analysed	The symbolic model does not prove computational security or post-quantum security of the algorithms; attack traces hold only under explicit assumptions and do not directly correspond to specification vulnerabilities. Every property requires a reachability/positive control.
Source text	655 numbered passages (182927 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	3	Data Confidentiality in TLS-Based EAP Methods
main	4	Post-Quantum Authentication in TLS-Based EAP Methods
main	7	Security Considerations
rfc9190	2	Protocol Overview
rfc9190	2.1	Overview of the EAP-TLS Conversation
rfc9190	2.1.1	Authentication
rfc9190	2.1.2	Ticket Establishment
rfc9190	2.1.3	Resumption
rfc9190	2.1.4	Termination
rfc9190	2.1.5	No Peer Authentication
rfc9190	2.1.6	Hello Retry Request
rfc9190	2.1.7	Identity
rfc9190	2.1.8	Privacy
rfc9190	2.1.9	Fragmentation
rfc9190	2.2	Identity Verification
rfc9190	2.3	Key Hierarchy
rfc9190	2.4	Parameter Negotiation and Compliance Requirements

Document	Section	Title
rfc9190	2.5	EAP State Machines
rfc9190	3	Detailed Description of the EAP-TLS Protocol
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
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

Document	Section	Title
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
rfc10024	4	Negotiated Groups
rfc10024	4.1	Client Share
rfc10024	4.2	Server Share
rfc10024	4.3	Shared Secret

The purpose of the protocol as the source step located it in the documents: One full EAP-TLS 1.3 mutual authentication; X25519MLKEM768 and ML-DSA-65 fixed; models the EAP Identity/Success conditions, MSK export via the TLS exporter, TLS Finished, and the handshake transcript. Trusted key transport between the EAP server and the authenticator is a boundary; EST, TEAP, and resumption are excluded.

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 agreement on a real peer run	the EAP peer is assured, on EAP-Success, that a real EAP server ran the handshake with it; the EAP server is assured, on accepting the client Finished, that a real EAP peer ran it	RFC 9190 s0046/s0049: the server sends EAP-Success only after the handshake completed with the peer's Finished; the certificates and CertificateVerify of both sides authenticate the parties.	s0046, s0049, s0497, s0523
A03	agreement on the exported key material	the EAP peer and the EAP server derive the same Key_Material/MSK for one run	RFC 9190 s0121-s0124: both sides export Key_Material from the same TLS session; RFC 9846 s0523/s0524 binds it to the transcript.	s0121, s0122, s0124, s0523
A04	injective agreement on keying	each acceptance of the MSK on one side corresponds to a distinct run on the other side	implied by the Finished semantics of RFC 9846 s0524 with fresh ephemeral shares (RFC 9190 s0046); the documents do not state injectivity explicitly.	s0046, s0524

ID	Name	About	What the documents say	Where
A06	success indication after client Finished	the EAP peer, accepting the protected 0x00, is assured that the server processed its Finished and holds the session keys	RFC 9190 s0138: the server MUST NOT send the protected success indication before it has processed the client Finished.	s0138, s0046
A08	transcript and negotiation binding	what the peer accepts is bound to the key shares and to the negotiated group and signature algorithm of this handshake	RFC 9846 s0533 the Finished covers the transcript; RFC 10024 s0652 the hybrid shared secret; main s0011/s0019 the PQ group and signature requirement.	s0011, s0019, s0533, s0652
K01	secrecy of MSK and long-term keys	the Key_Material/MSK of a session, kept from the attacker; the certificate signing keys, not obtainable from the network	RFC 9190 s0121-s0124 the MSK is exported from the TLS session; main s0009/s0010 require long-term confidentiality.	s0121, s0124, s0009
C01	forward secrecy of the MSK	the Key_Material/MSK of a completed session, kept from an attacker that later obtains the certificate signing keys	main s0002/s0003/s0009: the HNDL threat is answered by ephemeral key exchange; RFC 9190 s0046 forward secrecy comes from the ephemeral key_share.	s0002, s0003, s0009, s0046
C08	hybrid security with one part broken	the session keys, kept from an attacker that breaks one component of X25519MLKEM768	main s0012: a PQ/T hybrid shared secret remains secure as long as either component algorithm remains unbroken.	s0012, s0652, s0010
R01	freshness of the success indication	the 0x00 success indication the peer accepts, produced in this handshake over the peer's own key share and not replayed	RFC 9846 s0527 warns that the ClientHello may be replayed before the handshake completes; RFC 9190 s0138 makes the 0x00 the proof of a completed run.	s0138, s0527
R02	one acceptance per message	the client Finished and the 0x00 indication, accepted at most once per session	RFC 9846 s0524 the recipient MUST verify the Finished; RFC 9190 s0138 one protected success indication per run.	s0524, s0138
R05	only PQ parameters accepted	the negotiated group and signature algorithm, which the peer must take only from the post-quantum set its deployment requires	main s0009/s0010/s0019: deployments requiring long-term confidentiality MUST negotiate a PQ/T hybrid group and MUST support a PQ signature scheme.	s0009, s0010, s0019
S01	unique nonce per traffic key	each AEAD invocation under one traffic key, which must use a distinct nonce	RFC 9846 s0618 derives the per-record nonce from the record sequence number; the documents state the requirement, not an attack.	s0138, s0618
S04	MSK bound to the TLS identity	the MSK delivered to the authenticator, which belongs to the identity authenticated inside TLS and not to the outer EAP identifier	RFC 9190 s0115: the EAP-Response/Identity is not authenticated and MUST NOT be used for authorization; s0121/s0141 the MSK comes from the TLS session.	s0115, s0121, s0141
A03.14	peer and server agree on key material	the peer's and the server's Key_Material for one run (user question Q1)	as A03: RFC 9190 s0121-s0124 with the RFC 9846 s0523 Finished binding.	s0121, s0124, s0523
U01	MSK secrecy and ordered EAP-Success	the MSK, known only to the two parties, and EAP-Success, sent only after the client Finished and the protected 0x00 (user question Q2)	RFC 9190 s0138/s0141 and s0121/s0124.	s0138, s0141, s0124
U02	outer identity cannot detach the MSK	which session the MSK belongs to, when the outer EAP identity is substituted or replayed (user question Q3)	RFC 9190 s0115 the outer identity is unauthenticated and must not be used for authorization; s0121 the MSK is derived from the TLS session only.	s0115, s0121
X01	server impersonation after signing-key recovery	the EAP peer's assurance that it ran the handshake with the real EAP server	main s0018: an on-path attacker able to recover a server's private key within the certificate validity period can impersonate access points in real time.	s0014, s0018

ID	Name	About	What the documents say	Where
X02	peer identity confidentiality in the handshake	the peer identity in the client certificate, kept from a passive observer	RFC 9190 s0105/s0108: with TLS 1.3 the certificate messages are encrypted and EAP-TLS is always used with privacy; main s0002 warns that a recorded handshake exposes the client identity once the key exchange is broken.	s0105, s0108, s0004

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

The security statements of the documents, paragraph by paragraph

The host listed the 191 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: 70. Not taken: 121. Which paragraphs are listed follows from the layout of the standard; where each went is the analyst's judgement.

Paragraphs	Where	Taken up as	Why not
P001 to P002	draft-ietf-tls-mldsa:181-192	The prover cannot be asked	signature unforgeability under chosen message attacks and side-channel failures are properties of the primitive and its implementation
P003	main:564-568	Outside the scope	it is about the EST certificate provisioning extensions, which the task excludes
P004 to P006	main:570-582	No goal stated	it only points at the security considerations of other documents; page furniture follows
P007	rfc10024:268-271	A08, C08	
P008	rfc10024:273-276	The prover cannot be asked	side-channel resistance of the KEM implementation is not a property of the symbolic model
P009	rfc10024:278-280	No goal stated	it states that the groups use fixed-length values, which forbids no attack the prover could run
P010 to P011	rfc10024:282-297	The prover cannot be asked	the disclosure of encapsulation randomness and the state of an insecure random number generator are properties of the random number generator
P012	rfc10024:299-306	C08, `x25519_broken`	
P013	rfc10024:308-310	The prover cannot be asked	it is about one insecure random number generator being shared by both algorithms
P014 to P015	rfc3748:2182-2190	No goal stated	it introduces the generic EAP threat model and says why claims sections exist
P016 to P017	rfc3748:2194-2203	`network`	
P018	rfc3748:2205-2206	X02, `network`	
P019	rfc3748:2208-2208	A02, A06, `network`	
P020	rfc3748:2210-2213	R02, `network`	
P021	rfc3748:2215-2216	No goal stated	an offline dictionary attack needs a password chosen by a person; EAP-TLS with certificates has none
P022	rfc3748:2218-2219	A02, `network`	
P023	rfc3748:2221-2222	R05, A08, `network`	
P024	rfc3748:2224-2225	K01, A03	

Paragraphs	Where	Taken up as	Why not
P025	rfc3748:2233-2234	Outside the scope	it is about the ciphersuites used after authentication on the lower layer
P026	rfc3748:2236-2238	Outside the scope	downgrading of the lower-layer ciphersuite is outside the modelled EAP-TLS exchange
P027	rfc3748:2240-2244	Outside the scope	information given out of band through AAA or the lower layer is at the boundary the task excludes
P028	rfc3748:2246-2252	`network`	
P029	rfc3748:2256-2258	No goal stated	it requires a claims section in method specifications
P030	rfc3748:2260-2261	No goal stated	it asks the specification to name its authentication technology
P031	rfc3748:2263-2271	A02, A04, R02, K01	
P032	rfc3748:2273-2276	The prover cannot be asked	the effective key strength in bits is not a question the prover can be asked
P033	rfc3748:2286-2294	The prover cannot be asked	the definition of key strength in bits is computational
P034	rfc3748:2296-2299	No goal stated	a note on the difficulty of defining key strength
P035	rfc3748:2301-2308	The prover cannot be asked	it is about how the key strength depends on the derivation
P036	rfc3748:2310-2313	A03	
P037	rfc3748:2315-2317	No goal stated	it requires a method to say which claims it does not make
P038	rfc3748:2321-2322	No goal stated	a list of terms follows
P039	rfc3748:2324-2328	A08, R05	
P040	rfc3748:2339-2344	A02	
P041	rfc3748:2346-2351	A02, R01	
P042	rfc3748:2353-2355	R02	
P043	rfc3748:2357-2361	K01, X02	
P044	rfc3748:2363-2368	A03, K01	
P045	rfc3748:2370-2374	The prover cannot be asked	key strength in bits
P046	rfc3748:2376-2383	No goal stated	no password is used by EAP-TLS with certificates
P047	rfc3748:2392-2396	Outside the scope	fast reconnect is resumption, which the task excludes
P048	rfc3748:2398-2405	No goal stated	cryptographic binding is about tunnel methods; EAP-TLS is not one
P049	rfc3748:2407-2411	A04, C01	
P050	rfc3748:2413-2417	Outside the scope	fragmentation and reassembly are not modelled
P051	rfc3748:2419-2423	Outside the scope	channel binding needs lower-layer channel properties, which are at the excluded boundary
P052	rfc3748:2425-2427	No goal stated	it says the list of claims is not exhaustive
P053	rfc3748:2431-2434	No goal stated	it says the Identity exchange is optional
P054 to P055	rfc3748:2436-2449	X02	
P056	rfc3748:2451-2455	S04, U02	
P057	rfc3748:2459-2461	Outside the scope	tunnelled EAP methods are excluded by the task

Paragraphs	Where	Taken up as	Why not
P058	rfc3748:2463-2471	No goal stated	it notes that EAP does not permit untunnelled method sequences
P059	rfc3748:2473-2481	Outside the scope	the rogue authenticator that tunnels EAP is outside the modelled full handshake
P060 to P064	rfc3748:2483-2502	Outside the scope	the mitigations are about EAP tunnelling methods, which the task excludes
P065	rfc3748:2506-2508	R01, R02, `network`	
P066	rfc3748:2510-2514	R02, `network`	
P067	rfc3748:2516-2520	A06, `network`	
P068 to P071	rfc3748:2522-2542	No goal stated	per-packet MICs are not used by EAP-TLS; the TLS Finished covers the handshake, which the recorded answers speak about
P072 to P074	rfc3748:2551-2567	A02, R02	
P075	rfc3748:2571-2576	No goal stated	dictionary attacks against password methods
P076	rfc3748:2578-2580	No goal stated	it recommends dictionary-resistant methods
P077	rfc3748:2582-2587	Outside the scope	it recommends tunnelling, which the task excludes
P078	rfc3748:2591-2595	A02	
P079	rfc3748:2597-2598	No goal stated	it notes that EAP does not require full duplex
P080	rfc3748:2604-2611	A02	
P081	rfc3748:2615-2619	R05	
P082	rfc3748:2621-2629	No goal stated	it recommends that one peer name map to one EAP method
P083	rfc3748:2633-2634	No goal stated	implementation dependence of the lower layer
P084	rfc3748:2636-2643	Outside the scope	it is about PPP and 802.1X behaviour on failed authentication
P085	rfc3748:2645-2651	Outside the scope	it is about retries and failure counters of the lower layer
P086	rfc3748:2659-2665	A02, A03	
P087	rfc3748:2667-2671	No goal stated	it says the MSK and EMSK are not used to protect data directly
P088	rfc3748:2673-2677	Outside the scope	AAA-Key derivation on the AAA server is outside the modelled exchange
P089	rfc3748:2679-2682	R01, A04	
P090	rfc3748:2684-2687	No goal stated	it says EAP methods export MSK and EMSK but not transient session keys
P091	rfc3748:2689-2693	No goal stated	it notes that ciphersuite negotiation may happen before or after
P092	rfc3748:2695-2699	R05, A08	
P093	rfc3748:2701-2704	The prover cannot be asked	the strength of the transient session keys is computational
P094	rfc3748:2710-2713	The prover cannot be asked	a 128-bit effective key strength is computational
P095	rfc3748:2715-2720	K01	
P096	rfc3748:2722-2729	The prover cannot be asked	cryptographic separation of substrings is a computational property of the key derivation
P097	rfc3748:2731-2735	K01	

Paragraphs	Where	Taken up as	Why not
P098	rfc3748:2737-2740	Outside the scope	key lifetime handling between the parties is outside the modelled run
P099	rfc3748:2742-2744	No goal stated	it says the document gives no detailed key derivation guidance
P100	rfc3748:2746-2752	No goal stated	it recommends re-using established key derivation functions
P101	rfc3748:2765-2770	Outside the scope	data protection on the lower layer is outside the modelled exchange
P102	rfc3748:2772-2776	Outside the scope	lower-layer protection is outside the modelled exchange
P103	rfc3748:2778-2783	Outside the scope	lower-layer ciphersuite downgrade is outside the modelled exchange
P104	rfc3748:2785-2789	Outside the scope	protecting the lower-layer negotiation is outside the modelled exchange
P105 to P110	rfc3748:2793-2828	Outside the scope	link-layer indications of PPP, IEEE 802 and 802.11 are outside the modelled EAP-TLS exchange
P111	rfc3748:2832-2838	A02, A03	
P112	rfc3748:2840-2842	Outside the scope	the split between authenticator and backend server is the boundary the task puts out of scope
P113	rfc3748:2844-2847	Outside the scope	that the peer cannot validate the authenticator with EAP alone concerns the authenticator identity, not the EAP server authentication modelled here
P114	rfc3748:2849-2855	Outside the scope	protection of the AAA protocol is at the excluded boundary
P115 to P117	rfc3748:2857-2880	Outside the scope	the secure association protocol and the AAA-Key transport are at the excluded boundary
P118	rfc3748:2884-2887	No goal stated	it says cleartext password authentication is not defined
P119	rfc3748:2889-2892	Outside the scope	encapsulation in RADIUS is outside the modelled exchange
P120	rfc3748:2894-2898	No goal stated	cleartext passwords are not used by EAP-TLS
P121 to P127	rfc3748:2902-2945	Outside the scope	channel binding of lower-layer identifiers and out-of-band information is at the boundary the task excludes
P128	rfc3748:2949-2956	A06, R01, `network`	
P129	rfc3748:2958-2963	A06, `network`	
P130	rfc3748:2965-2969	A02, A06, `network`	
P131	rfc3748:2975-2983	The prover cannot be asked	the forged Success packet is a denial of service against the peer, and availability is not what the prover answers
P132	rfc3748:2985-2989	The prover cannot be asked	idle timeouts are a liveness property
P133	rfc3748:2991-2997	A06, U01	
P134	rfc3748:2999-3002	Outside the scope	authorization policy before the success indication is outside the modelled exchange
P135	rfc3748:3004-3014	Outside the scope	synchronization of authorization with the authenticator is at the excluded boundary
P136	rfc3748:3016-3021	A06	
P137	rfc3748:3028-3030	No goal stated	it describes when a peer discards a Failure packet
P138	rfc3748:3032-3037	A06, A02	
P139	rfc3748:3039-3046	The prover cannot be asked	denial-of-service resistance is not a property the prover answers
P140	rfc3748:3048-3056	A06, U01	

Paragraphs	Where	Taken up as	Why not
P141	rfc9190:992-993	A02, K01	
P142 to P143	rfc9190:997-1005	A02	
P144	rfc9190:1007-1012	K01, X02	
P145	rfc9190:1014-1018	C01	
P146	rfc9190:1020-1024	A08, R05	
P147	rfc9190:1028-1029	A02	
P148	rfc9190:1033-1035	A02, `network`	
P149	rfc9190:1039-1040	A02, C01, `ltk_leak`	
P150 to P153	rfc9190:1042-1083	Outside the scope	certificate revocation checking is not part of the modelled message flow
P154	rfc9190:1087-1088	A06, R01, `network`	
P155	rfc9190:1090-1096	A06, U02, `network`	
P156	rfc9190:1098-1104	A06	
P157	rfc9190:1108-1110	Outside the scope	it is about deployments without peer authentication, which the task excludes
P158	rfc9190:1112-1121	A02	
P159	rfc9190:1123-1134	Outside the scope	non-EAP information from the encapsulating protocol is at the excluded boundary
P160	rfc9190:1136-1143	S04, U02, `network`	
P161	rfc9190:1145-1147	Outside the scope	rejecting on non-EAP information is outside the modelled exchange
P162	rfc9190:1149-1155	A02	
P163 to P172	rfc9190:1159-1240	Outside the scope	the whole of section 5.7 is about resumption, which the task excludes
P173 to P174	rfc9190:1244-1249	X02	
P175	rfc9190:1251-1255	X02, `network`	
P176	rfc9190:1257-1272	The prover cannot be asked	leakage through packet sizes and padding is a length side channel, not a symbolic property
P177	rfc9190:1274-1280	X02	
P178	rfc9190:1282-1293	X02, R05, `network`	
P179	rfc9190:1295-1312	X02	
P180	rfc9190:1316-1316	No goal stated	it introduces the section on pervasive monitoring
P181	rfc9190:1318-1328	X02	
P182	rfc9190:1330-1335	C01	
P183	rfc9190:1339-1339	No goal stated	it introduces the list of earlier TLS attacks
P184	rfc9190:1341-1351	No goal stated	it tells implementations to follow current guidance and points at documents
P185	rfc9190:1355-1355	No goal stated	it introduces cross-protocol resumption

Paragraphs	Where	Taken up as	Why not
P186 to P187	rfc9190:1357-1372	Outside the scope	cross-protocol resumption is resumption, which the task excludes
P188	rfc9191:487-490	A02	
P189	rfc9191:492-493	Outside the scope	certificate compression is not modelled
P190	rfc9191:495-496	No goal stated	it points at the security considerations of the referenced documents
P191	rfc9846:4676-4677	A02, K01	

The parties and the protected values, one by one

The host listed the 31 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: 25. 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	EAP peer	party	`ltk_leak`	
S02	EAP server	party	`ltk_leak`	
S03	EAP authenticator	party	Nothing gained	the authenticator only relays EAP packets: the MSK it receives is lost by definition for the secrecy targets, and it holds nothing that lets it forge the TLS handshake, so what remains is exactly the network attacker the authentication targets are answered under
S04	peer certificate and private key	value	`ltk_leak`	
S05	server certificate and private key	value	`ltk_leak`	
S06	peer identity	value	Everything lost by definition	holding the peer identity is exactly the failure of the peer identity confidentiality goal, which is answered under the network attacker
S07	transcript hash	value	Nothing gained	the transcript hash is a public handshake value
S08	exporter_secret	value	Everything lost by definition	the exporter secret yields the MSK, so the secrecy goals that speak about it are lost by definition; the authentication goals do not depend on it and are answered under the network attacker
S09	Key_Material/MSK	value	Everything lost by definition	the MSK is the value the secrecy goals speak about; holding it loses those goals by definition
S10	Finished verify_data	value	Nothing gained	the Finished verify_data travels in the handshake
S11	transcript	value	Nothing gained	the transcript is public handshake data
S12	session key	value	Everything lost by definition	the traffic keys of the TLS session are what the secrecy goals protect
S13	Finished	value	Nothing gained	the Finished message travels in the handshake
S14	negotiated group	value	Nothing gained	the negotiated group is public
S15	signature algorithm	value	Nothing gained	the negotiated signature algorithm is public
S16	MSK	value	Everything lost by definition	MSK secrecy is the goal itself
S17	EMSK	value	Everything lost by definition	the EMSK comes from the same Key_Material as the MSK; holding it loses the secrecy goal about the exported keying material

ID	Party or value	Kind	Attacker	Why none
S18	Key_Material	value	Everything lost by definition	Key_Material is the exported keying material the secrecy goals speak about
S19	peer private key	value	`ltk_leak`	
S20	server private key	value	`ltk_leak`	
S21	session payload	value	Everything lost by definition	session payloads travel under the session keys; holding them is the failure of the confidentiality goal
S22	shared secret	value	Everything lost by definition	the hybrid shared secret yields every session key, so the secrecy goals are lost by definition; it is what the two component attackers are given separately
S23	0x00 success indication	value	Nothing gained	the 0x00 indication is the peer's own success signal, not a secret
S24	random number	value	Nothing gained	the fresh values travel in the handshake
S25	client Finished	value	Nothing gained	the client Finished travels in the handshake
S26	negotiated version	value	Nothing gained	the negotiated version is public
S27	key exchange group	value	Nothing gained	the key exchange group is public
S28	AEAD key	value	Everything lost by definition	AEAD keys are derived from the session keys; holding them loses the confidentiality goals by definition
S29	nonce/IV	value	Nothing gained	nonces travel with the records
S30	EAP identity	value	Nothing gained	the outer EAP identifier is sent in the clear in the Identity Response
S31	TLS authenticated identity	value	Nothing gained	the TLS authenticated identity is the certificate identity, known to both parties and used for routing and authorization

The properties under each attacker

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

Attacker	Has	Basis	Where
`network`	the network only: it reads, drops, changes and sends messages, and holds no secret of a party		
`ltk_leak`	the ML-DSA-65 private keys of the peer certificate and of the server certificate, recovered before or during a session	main s0014 and s0018: within the certificate validity period an attacker that recovers the private key can impersonate the certificate holder in real time; RFC 9190 section 5.4 names key compromise as a revocation reason.	s0014, s0018
`x25519_broken`	the X25519 part of the hybrid key agreement of a recorded handshake (the ephemeral scalars or the X25519 shared secret)	main s0012: a hybrid shared secret remains secure only as long as either component algorithm remains unbroken; RFC 10024 s0644 notes that a CRQC observer can recover the ECDH scalar.	s0012, s0652
`mlkem_broken`	the ML-KEM-768 part of the hybrid key agreement of a recorded handshake (the shared secret or the decapsulation key)	main s0012 and s0010/s0652: the ML-KEM-768 component of X25519MLKEM768.	s0012, s0652

ID	network	ltk_leak	x25519_broken	mlkem_broken
A02	Fails	Fails		
A03	No answer			
A04	No answer			
A06	No answer			
A08	No answer			
K01	Holds			
C01		No answer		
C08			No answer	No answer
R01	No answer			
R02	No answer			
R05	No answer			
S01	No answer			
S04	No answer			
A03.14	No answer			
U01	Holds			
U02	No answer			
X01		Fails		
X02	Holds			

Source and targets

Source: 655 units, 182927 bytes, frozen in source-delivery-01.json. Targets: 15 approved (A02, A03, A04, A06, A08, K01, C01, C08, R01, R02, R05, S01, S04, U01, U02); the questions asked: 3. The list of targets was approved as drafted, beforehand, with --approve-targets.

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

- approve_package: approved source package protocol-dd1026aa855b81ab805a71240f38

This step as it reports itself: 1 requests (1 failed), 0.000 USD, 80177 tokens in and 5710 out, 32 s.

Notes on the prover and the model

- A02: answered in ProVerif, and the analyst names dh_algebra for this target
- A03: answered in ProVerif, and the analyst names dh_algebra for this target
- A08: answered in ProVerif, and the analyst names dh_algebra for this target
- K01: answered in ProVerif, and the analyst names dh_algebra for this target
- C01: answered in ProVerif, and the analyst names dh_algebra for this target
- C08: answered in ProVerif, and the analyst names dh_algebra for this target
- R05: answered in ProVerif, and the analyst names dh_algebra for this target
- S01: answered in ProVerif, and the analyst names dh_algebra for this target
- S04: answered in ProVerif, and the analyst names dh_algebra for this target

- U01: answered in ProVerif, and the analyst names dh_algebra for this target
- U02: answered in ProVerif, and the analyst names dh_algebra for this target
- X01: answered in ProVerif, and the analyst names dh_algebra for this target
- X02: answered in ProVerif, and the analyst names dh_algebra for this target

A ProVerif model that simplifies algebra (xor, dh_algebra, other_algebra) gives the attacker less than the real operation: a proof there is a proof against a weaker attacker, an attack there is an attack. A ProVerif model that leaves out a check on a value that changes (changing_state) lets parties accept more: a proof there holds, an attack there may be nothing but the missing check. Tamarin+ Agent models these as they are.

What the analyst named for targets:

step	targets	features	reason
31	A02, A03, A04, A06, A08, K01, C01, C08, R01, R02, R05, S01, S04, A03.14, U01, U02, X01, X02	dh_algebra, plain_primitives, many_sessions	The hybrid shared secret is written as a pair of a KEM secret and a Diffie-Hellman term with the single law g^xy = g^yx, so every answer that mentions the session keys or the transcript rests on that simplification; ProVerif does not identify the peer's exp(exp(g,x),y) with the server's exp(exp(g,y),x) inside the transcript hash, which is where the inconclusive and several falsified results come from. The key derivation, the AEAD, the KEM and the signatures are otherwise plain primitives, modelled with their usual perfect-cryptography laws. The properties are asked over any number of sessions of both roles, which ProVerif does cover, but the fingerprints of the runs are hashed together, so agreement on one run is not decidable in this shape.

The analyst's closing note

A model of the plain run (peer, EAP server, X25519MLKEM768 hybrid shared secret, TLS 1.3 key schedule, TLS exporter, Finished over the transcript, protected 0x00 success indication, EAP-Success) is in model.pv and its normal run is found. All 18 targets have an answer or an explicit record that none came back. Proved for any number of sessions: EAP-Success follows both the accepted client Finished and the protected 0x00, the certificate signing keys are not obtainable from the network, and the identity in the client certificate is not obtained by a passive attacker. Two attacks reproduce what the documents state: an attacker holding the recovered server signing key impersonates the EAP server to a peer, and an attacker on the network alone makes a peer accept an EAP-Success the server never sent, since EAP-Success carries no protection. No new weakness was found. Six target and attacker pairs stay open and a number of answers are marked as artifacts of the model: the hybrid shared secret is written with only the law g^xy = g^yx and ProVerif does not identify the peer's and the server's spellings of it inside the transcript hash, so agreement on one run, injectivity, freshness, the negotiated parameters, the nonce rule and the hybrid component claims could not be decided; the model also lets any attacker take the modelled component disclosures, which is why the forward-secrecy and hybrid targets have no usable answer. What blocks the rest is that arithmetic, which Tamarin+ Agent would settle, and a model that separates the disclosures by phase. No bounded model and no state review were needed: no target rests on a value that changes.