Security analysis of IKEv2 ML-DSA authentication

This report in another language: 中文.

draft-ietf-ipsecme-ikev2-pqc-auth-12 · ProVerif+ Agent report

Documents	Signature Authentication in the Internet Key Exchange Version 2 (IKEv2) using PQC (protocol definition); Internet Key Exchange Protocol Version 2 (IKEv2) (supporting); Signature Authentication in the Internet Key Exchange Version 2 (IKEv2) (supporting)
Properties examined	13: 12 hold, 1 fail (1 attack)
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 draft adds post-quantum authentication to the IKEv2 initial exchange. An initiator and a responder run IKE_SA_INIT, where they agree on an SA, exchange Diffie-Hellman shares and nonces, and announce the hash algorithms they support for signatures. Each party holds an ML-DSA-65 signing key and a trust anchor for the other's certificate. In IKE_AUTH each party signs a block built from the first-round message it received, the other side's nonce and a PRF of its own identity, and sends the signature with its identity inside the encrypted and integrity-protected payload.

When the exchange ends, each side is sure that the peer holds the key named in the certificate it accepted, that the peer's signature covers both nonces and both IKE_SA_INIT messages, and that the two of them share the same IKE SA keys. The session key is kept from anyone on the network, and stays secret from an attacker who later obtains both signing keys.

The parties:

- **Initiator**: proposes the SA, signs first. Holds: ML-DSA-65 signing key; responder's trust anchor; its identity IDi.
- **Responder**: selects the SA, answers. Holds: ML-DSA-65 signing key; initiator's trust anchor; its identity IDr.

Figure 1 shows the exchange as the documents specify it.

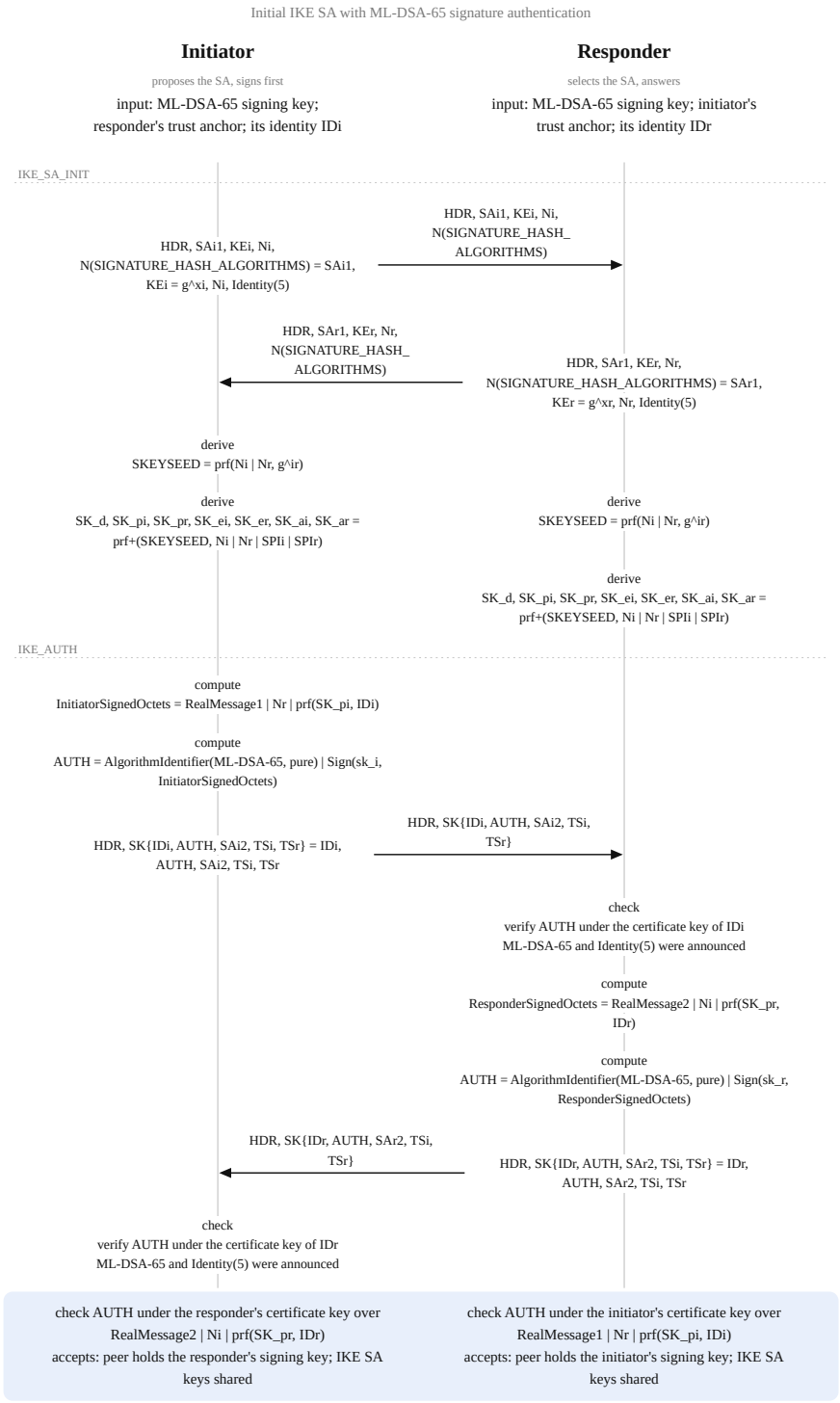


Figure 1. The protocol as specified.

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.

No finding: every guarantee asked about holds in the model, and both failures are ones the documents state themselves. Mutual ML-DSA-65 authentication, the binding of identities, both nonces and both IKE_SA_INIT messages, single-use agreement, and the refusal of traditional signatures in a PQC-required deployment all hold for any number of sessions, also against an attacker that can forge every traditional signature. Two known weaknesses are reproduced: an on-path attacker reads the initiator's identity from IKE_AUTH when it is accepted as the Diffie-Hellman peer, and a deployment that allows fallback is impersonated outright by a traditional-signature forger. Most answers rest on a model that idealises Diffie-Hellman to its single law, so the forward-secrecy and session-key results hold under that idealisation and need a prover with the DH algebra to be settled as they are. All thirteen targets have an answer, and every proof carries a control that fails.

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	**peer identity authentication.** Whenever the initiator (or the responder) accepts the peer's identity in IKE_AUTH, there must be a genuine run of the peer with this party's identity as its communication target.	no compromise	Holds	medium	yes
A03	**session binding of the AUTH payload.** Whenever a party accepts, there is a matching peer run, and both sides agree on each other's identities, Ni, Nr and the contents of the first-round IKE_SA_INIT messages.	no compromise	Holds	medium	yes
A04	**no replay of AUTH across sessions.** Every acceptance by this party corresponds to a distinct genuine peer run (injective matching); the same peer run cannot be accepted more than once.	no compromise	Holds	medium	yes
A08	**signature algorithm binding.** The signature algorithm and hash algorithm accepted by the receiver must come from the peer's genuine proposal, from capability signals covered by AUTH, and must match the algorithm actually used to verify the signature.	no compromise	Holds	medium	yes
R05	**no downgrade to traditional AUTH.** In a deployment that requires PQC signature authentication, the receiver never accepts a session that falls back to traditional signature authentication.	no compromise	Holds	medium	yes
K01	**IKE SA and signing key secrecy.** The attacker cannot derive the IKE SA session keys (SKEYSEED and its derived keys) or either party's long-term ML-DSA signing private key.	no compromise	Holds	low	yes
C01	**forward secrecy of the IKE SA key.** After the session completes and is erased, leaking both parties' long-term signing private keys does not let the attacker obtain that session's IKE SA keys.	both parties' long-term ML-DSA signing private keys (after the session completes)	Holds	medium	yes
C02	**no impersonation from own key leak.** After the receiver's own ML-DSA signing private key leaks, and provided this party still executes honestly, the attacker cannot impersonate the uncompromised peer to make this party accept.	the receiver's ML-DSA signing private key (leaked before the session is established)	Holds	low	no

ID	Property	Compromise assumed	Result	Value	In the documents
C08	**PQC auth survives broken legacy signatures.** When the traditional signature primitives are broken (the attacker can forge traditional signatures) while ML-DSA signatures remain secure, identity authentication and algorithm binding still hold in sessions that enforce PQC authentication.	traditional signature algorithms' signing private keys (before the session is established)	Holds	medium	yes
A03.10	**AUTH binds identities and nonces.** Whenever a party accepts, there is a matching peer run, and both sides agree on each other's identities, Ni, Nr and the contents of the first-round messages.	no compromise	Holds	medium	yes
A08.11	**modified capability signals are detected.** The signature algorithm and hash algorithm accepted by the receiver must come from the peer's genuine proposal, from capability signals covered by AUTH, and must match the algorithm actually used to verify the signature.	no compromise	Holds	medium	unclear
C02.12	**own key leak does not break peer identity.** After the receiver's own ML-DSA signing private key leaks, and provided this party still executes honestly, the attacker cannot impersonate the uncompromised peer to make this party accept; a normal session also exists as a positive example.	the receiver's ML-DSA signing private key (leaked before the session is established)	Holds	low	no
X01	**identity protection of IDi and IDr.** An attacker that only observes and modifies the network never learns the ID payload contents of the IKE_AUTH exchange, so it cannot tell which identity the initiator claimed or which identity the responder used.	no compromise	Attack holds (stated in the documents)	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.

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: binding of AUTH to identities, nonces and first-round messages	Yes: the AUTH payload binds both identities, both nonces and both IKE_SA_INIT messages, in the initiator-to-responder and responder-to-initiator directions, for any number of sessions, and also when traditional signatures can be forged.	A03, A03.10, A02, A04
Q2: whether a successful session still verifies against the actual signature when the algorithm capability signals are modified	Yes, with one caveat: the SIGNATURE_HASH_ALGORITHMS and SUPPORTED_AUTH_METHODS notifications are inside the signed data, so modifying them breaks authentication and a deployment that requires PQC never accepts a traditional AUTH; but the AlgorithmIdentifier inside the Authentication Data is not itself under the signature, so the receiver's choice of verification algorithm rests on an attacker-modifiable field whose only protection is that verification under the wrong algorithm fails.	A08, A08.11, R05, C08
Q3: impact of a PQ signing key leak on identity authentication, and the positive control	A leaked own signing key does not let an attacker impersonate the other, uncompromised peer to the accepting party, and a positive control shows the same configuration completes a normal session; separately, an attacker who obtains both signing keys after a session learns no key material of that session.	C02, C02.12, C01

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:

- Diffie-Hellman is idealised to its single law g^xy = g^yx; no algebraic attack on the exchange is represented, and the model has no quantum assumption about the DH secret. Tamarin+ Agent, which models DH as it is, would settle the forward-secrecy questions.
- Certificates and trust anchors are taken as given: each role receives the peer's public key and the identity that key belongs to, so certificate-chain validation is not analysed.
- The Child SA negotiation inside IKE_AUTH (SAi2, TSi, TSr) and everything after it, including the CREATE_CHILD_SA and INFORMATIONAL exchanges, is outside the model, as are EAP and rekeying.
- Message IDs, the window for overlapping requests, retransmissions and fragmentation are not modelled; the nonces carry the freshness argument.
- The IKE_AUTH payloads are protected with an idealised encryption and MAC, so AEAD nonce reuse and the details of the SK_e and SK_a key schedule are not analysed.
- Traditional signatures are modelled as an unbroken policy choice whose keys the attacker holds; no attack on a specific legacy algorithm is analysed.
- The first-round Diffie-Hellman share is not validated in the model, because the documents do not require it; the identity-protection attack depends on that, and a party that validates the peer's share would stop it.
- Only the initial exchange between two peers is modelled. Scenarios with a security gateway, an endpoint, NAT traversal and the PAD/traffic-selector authorization of RFC 7296 Section 5.1 are not covered.

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: peer identity authentication

Whenever the initiator (or the responder) accepts the peer identity in IKE_AUTH, a real run of the peer must exist that has this party's identity as its communication target.

Both parties authenticate each other by ML-DSA-65 signature, and neither side accepts a peer identity without a matching run by that peer. The result also holds for an attacker that can forge every traditional signature.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A central guarantee of the draft, but one nobody doubted; its worth is that it is now mechanical and survives a CRQC attacker for traditional signatures.
- Stated in the documents: yes. Where the documents say it: s0103, s0387
- Rests on: Certificates and trust anchors are assumed: each role is given the peer's public key and the identity that key belongs to, and certificate chains are not checked.; The model contains one honest pair; the queries speak of that pair by name.; The attacker in this model knows every traditional signature key, so the proof is against a stronger attacker than the network alone.

A03: session binding of the AUTH payload

Whenever one side accepts, a matching peer run exists, and both sides agree on the mutual identities, Ni, Nr, and the contents of the first-round IKE_SA_INIT messages.

The signature really does bind what the draft says it binds: both identities, both nonces and the two IKE_SA_INIT messages. A message modified on the way, in either direction, breaks the AUTH and the session stops at that check.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Confirms in a prover the property the draft argues for in prose.
- Stated in the documents: yes. Where the documents say it: s0387, s0389, s0391
- Rests on: The signed blocks are modelled as the draft describes them; length, SPI and fragmentation details are left out.; The responder's side is stated with the initiator's running event, because the responder accepts before the initiator does.; The attacker is given all traditional signature keys.

A04: no replay of AUTH across sessions

Every acceptance by this party corresponds to a distinct real peer run (injective matching); the same peer run cannot be accepted more than once.

Accepting a session cannot be made to count twice: every acceptance maps to one distinct peer run, so a recorded AUTH cannot be replayed into a second session. This is the nonce argument of the draft, shown mechanically.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The draft states it in prose; the model shows it for any number of sessions.
- Stated in the documents: yes. Where the documents say it: s0012, s0045, s0387

- Rests on: Message identifiers and the message-ID window are not modelled; only the nonces make each session unique here.; Child SA creation in messages 3 and 4 is an opaque constant.

A08: signature algorithm binding

The signature algorithm and hash algorithm accepted by the receiver must come from the peer's real proposal and from a capability signal covered by the AUTH, and must match the algorithm actually used to verify the signature.

In a deployment that requires ML-DSA-65 no session is ever accepted with a traditional signature, and the algorithm and hash a receiver uses come from the peer's own messages, which the AUTH covers. Modifying the SIGNATURE_HASH_ALGORITHMS notification therefore only breaks the exchange.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Answers the draft's intent, with the caveat that the algorithm identifier in the payload is not covered by any signature.
- Stated in the documents: yes. Where the documents say it: s0019, s0020, s0044, s0557
- Rests on: Algorithms are distinct symbolic constructors; algorithm confusion on real signature schemes is outside the model.; The strict configuration is a check on the responder's policy; the loose configuration is a second instance of the same party.; Only the Identity hash is modelled for ML-DSA; the traditional algorithm is a single other choice.
- Open: The AlgorithmIdentifier inside the Authentication Data is not itself under the signature, so a receiver that picks the verification algorithm from the payload trusts an attacker-modifiable field; the symbolic model can only show that verification under a relabelled algorithm fails.

R05: no downgrade to traditional AUTH

In deployments that require PQC signature authentication, the receiver does not accept a session that falls back to traditional signature authentication.

With PQC authentication required and no fallback, an attacker that can forge traditional signatures gains nothing: no session is established with a traditional AUTH. The same attacker succeeds at once when fallback is allowed, which is the attack the draft describes.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The draft's recommendation is confirmed as a check, and its warning is reproduced in the control.
- Stated in the documents: yes. Where the documents say it: s0044
- Rests on: The policy is a check in the accepting party, as the draft requires of deployments; how an operator configures it is not analysed.; Only two configurations are modelled: PQC required, and fallback allowed.

K01: IKE SA and signing key secrecy

The attacker cannot derive the session keys of the IKE SA (SKEYSEED and the keys derived from it) or either party's long-term ML-DSA signing private key.

Neither the IKE SA key material of an authenticated session nor either ML-DSA signing key reaches the attacker during the run. The session-key secrecy is claimed only for sessions that completed authentication, and the control shows that a session whose AUTH has not been checked yet has its seed known to an attacker that answered IKE_SA_INIT itself.

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

- Value low: Follows from the idealised DH and from keys that are never sent; the real question needs the DH algebra and a quantum assumption.
- Stated in the documents: yes. Where the documents say it: s0379, s0382
- Rests on: Diffie-Hellman is idealised to its single law, so no algebraic attack on the exchange is represented.; The session-key event is emitted only after the AUTH checks, so a half-open exchange is not counted as a session.
- Open: The secrecy of the session keys rests on the peer's Diffie-Hellman share; the model idealises DH to g^xy = g^yx and says nothing about the quantum security of the exchange.

C01: forward secrecy of the IKE SA key

After a session completes and is erased, leaking both parties' long-term signing private keys must not let the attacker obtain that session's IKE SA keys.

An attacker who obtains both ML-DSA signing keys after a session gets no key material of that session: the session key comes from the ephemeral exchange and the nonces, not from the signing keys. The control shows that the leak itself works.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A central guarantee that holds in the model; its weight depends on the DH idealisation, which the scope fixes.
- Stated in the documents: yes. Where the documents say it: s0379, s0381, s0363
- Rests on: Diffie-Hellman is idealised and the ephemeral exponents are never output, which is what the source assumes.; The leak happens in a later phase, after every exchange of the modelled run is over.; Quantum resistance of the DH exchange is neither claimed nor analysed.
- Open: Whether the answer survives a Diffie-Hellman algebra that allows more than the commutative law, which is where a real forward-secrecy argument has to live.

C02: no impersonation from own key leak

After the receiver's own ML-DSA signing private key leaks, and provided this party still executes honestly, the attacker cannot impersonate an un-leaked peer and make this party accept.

A leaked own signing key does not let the attacker pose as the other, honest peer: acceptance at the compromised party still implies that the other peer ran a matching session. Key compromise impersonation is therefore not possible, which is what signing the peer's nonce under one's own key promises.

- Result: Holds; answers what is asked: yes; usable
- Value low: The draft asks nothing about key-compromise impersonation; the result follows directly from the signature construction, so it is cheap to state but surprises nobody.
- Stated in the documents: no. Source text it rests on: s0387, s0103
- Rests on: The compromised party is a separate honest pair from the pair used for A02/A03, so the leak cannot be confused with the ordinary-authentication answers.; The peer identity and key are given to the party by configuration, so certificate validity is assumed.; Only the accepting party's signing key is given to the attacker; its Diffie-Hellman exponents stay secret.

C02.12: own key leak does not break peer identity

After the receiver's own ML-DSA signing private key leaks, and provided this party still executes honestly, the attacker cannot impersonate an un-leaked peer and make this party accept; at the same time a normal session exists as a positive example.

The same conclusion as the key-compromise claim, with the normal session shown to be reachable in the same configuration, so the proof is not about a situation that cannot arise.

- Result: Holds; answers what is asked: yes; usable
- Value low: A control for the previous claim rather than a result of its own.
- Stated in the documents: no. Source text it rests on: s0387, s0103
- Rests on: The positive control is one reachable session, not a proof about an arbitrary one.

C08: PQC auth survives broken legacy signatures

When the traditional signature primitive is broken (an attacker can forge traditional signatures) while ML-DSA signatures remain secure, identity authentication and algorithm binding still hold in sessions that enforce PQC authentication.

With every traditional signature key in the attacker's hands, sessions in a deployment that requires ML-DSA-65 are still authenticated by ML-DSA-65 alone, and the peer identity is still bound. The broken legacy algorithm only affects deployments that allow fallback, where the draft says the attack works.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Confirms the draft's own remedy under its own attack model: Section 9's argument made mechanical.
- Stated in the documents: yes. Where the documents say it: s0044, s0038
- Rests on: The ability to forge traditional signatures is modelled by giving the attacker those keys outright, which is the draft's assumption.; ML-DSA-65 is an unbreakable symbolic signature; its computational security is not analysed.

A03.10: AUTH binds identities and nonces

Whenever one party accepts, a matching peer run exists, and both parties agree on each other's identities, Ni, Nr, and the contents of the first-round messages.

The AUTH payload binds the identities, both nonces and both IKE_SA_INIT messages, so this part of question Q1 is answered yes.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Answers the question asked; the substance is the same as A03.
- Stated in the documents: yes. Where the documents say it: s0387, s0389, s0391
- Rests on: As for A03: the signed blocks are modelled from the draft's definition, without the SPI and fragmentation details.

A08.11: modified capability signals are detected

The signature algorithm and hash algorithm accepted by the receiving party must come from capability signals that were genuinely proposed by the peer and covered by AUTH, and must match the algorithm actually used to verify the signature.

Modifying a capability notification does not produce an accepted session with an algorithm other than the one really announced and verified: the notifications are inside the signed data, and a deployment that requires PQC accepts only ML-DSA-65.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Answers question Q2 affirmatively; the documents say the notifications are AUTH-protected but say nothing about the algorithm identifier field of the AUTH payload itself, so the part that could matter is left open.
- Stated in the documents: unclear. Source text it rests on: rfc7296:489, s0019, s0020, s0044
- Rests on: As for A08: algorithm identifiers are distinct symbolic constructors, so cross-algorithm confusion cannot be represented.
- Open: Whether an attacker can relabel the AlgorithmIdentifier inside the AUTH payload and have the signature verify under the named algorithm: a question about signature schemes, outside the model.

X01: identity protection of IDi and IDr

An attacker that only observes and modifies the network never learns the ID payload contents of the IKE_AUTH exchange, so it cannot tell which identity the initiator claimed or which identity the responder used.

The claim fails: an attacker on the path reads the initiator's identity out of the IKE_AUTH request. It answers IKE_SA_INIT with a Diffie-Hellman share of g, so the shared secret is the initiator's own public value, derives the encryption key and decrypts the request. No AUTH is completed in that trace, which is exactly the caveat RFC 7296 states.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value medium: A known consequence is reproduced, but the mechanism is now explicit: it is the unvalidated Diffie-Hellman share, not a weakness of the ML-DSA authentication.
- Stated in the documents: yes. Where the documents say it: rfc7296:489, s0084, s0379, s0381
- Rests on: The attack needs the degenerate share g to be accepted; a party that validates the peer's Diffie-Hellman public value would stop it, and the model has no such check because the documents do not require one.; The identity is treated as a name the parties hold, not one the attacker already knows.; No EAP or later exchange is modelled, and the identity of the initiator is not otherwise exposed in the modelled run.

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		a02_i	checked		4 proved
A03	proved		a03_i	checked		3 proved
A04	proved		a04_i	checked		3 proved
A08	proved		a08_notrad	checked		4 proved
R05	proved		a08_notrad	checked		2 proved
K01	proved		k01_sess	checked		3 proved
C01	proved		c01	checked		1 proved
C02	proved		c02_r	checked		2 proved
C08	proved		a08_notrad	checked		2 proved
A03.10	proved		a03_i	checked		2 proved
A08.11	proved		a08_notrad	checked		3 proved
C02.12	proved		c02_r	checked		2 proved
X01	none				x01 (holds_in_source)	1 attack

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

target	claim	attacker	answer	query	control
A02	initiator side: whenever the initiator accepts peer id_b's identity, some responder run by id_b has accepted id_a as peer (non-injective agreement on the peer identity); the attacker has the network only	`network`	proved	a02_i	checked
A02	responder side: whenever the responder accepts id_a as its peer, the initiator ran a session with this responder as peer and with the same Ni, Nr and IKE_SA_INIT request (the responder accepts before the initiator does, so the running event is the matching one)	`network`	proved	a02_r	checked
A02	initiator side, with every traditional signature key in the attacker's hands: the peer identity is still authenticated by ML-DSA-65 only	`crqc_traditional`	proved	a02_i	checked
A02	responder side, with every traditional signature key in the attacker's hands: the peer identity and first-round messages are still authenticated by ML-DSA-65 only	`crqc_traditional`	proved	a02_r	checked
A03	initiator side: the responder's acceptance agrees with the initiator's on both identities, Ni, Nr and the contents of IKE_SA_INIT request and response, so the AUTH covers the peer's nonce and the first-round messages	`network`	proved	a03_i	checked
A03	responder side: the initiator's run and the responder's acceptance agree on the peer identity, Ni, Nr and the IKE_SA_INIT request contents	`network`	proved	a02_r	checked

target	claim	attacker	answer	query	control
A03	initiator side with traditional signatures forgeable: the peer's nonce, identities and both IKE_SA_INIT messages are still bound by ML-DSA-65 AUTH	`crqc_traditional`	proved	a03_i	checked
A04	initiator side: each acceptance of the responder corresponds to a distinct responder acceptance (injective agreement), so a recorded AUTH is not counted twice	`network`	proved	a04_i	checked
A04	responder side: each acceptance of the initiator corresponds to a distinct initiator run (injective agreement)	`network`	proved	a04_r	checked
A04	initiator side with traditional signatures forgeable: no recorded AUTH is accepted twice, each acceptance maps to a distinct peer acceptance	`crqc_traditional`	proved	a04_i	checked
A08	a responder whose deployment requires PQC authentication never accepts an AUTH made with a traditional signature algorithm, whatever algorithm identifier the AUTH payload carries	`network`	proved	a08_notrad	checked
A08	the algorithm and hash the accepting party uses are the peer's genuine, AUTH-covered signal: the peer's own run carries the same IKE_SA_INIT messages that the AUTH was verified over, so a modified notification cannot be authenticated	`network`	proved	a03_i	checked
A08	with traditional signatures forgeable, a PQC-required deployment still accepts only ML-DSA-65 AUTH, so no downgrade of the algorithm identifier is accepted	`crqc_traditional`	proved	a08_notrad	checked
A08	the algorithm signal is the peer's genuine one even when traditional signatures are forgeable, because the AUTH covers both IKE_SA_INIT messages	`crqc_traditional`	proved	a03_i	checked
R05	in a deployment that requires ML-DSA authentication and forbids fallback, no session is accepted with a traditional signature authentication, so an attacker cannot force a downgrade	`network`	proved	a08_notrad	checked
R05	no downgrade: with a CRQC forger on the network a PQC-required deployment establishes no session authenticated with a traditional signature	`crqc_traditional`	proved	a08_notrad	checked
K01	the responder's ML-DSA signing key is never known to the attacker during the run	`network`	proved	k01_ltk2	checked
K01	the IKE SA key material (SKEYSEED and the seven derived keys) of an authenticated session is never known to the attacker during the run	`network`	proved	k01_sess	checked
K01	the initiator's ML-DSA signing key is never known to the attacker during the run	`network`	proved	k01_ltk	checked
C01	no IKE SA key material of an established session reaches an attacker who obtained both ML-DSA long-term signing keys after the exchange (SKEYSEED is derived from the ephemeral DH secret and the nonces, not from the signing keys)	`ltk_after_session`	proved	c01	checked
C02	with the accepting party's own ML-DSA signing key known to the attacker from before the session, that party still accepts the peer's identity only if the peer ran a matching session, so the attacker cannot impersonate the uncompromised peer to it	`own_ltk`	proved	c02_r	checked

target	claim	attacker	answer	query	control
C02	weaker reading of the same claim: with its own signing key public, the accepting party's every acceptance still implies that the peer ran some session with it as peer	`own_ltk`	proved	c02_dn	checked
C08	with every traditional signature key known to the attacker, a deployment requiring PQC authentication still never accepts a traditional AUTH, so the broken legacy algorithm gives no way in	`crqc_traditional`	proved	a08_notrad	checked
C08	with traditional signatures forgeable, ML-DSA-65 still authenticates the peer's identity on the responder side	`crqc_traditional`	proved	a02_r	checked
A03.10	AUTH binds the peer identity, both nonces and the IKE_SA_INIT messages: the responder's acceptance agrees with the initiator's session on all of them (question Q1)	`network`	proved	a03_i	checked
A03.10	AUTH still binds identities and nonces when traditional signatures are forgeable (question Q1)	`crqc_traditional`	proved	a03_i	checked
A08.11	a modified capability signal fails: the responder accepts the peer's AUTH only with the Identity hash announced in the signed IKE_SA_INIT request and the ML-DSA-65 identifier verified (question Q2)	`network`	proved	a08_notrad	checked
A08.11	the peer's algorithm signal is genuine and a PQC-required deployment accepts only ML-DSA-65, even with traditional signatures forgeable (question Q2)	`crqc_traditional`	proved	a08_notrad	checked
A08.11	the accepted algorithm and hash come from the peer's own AUTH-covered IKE_SA_INIT messages (question Q2)	`network`	proved	a03_i	checked
C02.12	same claim as the first one, with the normal session given as the positive control: the acceptance under the leak is not vacuous, a session between the two parties is reachable	`own_ltk`	proved	c02_r	checked
C02.12	weaker reading of the same claim (any peer run, not only the one with the same nonces), with the positive control of a reachable session (question Q3)	`own_ltk`	proved	c02_dn	checked
X01	the IDi value the initiator puts in the IKE_AUTH request is not hidden from an on-path attacker: it can read it before any AUTH verification	`network`	attack (holds_in_source)	x01	

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-20261114T222941, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 40, prover 27 s, wall 14 min, cost 0.126 USD.

ProVerif: 108 proofs run, 108 decided, 0 without an answer, 21 s, 38 answers recorded.

What was analysed

Protocol	IKEv2 ML-DSA authentication
Version	draft-ietf-ipsecme-ikev2-pqc-auth-12
Analysed	Fixed ML-DSA-65, RFC7427 Digital Signature AUTH and the initial IKE SA, the complete SignedOctets, identity and algorithm selection; DH is idealised, and no claim is made that the DH session keys are quantum-resistant on the strength of the signatures; EAP and rekey are not covered.
Not analysed	The symbolic model does not prove the computational or quantum resistance of the algorithms; an attack trace holds only under explicit assumptions and is not directly the same as a flaw in the specification. Every property needs a reachability/positive control.
Source text	573 numbered passages (187263 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	General Framework for PQC Authentication in IKEv2
main	3.1	Specifying PQC Signature Algorithms
main	3.2	Signature Generation and Verification
main	3.2.1	Handling PQC Signatures in IKEv2
main	3.3	Mechanisms for Signaling Supported Key Pair Types
main	4	Specifying ML-DSA within IKEv2
main	6	Use of ML-DSA and SLH-DSA
main	9	Security Considerations
rfc7296	1	Introduction
rfc7296	1.1	Usage Scenarios
rfc7296	1.1.1	Security Gateway to Security Gateway in Tunnel Mode
rfc7296	1.1.2	Endpoint-to-Endpoint Transport Mode
rfc7296	1.1.3	Endpoint to Security Gateway in Tunnel Mode
rfc7296	1.1.4	Other Scenarios
rfc7296	1.2	The Initial Exchanges
rfc7296	1.3	The CREATE_CHILD_SA Exchange
rfc7296	1.3.1	Creating New Child SAs with the CREATE_CHILD_SA Exchange

Document	Section	Title
rfc7296	1.3.2	Rekeying IKE SAs with the CREATE_CHILD_SA Exchange
rfc7296	1.3.3	Rekeying Child SAs with the CREATE_CHILD_SA Exchange
rfc7296	1.4	The INFORMATIONAL Exchange
rfc7296	1.4.1	Deleting an SA with INFORMATIONAL Exchanges
rfc7296	1.5	Informational Messages outside of an IKE SA
rfc7296	1.6	Requirements Terminology
rfc7296	1.7	Significant Differences between RFC 4306 and RFC 5996
rfc7296	1.8	Differences between RFC 5996 and This Document
rfc7296	2	IKE Protocol Details and Variations
rfc7296	2.1	Use of Retransmission Timers
rfc7296	2.2	Use of Sequence Numbers for Message ID
rfc7296	2.3	Window Size for Overlapping Requests
rfc7296	2.4	State Synchronization and Connection Timeouts
rfc7296	2.5	Version Numbers and Forward Compatibility
rfc7296	2.6	IKE SA SPIs and Cookies
rfc7296	2.6.1	Interaction of COOKIE and INVALID_KE_PAYLOAD
rfc7296	2.7	Cryptographic Algorithm Negotiation
rfc7296	2.8	Rekeying
rfc7296	2.8.1	Simultaneous Child SA Rekeying
rfc7296	2.8.2	Simultaneous IKE SA Rekeying
rfc7296	2.8.3	Rekeying the IKE SA versus Reauthentication
rfc7296	2.9	Traffic Selector Negotiation
rfc7296	2.9.1	Traffic Selectors Violating Own Policy
rfc7296	2.9.2	Traffic Selectors in Rekeying
rfc7296	2.10	Nonces
rfc7296	2.11	Address and Port Agility
rfc7296	2.12	Reuse of Diffie-Hellman Exponentials
rfc7296	2.13	Generating Keying Material
rfc7296	2.14	Generating Keying Material for the IKE SA
rfc7296	2.15	Authentication of the IKE SA
rfc7296	2.16	Extensible Authentication Protocol Methods
rfc7296	2.17	Generating Keying Material for Child SAs
rfc7296	2.18	Rekeying IKE SAs Using a CREATE_CHILD_SA Exchange
rfc7296	2.19	Requesting an Internal Address on a Remote Network
rfc7296	2.20	Requesting the Peer's Version
rfc7296	2.21	Error Handling

Document	Section	Title
rfc7296	2.21.1	Error Handling in IKE_SA_INIT
rfc7296	2.21.2	Error Handling in IKE_AUTH
rfc7296	2.21.3	Error Handling after IKE SA is Authenticated
rfc7296	2.21.4	Error Handling Outside IKE SA
rfc7296	2.22	IPComp
rfc7296	2.23	NAT Traversal
rfc7296	2.23.1	Transport Mode NAT Traversal
rfc7296	2.24	Explicit Congestion Notification (ECN)
rfc7296	2.25	Exchange Collisions
rfc7296	2.25.1	Collisions while Rekeying or Closing Child SAs
rfc7296	2.25.2	Collisions while Rekeying or Closing IKE SAs
rfc7427	3	Authentication Payload

This is where the source step placed the purpose of the protocol within the documents: fixed ML-DSA-65, RFC7427 Digital Signature AUTH and the initial IKE SA, the complete SignedOctets, identities and algorithm selection; DH is idealized, and no claim of a DH session key being quantum-resistant is made on the strength of the signature; EAP and rekey are not covered.

The properties against the documents

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

ID	Name	About	What the documents say	Where
A02	peer identity authentication	The initiator is assured that the peer it authenticated is the responder it intended to talk to; the responder is assured likewise of the initiator.	Each peer asserts its identity in IDi/IDr and proves knowledge of the corresponding key with the AUTH payload; both parties MUST verify all signatures and MACs (RFC 7296 Sections 1.2 and 2.15).	s0002, s0055, s0103, s0387
A03	session binding of the AUTH payload	Both parties, about the peer identity, both nonces Ni and Nr, and the contents of the two IKE_SA_INIT messages carried under the signatures.	SignedOctets contains RealMessage1 or RealMessage2, the other side's nonce and prf(SK_p, ID'); RFC 7296 Section 2.15 states it is critical that each side sign the other side's nonce.	s0012, s0045, s0103, s0387, s0389, s0391
A04	no replay of AUTH across sessions	Both parties, about how many peer runs an acceptance can be attributed to: one peer run cannot be counted twice by the other side.	The signed data is unique per session because the nonces are included, so a recorded AUTH cannot be presented again in another session (draft Section 9; RFC 7296 Section 2.15).	s0012, s0045, s0387

ID	Name	About	What the documents say	Where
A08	signature algorithm binding	The accepting party, about the signature and hash algorithm it uses to verify AUTH: it must be one the peer really announced and that the AUTH covers, and it must be the algorithm actually used for verification.	The SIGNATURE_HASH_ALGORITHMS and SUPPORTED_AUTH_METHODS notifications are protected by the AUTH payload; the AlgorithmIdentifier inside the Authentication Data selects the algorithm (RFC 7427 Section 3; draft Section 9).	s0019, s0020, s0044, s0557, s0566
R05	no downgrade to traditional AUTH	The accepting party, about the authentication method of the session: in a deployment that requires PQC signature authentication no session is established with a traditional signature algorithm.	When PQC authentication is mandatory an implementation MUST NOT fall back to a traditional algorithm; deployments requiring protection MUST require PQC authentication and MUST NOT allow fallback (draft Section 9).	s0044
K01	IKE SA and signing key secrecy	Both parties, about the IKE SA key material (SKEYSEED, SK_d, SK_e*, SK_a*, SK_p*) and about their ML-DSA long-term signing keys: neither is derivable by the attacker.	SKEYSEED = prf(Ni | Nr, g^ir) and the seven derived keys (RFC 7296 Section 2.14); the ML-DSA private keys never leave the signer.	s0379, s0382, s0387
C01	forward secrecy of the IKE SA key	A completed IKE SA, about an attacker who later obtains both long-term ML-DSA signing keys: the session key material stays secret.	The session keys are derived from the ephemeral DH exchange and the nonces, not from the signing keys; RFC 7296 Section 5 assumes DH exponents are erased after use.	s0379, s0381, s0382, s0363
C02	no impersonation from own key leak	The accepting party, about the identity of its peer, when its own ML-DSA long-term signing key has leaked.	AUTH is a signature under the signer's own long-term key over data that includes the other side's nonce (RFC 7296 Section 2.15); the draft does not discuss key-compromise impersonation separately.	s0387, s0103, s0044
C08	PQC auth survives broken legacy signatures	The accepting party, about its peer's identity and the algorithm actually used, when traditional signatures can be forged but ML-DSA is unbroken.	Section 9: an attacker that can forge traditional signatures can impersonate a peer that supports traditional authentication and can downgrade by modifying the notifications; requiring PQC authentication removes the attack.	s0044, s0038, s0049
A03.10	AUTH binds identities and nonces	Both parties, about the nonces, identities and first-round message contents the peer accepted (user question Q1).	User question Q1; RFC 7296 Section 2.15 defines InitiatorSignedOctets and ResponderSignedOctets from RealMessage1/2, the peer's nonce and prf(SK_p, ID').	s0387, s0389, s0391, s0103
A08.11	modified capability signals are detected	The accepting party, about the algorithm and hash selection used in the session, when notification payloads are modified (user question Q2).	User question Q2; the notifications are covered by AUTH, and a PQC algorithm requiring the Identity hash must not be used with a peer that did not announce it (draft Sections 3.2.1 and 9).	s0019, s0020, s0044, s0557
C02.12	own key leak does not break peer identity	The accepting party, about its peer's identity, when its own ML-DSA signing key has leaked, with a normal session as positive control (user question Q3).	User question Q3; same construction as C02, with an explicit positive control on the normal run.	s0387, s0103, s0044
X01	identity protection of IDi and IDr	The initiating party, about its own identity value IDi as carried in the IKE_AUTH request, and the responder's IDr, against an attacker on the path.	RFC 7296 Section 1.2: the IKE_AUTH messages are encrypted, so the identities are hidden from eavesdroppers; an on-path attacker that cannot complete IKE_AUTH can nonetheless see the identity of the initiator (RFC 7296 Section 5, P09).	s0084, s0097, s0102

Added from the documents by the analyst: X01.

The security statements of the documents, paragraph by paragraph

The host listed the 53 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: 12. Not taken: 41. 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:595-610	R05, A08, `crqc_traditional`	
P02	main:612-620	A03, A04	
P03	main:622-631	The prover cannot be asked	NIST security categories: strength of a primitive.
P04	main:636-636	No goal stated	Page footer and running header.
P05	main:640-640	No goal stated	Page header of the draft.
P06	main:643-649	The prover cannot be asked	Design strength of the ML-DSA parameter sets.
P07	main:651-653	The prover cannot be asked	Side-channel and non-repudiation properties of RFC 9881: not trace properties.
P08	main:655-662	The prover cannot be asked	Signature-count limits of a primitive; SLH-DSA is outside the fixed ML-DSA-65 scope.
P09	rfc7296:6478-6488	X01, `network`	
P10	rfc7296:6490-6494	Outside the scope	EAP authentication; the task excludes EAP.
P11	rfc7296:6496-6500	Outside the scope	Rekeying with CREATE_CHILD_SA; only the initial IKE SA is in scope.
P12	rfc7296:6502-6514	The prover cannot be asked	Strength of Diffie-Hellman groups; the model idealises DH.
P13	rfc7296:6516-6519	The prover cannot be asked	Same Diffie-Hellman group strength discussion.
P14	rfc7296:6525-6527	The prover cannot be asked	Continuation of the DH strength discussion (page break).
P15	rfc7296:6529-6530	C01	
P16	rfc7296:6532-6538	The prover cannot be asked	Denial of service: availability, not a trace property.
P17	rfc7296:6540-6542	The prover cannot be asked	PRF output length and algorithm strength.
P18	rfc7296:6544-6549	The prover cannot be asked	Quality of the random number generator.
P19	rfc7296:6551-6556	Outside the scope	The task fixes the IKE SA encryption and integrity algorithms; only signature algorithm selection is analysed.
P20	rfc7296:6558-6563	Outside the scope	Pre-shared-key authentication; the task fixes signature authentication.

Paragraphs	Where	Taken up as	Why not
P21	rfc7296:6565-6572	Outside the scope	NAT detection and address hiding: NAT traversal is outside the modelled exchange.
P22	rfc7296:6578-6580	Outside the scope	Continuation of the NAT detection paragraph (page break).
P23	rfc7296:6582-6606	Outside the scope	Non-key-generating EAP methods; EAP is excluded.
P24	rfc7296:6608-6612	Outside the scope	Requirement on EAP deployments.
P25	rfc7296:6614-6619	The prover cannot be asked	Reassembly-buffer exhaustion: availability.
P26 to P27	rfc7296:6621-6633	A02	
P28	rfc7296:6637-6643	Outside the scope	Traffic selector authorization for Child SAs; Child SAs are outside the modelled exchange.
P29 to P35	rfc7296:6645-6696	Outside the scope	Traffic selector authorization.
P36	rfc7427:398-400	The prover cannot be asked	Selecting suitable hash functions: strength of primitives.
P37 to P38	rfc7427:402-417	A08	
P39 to P40	rfc7427:419-429	A08, A03	
P41	rfc7427:431-435	The prover cannot be asked	Downgrade to catastrophically weak ciphers: strength of primitives.
P42	rfc7427:437-439	The prover cannot be asked	MD5 is not in the hash registry.
P43	rfc7427:441-444	The prover cannot be asked	Weaknesses of RSASSA-PKCS1-v1_5 and padding methods: strength of primitives, and the task fixes ML-DSA-65.
P44	rfc7427:446-449	The prover cannot be asked	DSA with SHA-1 limits: strength of primitives.
P45	rfc9593:446-450	No goal stated	Points to the security considerations of RFC 7296, covered paragraph by paragraph there.
P46	rfc9593:452-459	R05, `network`	
P47	rfc9593:461-465	R05, C08, `crqc_traditional`	
P48	rfc9881:573-574	No goal stated	Points to the security considerations of RFC 5280, which the model assumes away.
P49	rfc9881:576-579	The prover cannot be asked	SUF-CMA of ML-DSA: strength of a primitive.
P50	rfc9881:581-590	The prover cannot be asked	PRNG quality for key generation.
P51	rfc9881:592-604	The prover cannot be asked	Side-channel leakage and key recovery from power analysis.
P52	rfc9881:606-614	The prover cannot be asked	Deterministic versus randomized signing: local choice, identical verification, as the model reflects.
P53	rfc9881:616-627	The prover cannot be asked	Non-repudiation needs a dispute model, not a trace property.

The catalogue of properties, entry by entry

The properties were drafted from a catalogue of security and privacy properties. The draft left out 33 of its entries, and the analyst went through each of them: examined here as a property, or left out with the reason. Examined: 9. Left out: 24. That the documents do not speak of a property is not a reason the analyst may give.

Entry	Property of the catalogue	Examined as	Why not
K02	Payload and end-to-end confidentiality	X01	
K03	Strong secrecy / indistinguishability of secrets	K01	
K04	Key purpose, direction and session separation	The prover cannot be asked	Symbolic keys are distinct terms by construction, so a query cannot tell key reuse from separation.
D01	Message integrity and data origin authenticity	A03, A08	
A01	Peer aliveness	A02	
A05	Implicit key authentication	K01, A03	
A06	Explicit key confirmation	Not applicable	The initial exchange has no explicit key-confirmation message.
A07	Identity–credential–key binding	A02	
R01	Authentication / challenge freshness	A04	
R02	Message-level replay resistance	A04	
R03	At-most-once business operation	Not applicable	No business operation or consumable token takes part.
R04	Message acceptance order / window security	Outside the scope	Message-ID windows (RFC 7296 Sections 2.2 and 2.3) are outside the modelled exchange.
C03	Known-key security / compromise isolation	C01	
C04	Ephemeral secrets / partial-state compromise resilience	Outside the scope	Ephemeral or partial memory leakage is not modelled.
C05	Post-compromise security (PCS)	Outside the scope	Rekeying and post-compromise recovery are excluded by the task.
C06	Enhanced authentication after compromise of enrolment credentials	Not applicable	No registration record or PAKE-style authentication is involved.
C07	Resistance to offline dictionary attacks	Not applicable	No low-entropy shared secret takes part in signature authentication.
Z01	Correct enforcement of authorization	Not applicable	No protected operation or resource access is authorised.
Z02	Delegated authority does not expand	Not applicable	No delegation between parties.
Z03	Authorization bound to identity / session / operation	Not applicable	No authorisation token or service session is bound.
Z04	Revocation and expiry take effect	Not applicable	No revocation or expiry of a permission.
Z05	Cross-tenant / cross-service security isolation	Not applicable	No multi-tenant or cross-service isolation.
S01	Encryption nonce uniqueness	The prover cannot be asked	The SK{} envelope is idealised; AEAD nonce reuse is invisible to a symbolic prover.
S02	Retransmission / key reinstallation security	Outside the scope	Retransmission and key reinstallation belong to CREATE_CHILD_SA and INFORMATIONAL, which the task excludes.

Entry	Property of the catalogue	Examined as	Why not
S03	Rollback resistance and state monotonicity	Outside the scope	Rollback of security state concerns rekeying and stored SAs.
S04	Cross-phase security context consistency	Outside the scope	Cross-phase context confusion needs rekeying, roaming or resumption, all excluded.
S05	Termination and invalidation of old state	Outside the scope	Teardown and deauthentication are excluded.
P01	Identity anonymity	Not applicable	Peer identities are known certificate names; only eavesdropper identity protection is claimed, and that is the added goal.
P02	Session unlinkability	Not applicable	No ephemeral identity or pseudonym makes sessions unlinkable.
P03	Hiding of user / credential existence	Not applicable	No user or credential directory is queried.
V01	Conditional progress / error recovery	The prover cannot be asked	Progress and error recovery are liveness properties.
V02	No unexpected deadlock	The prover cannot be asked	Deadlock freedom is a liveness property.
V03	Limited resource abuse / amplification resistance	Outside the scope	Resource abuse and the anti-clogging cookie are outside the modelled authentication.

The parties and the protected values, one by one

The host listed the 27 parties and protected values that the drafted properties name, and the analyst said of each which attacker controls the party or holds the value, or why no such attacker was set. With an attacker: 4. Without: 23. The list follows from the properties; what was made of each entry is the analyst's judgement.

ID	Party or value	Kind	Attacker	Why none
S01	Initiator	party	`own_ltk`	
S02	Responder	party	`own_ltk`	
S03	Path attacker	party	Nothing gained	It is the attacker on the network itself; no property is asked about it.
S04	Acceptor	party	`own_ltk`	
S05	Uncompromised peer	party	Nothing gained	This is the peer whose key stays secret by the assumption of the key-compromise target; if the attacker has that key it simply is that peer, and a session it completes is a genuine run.
S06	Attacker	party	Nothing gained	The attacker itself.
S07	Peer identity	value	Nothing gained	Identities are certificate names any party can look up; knowing them leaves authentication askable and unaffected.
S08	Identity	value	Nothing gained	Identities are certificate names any party can look up; the confidentiality question about them is the added identity-protection goal, asked against the network attacker.
S09	Ni	value	Nothing gained	Identities are certificate names any party can look up; the confidentiality question about them is the added identity-protection goal, asked against the network attacker.

ID	Party or value	Kind	Attacker	Why none
S10	Nr	value	Nothing gained	Identities are certificate names any party can look up; the confidentiality question about them is the added identity-protection goal, asked against the network attacker.
S11	IKE_SA_INIT message	value	Nothing gained	Identities are certificate names any party can look up; the confidentiality question about them is the added identity-protection goal, asked against the network attacker.
S12	Signature	value	Nothing gained	Identities are certificate names any party can look up; the confidentiality question about them is the added identity-protection goal, asked against the network attacker.
S13	SIGNATURE_HASH_ALGORITHMS notification	value	Nothing gained	Notifications, the AlgorithmIdentifier and the selection they imply are values the peers announce; holding them gives nothing beyond the network, which already sees the IKE_SA_INIT notifications.
S14	SUPPORTED_AUTH_METHODS notification	value	Nothing gained	Notifications, the AlgorithmIdentifier and the selection they imply are values the peers announce; holding them gives nothing beyond the network, which already sees the IKE_SA_INIT notifications.
S15	AlgorithmIdentifier	value	Nothing gained	Notifications, the AlgorithmIdentifier and the selection they imply are values the peers announce; holding them gives nothing beyond the network, which already sees the IKE_SA_INIT notifications.
S16	Authentication method selection	value	Nothing gained	Notifications, the AlgorithmIdentifier and the selection they imply are values the peers announce; holding them gives nothing beyond the network, which already sees the IKE_SA_INIT notifications.
S17	Signature algorithm selection	value	Nothing gained	Notifications, the AlgorithmIdentifier and the selection they imply are values the peers announce; holding them gives nothing beyond the network, which already sees the IKE_SA_INIT notifications.
S18	SKEYSEED	value	Everything lost by definition	With SKEYSEED known the confidentiality of that IKE SA is lost by definition and no target asks a session property under that assumption.
S19	SK_d	value	Everything lost by definition	With SKEYSEED known the confidentiality of that IKE SA is lost by definition and no target asks a session property under that assumption.
S20	SK_ei/SK_er	value	Everything lost by definition	With SKEYSEED known the confidentiality of that IKE SA is lost by definition and no target asks a session property under that assumption.
S21	SK_ai/SK_ar	value	Everything lost by definition	With SKEYSEED known the confidentiality of that IKE SA is lost by definition and no target asks a session property under that assumption.
S22	SK_pi/SK_pr	value	Everything lost by definition	With SKEYSEED known the confidentiality of that IKE SA is lost by definition and no target asks a session property under that assumption.
S23	ML-DSA signature private key	value	`own_ltk`, `ltk_after_session`	
S24	IKE SA session keys	value	Everything lost by definition	Same as SKEYSEED: these keys are what the confidentiality targets protect.

ID	Party or value	Kind	Attacker	Why none
S25	Peer identity authentication	value	Nothing gained	These are the properties the targets ask, not values a party holds separately.
S26	Identity authentication	value	Nothing gained	These are the properties the targets ask, not values a party holds separately.
S27	Algorithm binding	value	Nothing gained	These are the properties the targets ask, not values a party holds separately.

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		
`crqc_traditional`	every traditional (pre-quantum) signature private key of both peers, so it can produce AUTH payloads with a traditional signature algorithm at will; it has no ML-DSA private key.	The draft assumes a CRQC-capable attacker that can forge traditional signatures (Section 9) and that PQC signature authentication may be optional per deployment policy.	s0044, s0045, s0049
`own_ltk`	the ML-DSA long-term signing private key of the party that is accepting the peer's identity, obtained before the session starts; the peer's ML-DSA key stays secret.	Key-compromise impersonation: RFC 7296 Section 2.15 makes AUTH a signature under the signer's own key over data that includes the other side's nonce, so the question is whether a leaked own key lets an attacker pose as the other, uncompromised peer.	s0387, s0389, s0391, s0103
`ltk_after_session`	both peers' ML-DSA long-term signing private keys, obtained after the IKE SA has been established and its key material erased.	The IKE SA keys come from the ephemeral Diffie-Hellman exchange and the nonces (RFC 7296 Section 2.14), and exponents are assumed erased after use (Section 5); the draft does not claim quantum resistance for the DH session key.	s0379, s0381, s0382, s0363

ID	`network`	`crqc_traditional`	`own_ltk`	`ltk_after_session`
A02	Holds	Holds		
A03	Holds	Holds		
A04	Holds	Holds		
A08	Holds	Holds		
R05	Holds	Holds		
K01	Holds			
C01				Holds
C02			Holds	
C08		Holds		
A03.10	Holds	Holds		
A08.11	Holds	Holds		

ID	network	crqc_traditional	own_ltk	ltk_after_session
C02.12			Holds	
X01	Fails			

Source and targets

Source: 573 units, 187263 bytes, frozen in source-delivery-01.json. Targets: 9 approved (A02, A03, A04, A08, R05, K01, C01, C02, C08); the questions asked: 3. The list of targets was approved as drafted, beforehand, with --approve-targets.

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

- approve_package: approved the source package protocol-c2cbd1d4b40623958b6c0349162f

This step as it reports itself: 1 requests (1 failed), 0.000 USD, 81276 tokens in and 4413 out, 26 s.

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
33	A02, A03, A03.10, A04, A08, A08.11, C01, C02, C02.12, C08, K01, R05, X01	plain_primitives, many_sessions	All answers rest on ML-DSA-65 as an unforgeable signature, a PRF, a MAC and an idealised encryption, with Diffie-Hellman reduced to its single law g^(xy)=g^(yx) (the model has no DH algebra beyond that, so no DH attack is represented except the degenerate share of g seen in the identity-protection attack). The roles are replicated, so the authentication, secrecy and forward-secrecy answers are about any number of sessions and any number of peers, and only the two fixed session-key values are tracked. No counter or sequence number is modelled, so no target rests on a value that changes.

The analyst's closing note

The model runs (normal run verified) and all thirteen targets have recorded answers, each with a control where a variant exists. Established by proof on model.pv: mutual ML-DSA-65 authentication (A02), the binding of both identities, both nonces and both IKE_SA_INIT messages (A03, A03.10), injective agreement so a recorded AUTH is not counted twice (A04), single-algorithm behaviour in a PQC-required deployment plus the algorithm/hash signal coming from the AUTH-covered messages (A08, A08.11), no fallback to traditional AUTH (R05), and the same guarantees when every traditional signature key is in the attacker's hands (C08). Session-key and signing-key secrecy during the run (K01), forward secrecy of the established IKE SA key against an attacker who later obtains both signing keys (C01), and resistance to key-compromise impersonation by a party whose own signing key has leaked (C02, C02.12) also hold. Reproduced and marked as stated in the documents: the initiator's identity is readable by an on-path attacker that answers IKE_SA_INIT itself (X01, RFC 7296 §1.2/§5), and a deployment that allows fallback is impersonated outright by a traditional-signature forger (the control of R05, draft §9). No finding: nothing the documents do not already say was established. Blocking a stronger result: Diffie-Hellman is idealised to g^xy = g^yx, so the forward-secrecy and session-key answers hold only under that idealisation and the degenerate-share mechanism behind the identity leak is the one DH artefact the model does show; certificate validation, message IDs, fragmentation, Child SA traffic selectors, EAP and rekeying are outside the model. Tamarin+ Agent, which models DH as it is, would settle the forward-secrecy question as the specification states it.