Security analysis of IKEv2 PPK in IKE_INTERMEDIATE

This report in another language: 中文.

rfc9867 · ProVerif+ Agent report

Documents	Mixing Preshared Keys in the IKE_INTERMEDIATE and CREATE_CHILD_SA Exchanges of the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-Quantum Security (protocol definition); Internet Key Exchange Protocol Version 2 (IKEv2) (supporting); Intermediate Exchange in the Internet Key Exchange Protocol Version 2 (IKEv2) (supporting)
Properties examined	11: 9 hold, 2 fail (3 attacks)
Findings	1; known results reproduced in Appendix B: 1

Contents

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

1 Protocol and scope

IKEv2 with a post-quantum preshared key negotiated in an IKE_INTERMEDIATE exchange is meant to make the initial IKE SA itself, and not only later exchanges, resistant to a quantum attacker. The initiator and the responder first run IKE_SA_INIT, which negotiates the algorithms, exchanges nonces and derives a first generation of IKE SA keys from Diffie-Hellman. They then run one IKE_INTERMEDIATE exchange in which the initiator offers a PPK identity and a PPK Confirmation and the responder answers with the identity it selected, after which both recompute all IKE SA keys from SKEYSEED' = prf+(PPK, SK_d). The IKE_AUTH exchange that follows authenticates the peers with their long-term credentials over a signed block that includes both nonces, the identity and the content of the intermediate exchange, and its payloads are protected by the PPK-derived keys.

When the exchange ends, each side is sure that the peer really ran it for these nonces, SPIs and intermediate transcript, and that no one without the PPK can read or forge the IKE_AUTH payloads, even an attacker who can compute the Diffie-Hellman shared secret. The initiator is also assured that the responder either confirmed one of the PPK identities it offered or, where the initiator requires PPK protection, that the establishment fails instead.

The parties:

- **Initiator**: proposes the PPK, requires PPK protection. Holds: own signing key; PPK with its PPK_ID.
- **Responder**: selects the PPK identity. Holds: own signing key; the PPK for that PPK_ID; policy PPK mandatory or optional.

Figure 1 shows the exchange as the documents specify it.

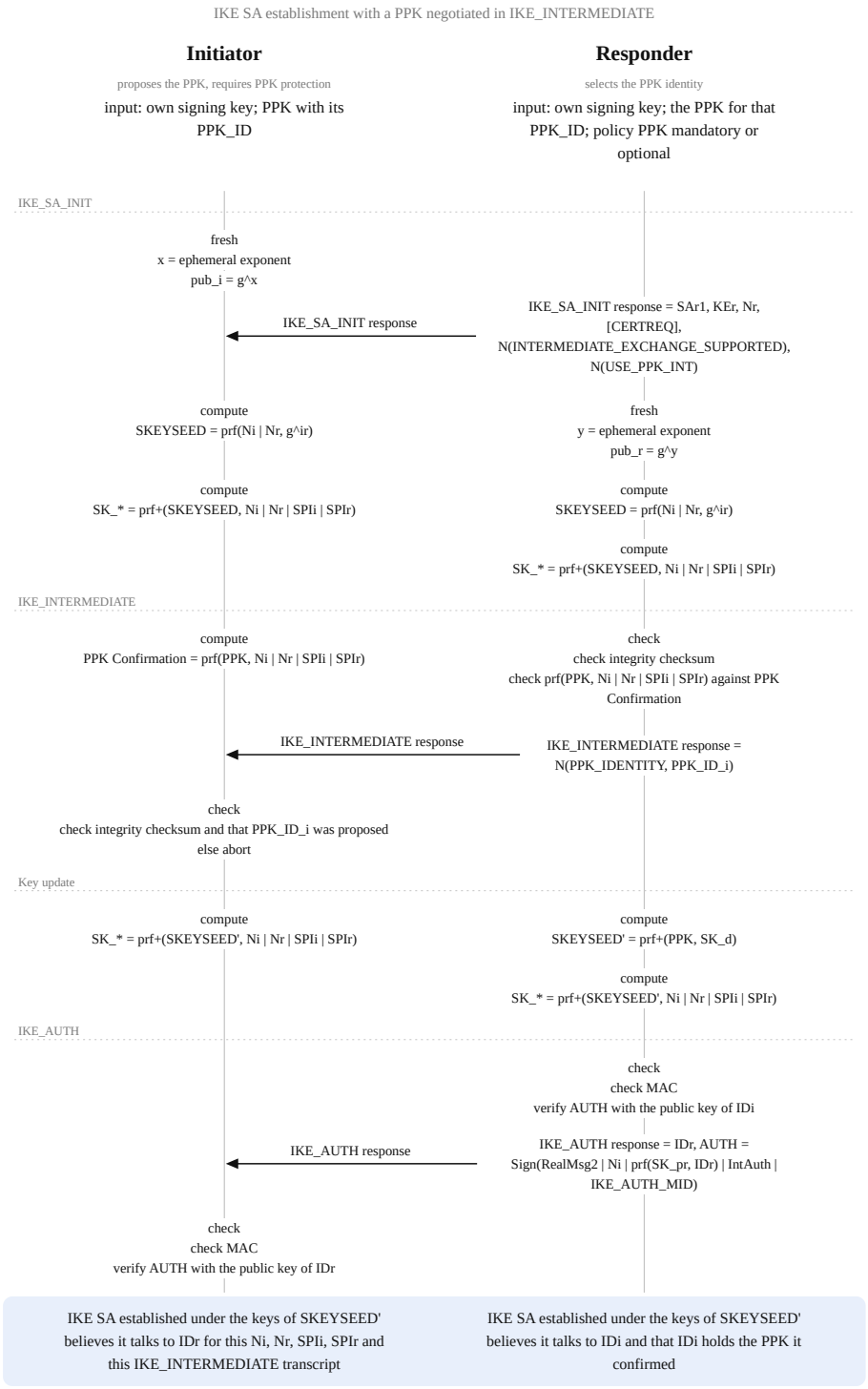


Figure 1. The protocol as specified.

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

The analysis has 1 finding: what follows is not stated in the documents. For 1 of them it is unclear whether the documents state it. 1 result that the documents already state is not counted and is in Appendix B.

Within the scope analysed, RFC 9867 keeps its promise: with the PPK secret, both the IKE_AUTH authentication and the IKE SA keys recomputed from SKEYSEED' survive an attacker that has broken Diffie-Hellman and the signature algorithms, and a network attacker can neither forge nor replay an IKE_AUTH. The policy rules of section 3.1 also fail closed: a mandatory responder never completes an IKE SA without a confirmed PPK, and an initiator that requires PPK never accepts a PPK-less session. One thing the documents do not say is left exposed: the PPK_ID and the PPK Confirmation travel in IKE_INTERMEDIATE, which only the IKE_SA_INIT keys protect, so a post-quantum attacker, or a network attacker that answers IKE_SA_INIT itself, reads them. The low-entropy-PPK dictionary consequence is a known result, restated here with its exact conditions. All results come from a symbolic model that idealises the primitives, gives the post-quantum attacker g^ir as a public function of the transcript, and does not model Key Exchange payload validation, Message ID counters or the identity-to-PPK policy.

The attack figures use the columns of Figure 1. Red is what the attacker does. Yellow marks a message that a party sends although the specified exchange does not have it at that point, and the outcome that the attack breaks.

2.1 The PPK identity and confirmation travel under keys a quantum attacker can break

Kind	Weakness of the protocol as specified
New	Unclear: the documents state it in part, or could not be searched (rfc9242:463, rfc8784:567)
Value	medium
Properties	X01 (section 3)
Source text	s0013, s0016, s0031, s0561
Shown by	ProVerif attack queries `x01_crqc`, `x01_net`; Figures 2 and 3

What happens. The initiator sends its PPK_ID and the PPK Confirmation inside the IKE_INTERMEDIATE request, protected only by the first generation of IKE SA keys from IKE_SA_INIT. An attacker that can compute that shared secret, which is what a quantum computer gives, reads both values off the wire. A network attacker reaches the same values more cheaply by answering the IKE_SA_INIT request itself, because then it is the Diffie-Hellman peer of that exchange.

What the attacker needs. A quantum computer able to compute the IKE_SA_INIT shared secret, which is what the whole extension is designed to survive; or, for the second route, only the ability to answer IKE_SA_INIT with a group element of its own choosing.

What the attacker gains. The attacker learns which PPK group a session uses and obtains prf(PPK, Ni|Nr|SPIi|SPIr) for that session. That value lets it test guesses of the PPK offline, so a PPK that people chose rather than generated becomes recoverable and the quantum protection of the session is lost. The PPK itself and the keys mixed from it are not exposed.

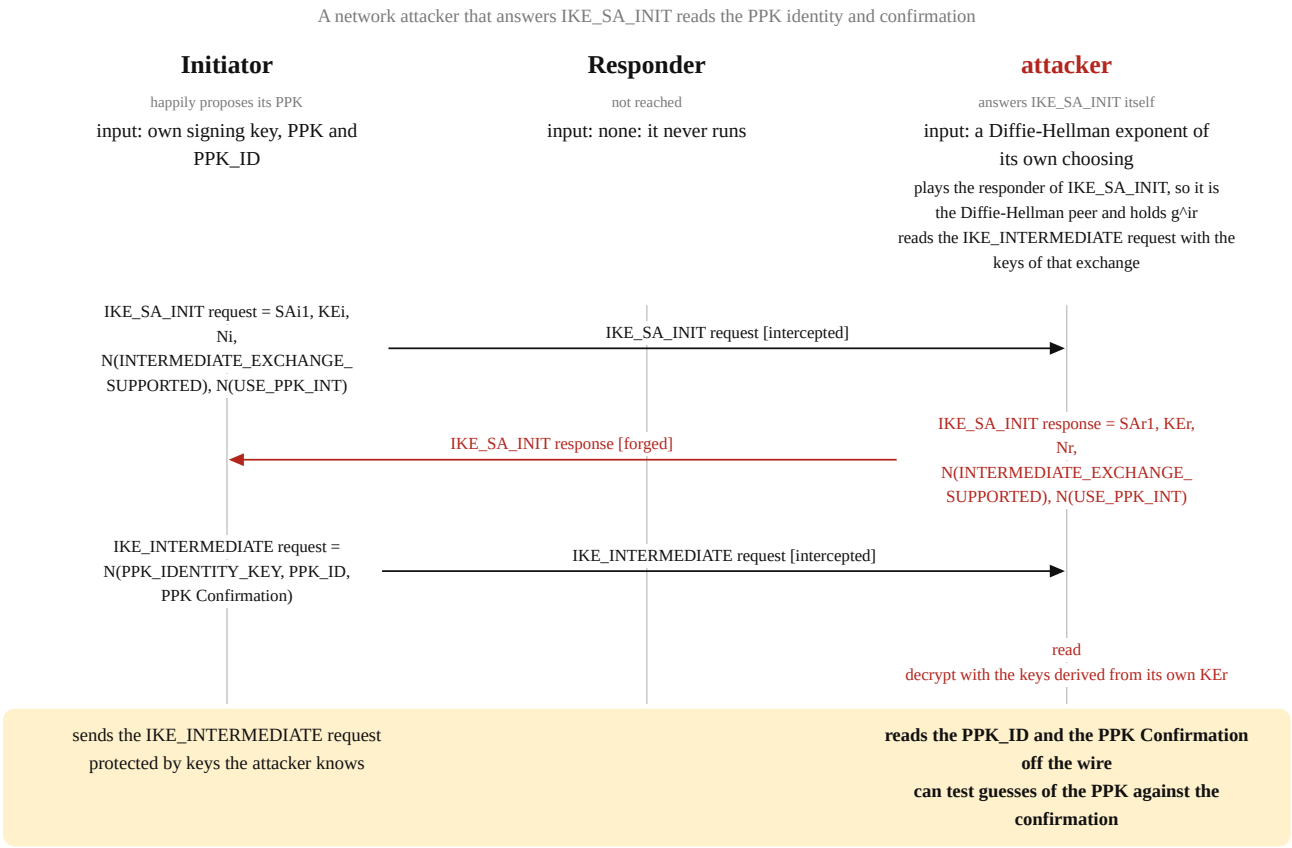


Figure 2. A network attacker that answers IKE_SA_INIT reads the PPK identity and confirmation. Property X01; attack query x01_net.

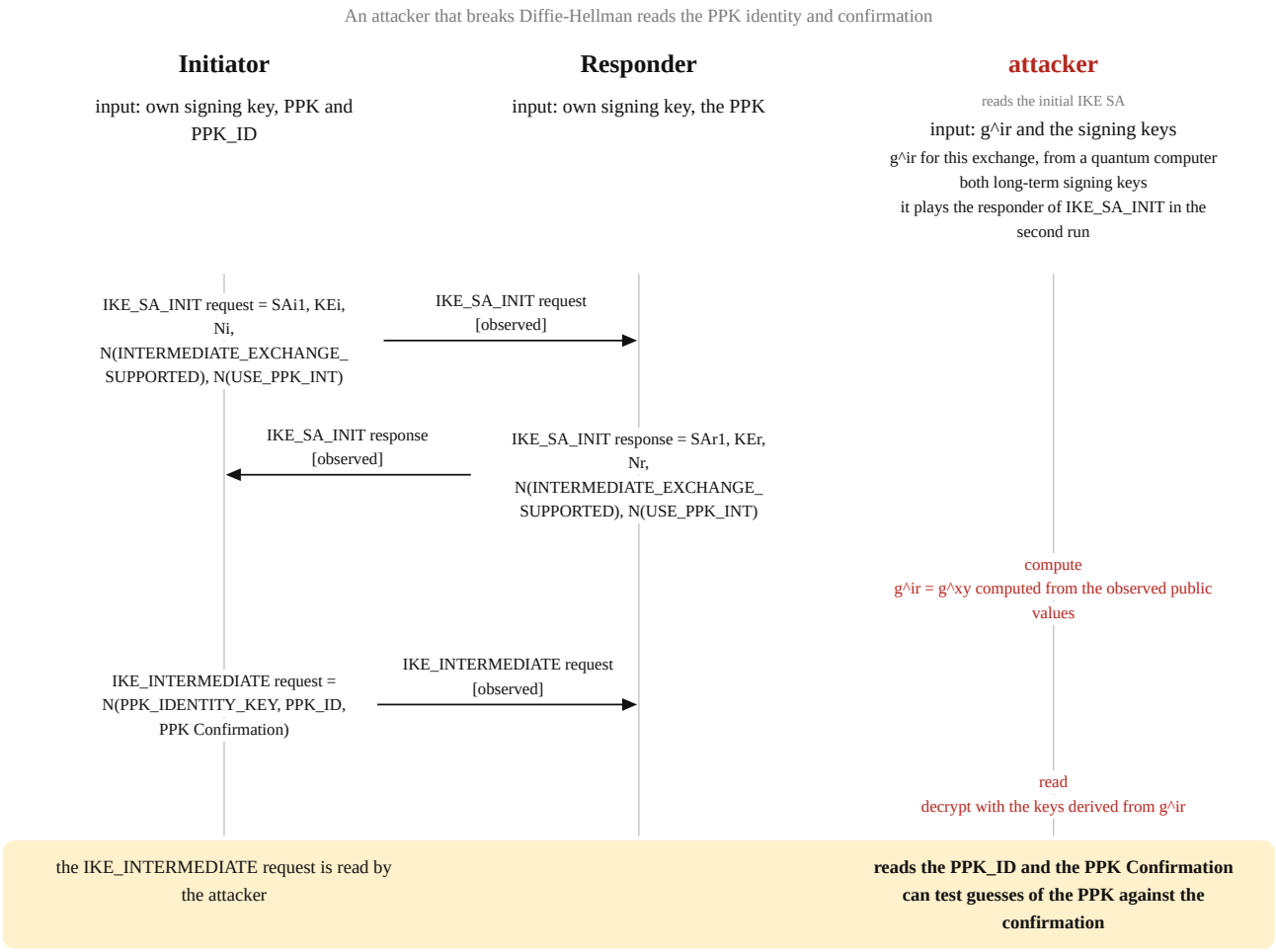


Figure 3. An attacker that breaks Diffie-Hellman reads the PPK identity and confirmation. Property X01; attack queries x01_crqc, x01_net.

Proposed change. None. No change of the messages removes the exposure: until the PPK is mixed in, the intermediate exchange can only be protected by keys a quantum attacker can break, so any PPK-dependent value sent there is readable by design. The remedy is a deployment requirement (a random PPK of at least 256 bits, as RFC 8784 states) plus keeping the confirmation short, not a change of the protocol's messages that a model could check.

3 Properties examined

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

ID	Property	Compromise assumed	Result	Value	In the documents
A03	**Matching run for IKE_AUTH acceptance.** Whenever an initiator or responder accepts the peer's authentication in IKE_AUTH, there is a matching real run of the peer in which the roles, both intended identities, and PPK_ID, Ni, Nr, SPIi, SPIr, IntAuth agree.	no compromise	Holds	high	yes

ID	Property	Compromise assumed	Result	Value	In the documents
A04	**Distinct peer runs per acceptance.** Every successful IKE_AUTH acceptance corresponds to a distinct real run of the peer; the same IKE_INTERMEDIATE/IKE_AUTH transcript cannot be accepted repeatedly as a new successful session.	no compromise	Holds	medium	unclear
A08	**PPK fields bound to this session.** The accepted PPK_ID, PPK Confirmation, IntAuth and IKE_AUTH_MID correspond to this IKE_SA_INIT's Ni, Nr, SPIi, SPIr and this IKE_INTERMEDIATE transcript, and agree with the identities authenticated in IKE_AUTH.	no compromise	Holds	medium	no
K01	**Secrecy of PPK-derived IKE SA keys.** The attacker cannot derive the PPK value, nor the IKE SA keys SK_d, SK_ei, SK_er, SK_ai, SK_ar, SK_pi, SK_pr derived from the PPK and SK_d.	no compromise	Holds	high	yes
R05	**No PPK-less session under mandatory policy.** Under a mandatory PPK policy, if the responder has no matching PPK or the value does not match, IKE SA establishment must fail; under an optional policy, if the initiator requires PPK protection, a session that does not use a PPK must not be accepted by the initiator as a successful PPK session.	no compromise	Holds	medium	yes
C04	**IKE_AUTH protected when Diffie-Hellman breaks.** When the attacker obtains the IKE_SA_INIT Diffie-Hellman shared secret g^ir but not the PPK, it cannot derive the IKE_AUTH message contents or the new IKE SA keys derived from SKEYSEED'.	the IKE_SA_INIT Diffie-Hellman shared secret g^ir (after IKE_SA_INIT completes, before or during IKE_INTERMEDIATE and IKE_AUTH)	Holds	medium	yes
A08.7	**PPK binding, as asked in Q1.** The accepted PPK_ID, PPK Confirmation, IntAuth and IKE_AUTH_MID correspond to this IKE_SA_INIT's Ni, Nr, SPIi, SPIr and this IKE_INTERMEDIATE transcript, and agree with the identities authenticated in IKE_AUTH.	no compromise	Holds	medium	no
U01	**DH break, as asked in Q2.** When the attacker obtains g^ir but not the PPK, it cannot derive the IKE_AUTH contents or the new IKE SA keys derived from SKEYSEED'.	the IKE_SA_INIT Diffie-Hellman shared secret g^ir (after IKE_SA_INIT completes, before or during IKE_INTERMEDIATE and IKE_AUTH)	Holds	medium	yes
R05.9	**PPK policy, as asked in Q3.** Under a mandatory PPK policy, if the responder has no matching PPK or the value does not match, IKE SA establishment must fail; under an optional policy, if the initiator requires PPK protection, a session that does not use a PPK must not be accepted by the initiator as a successful PPK session.	no compromise	Holds	medium	yes

ID	Property	Compromise assumed	Result	Value	In the documents
X01	**PPK fields hidden under broken DH.** An attacker that can decrypt the initial IKE SA (knows g^ir) does not learn the PPK_ID the initiator proposed or the PPK Confirmation value it sent in IKE_INTERMEDIATE.	the IKE_SA_INIT shared secret g^ir (so it can read every encrypted message of the initial IKE SA) and the long-term signature keys of both peers, from the start of the session; symmetric primitives stay out of reach	Fails (2 attacks)	medium	unclear
X02	**PPK not recoverable by offline guessing.** An attacker that obtains a PPK Confirmation value (by impersonating the responder in IKE_SA_INIT/IKE_INTERMEDIATE) cannot test guesses of a person-chosen PPK against it offline and recover the PPK.	several candidate PPK values and the PPK Confirmation value of one IKE_INTERMEDIATE exchange, obtained by impersonating the responder in IKE_SA_INIT and reading the IKE_INTERMEDIATE request; it can test a guess offline	Attack holds (stated in the documents)	low	yes

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

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

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: PPK identity binding to the intermediate transcript and identity	Yes: the PPK_ID and the PPK Confirmation a responder accepts belong to that session's Ni, Nr, SPIi and SPIr, and the IntAuth binding means the peer that completes IKE_AUTH is the one that sent the intermediate request; the association with the peer identity is checked only for a single identity, so the identity-to-PPK policy of section 3.1 stays open.	A08, A08.7, A03
Q2: confidentiality of IKE_AUTH contents and new keys when DH has leaked but the PPK has not	Yes: with the IKE_SA_INIT shared secret known and the PPK secret, neither the IKE_AUTH content nor the keys recomputed from SKEYSEED' can be derived, which is the point of mixing the PPK before IKE_AUTH.	C04, U01, K01
Q3: fallback policy and authentication failure are not treated as a successful PPK session	Yes, for the policies the document states: a mandatory responder never completes an IKE SA without a confirmed PPK, and an initiator that requires PPK never accepts a PPK-less session. The controls show that a peer which tolerates a missing PPK does complete such a session.	R05, R05.9

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:

- Symbolic model: no computational security, and no quantum attack on the primitives beyond what is modelled by publishing g^ir and the signing keys.
- Key Exchange payload validation is not modelled, so the attacker can be the Diffie-Hellman peer of an arbitrary exchange by choosing a group element; a real implementation rejects invalid public values.

- Message IDs are a fixed constant: the sequence rule of RFC 9242 section 3.2, the acceptance window of RFC 7296 section 2.3 and retransmissions are not modelled.
- One initiator identity and at most two PPK values: the identity-to-PPK policy of main s0029 (SHOULD abort, MAY continue) is not covered.
- Only one IKE_INTERMEDIATE exchange; several exchanges, CREATE_CHILD_SA, rekeying, NAT traversal, fragmentation and EAP are outside the scope.
- The PPK Confirmation is the full prf output rather than its first eight octets, and IntAuth is the inner payloads rather than the byte layout of RFC 9242 section 3.3.2.
- Availability properties, such as the DoS advice of RFC 8784 section 6 and RFC 9242 section 5, cannot be asked of the prover.

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

Appendix A: Reading of each property

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

A03: Matching run for IKE_AUTH acceptance

Whenever an initiator or responder accepts the peer's authentication in IKE_AUTH, there is a matching real run of the peer in which the roles, both intended identities, and PPK_ID, Ni, Nr, SPIi, SPIr, IntAuth agree.

IKE_AUTH authentication is sound in both directions: an AUTH payload is only accepted by a peer that ran the exchange for the same nonces, SPIs and intermediate transcript. The guarantee survives an attacker that knows the IKE_SA_INIT shared secret and both long-term signature keys, because the part of the signed data that carries prf(SK_pi, IDi) comes from the PPK-derived SK_pi, which such an attacker does not have. Against a network attacker the same correspondence holds.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value high: The main document states that the IKE_AUTH exchange is quantum secure even against an active attacker because the PPK is mixed into the keys and used in authentication; the analysis reproduces that mechanically for an attacker that has broken Diffie-Hellman and signatures.
- Stated in the documents: yes. Where the documents say it: s0039, s0377, s0568, s0574
- Rests on: Diffie-Hellman is modelled with the single law g^xy = g^yx, and signature unforgeability is idealised: a signature can only be made with the signing key.; The post-quantum attacker is modelled by making g^ir a public function of the transcript and publishing both long-term signing keys; no real cryptanalysis is modelled.; AUTH is modelled as a signature over RealMessage1 or RealMessage2, the peer nonce, prf(SK_p, ID) and IntAuth; the byte layout of RFC 7296 section 2.15 and RFC 9242 section 3.3.2 is abstracted.; Validation of the received Key Exchange payload is not modelled, so any group element is accepted as the peer's public value.; One identity per role: the association between an identity and the PPKs allowed for it (main s0029) is not modelled.; The control query is falsified only for the no-PPK path of the post-quantum attacker, which shows that the PPK is what carries the result.
- Open: Whether the same holds when the PPK is shared with several peers and the responder may continue with a PPK not associated with the authenticated identity (main s0029).
- Next step: Model two initiator identities with two PPKs and the responder policy of main s0029, to see whether a post-quantum attacker holding one PPK can complete IKE_AUTH as another identity.

A04: Distinct peer runs per acceptance

Every successful IKE_AUTH acceptance corresponds to a distinct real run of the peer; the same IKE_INTERMEDIATE/IKE_AUTH transcript cannot be accepted repeatedly as a new successful session.

No recorded IKE_INTERMEDIATE/IKE_AUTH transcript is accepted twice: every acceptance of a PPK-protected IKE SA corresponds to a distinct run of the peer. Binding to both nonces, the SPIs, IntAuth and the IKE_AUTH Message ID is what prevents replay, and it holds against the network attacker and against an attacker that has broken Diffie-Hellman and the signatures.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: The documents require Message IDs to be verified in IKE_INTERMEDIATE (rfc9242 s0555) and bind IKE_AUTH_MID into IntAuth, but do not state injectivity of authentication in so many words; the

result is what those rules achieve.

- Stated in the documents: unclear. Source text it rests on: s0377, s0555, s0569, s0573
- Rests on: Message IDs are modelled as a fixed constant rather than as counters with the ordering rule of RFC 9242 section 3.2, so the sequence check itself is not modelled.; The same idealisations of the primitives as for A03.; Injectivity is shown for the responder's acceptance of the initiator's AUTH; the other direction is shown by the non-injective correspondence.; The control is the same property without the PPK protection, which is falsified.
- Open: The acceptance window of Message IDs and the retransmission behaviour (RFC 7296 sections 2.2 and 2.3) are not modelled.
- Next step: Model the Message ID counters of both peers and the rule of RFC 9242 section 3.2 to check the check itself.

A08: PPK fields bound to this session

The accepted PPK_ID, PPK Confirmation, IntAuth and IKE_AUTH_MID correspond to Ni, Nr, SPIi and SPIr of this IKE_SA_INIT and to the transcript of this IKE_INTERMEDIATE, and are consistent with the identity authenticated in IKE_AUTH.

The PPK negotiation a responder accepts is tied to this IKE SA: the PPK Confirmation is checked against Ni, Nr, SPIi and SPIr, and IntAuth binds the intermediate messages into the AUTH payload that identifies the peer. The identity side of the claim is only checked for a single initiator identity, so the association between an identity and its authorised PPKs is not covered.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value medium: Searched the documents for the association between PPK, session parameters and identity: main s0013-s0014 define the confirmation over the session values, rfc9242 s0568-s0574 bind the intermediate transcript, and main s0029 states that the responder may notice a PPK-identity mismatch only later. No passage states the combined binding as a property, and none states that it fails.
- Stated in the documents: no. Source text it rests on: s0013, s0014, s0018, s0029, s0568, s0573
- Rests on: One PPK and one initiator identity, so the policy question of main s0029 is not modelled.; The PPK Confirmation is modelled as the full prf output, not truncated to eight octets.; The A and P chunks of RFC 9242 section 3.3.2 are abstracted to the inner payloads of the messages.
- Open: The case of several PPK identities proposed by the initiator (main s0016) and the identity-to-PPK policy of main s0029.
- Next step: Model two PPK identities and two initiator identities to check the policy of main s0029 under a post-quantum attacker.

A08.7: PPK binding, as asked in Q1

The accepted PPK_ID, PPK Confirmation, IntAuth and IKE_AUTH_MID correspond to Ni, Nr, SPIi and SPIr of this IKE_SA_INIT and to the transcript of this IKE_INTERMEDIATE, and are consistent with the identity authenticated in IKE_AUTH.

For question Q1 the same result holds: the accepted PPK_ID and PPK Confirmation belong to this IKE SA's nonces and SPIs, and the peer that completes IKE_AUTH is the one that sent the intermediate request. The identity side of the binding is only as strong as the single modelled identity allows.

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

- Value medium: No passage states this combined binding as a property, nor that it fails; the pieces are in main s0013-s0014 and rfc9242 s0568-s0574.
- Stated in the documents: no. Source text it rests on: s0013, s0014, s0018, s0029, s0568, s0573
- Rests on: The same idealisations and the single identity as for A08.; The PPK Confirmation is not truncated in the model.
- Open: Several proposed PPK identities and the identity-to-PPK policy.
- Next step: Add a second identity and a second PPK to test the policy of main s0029.

K01: Secrecy of PPK-derived IKE SA keys

The attacker cannot derive the PPK value, nor the IKE SA keys SK_d, SK_ei, SK_er, SK_ai, SK_ar, SK_pi and SK_pr derived from the PPK and SK_d.

The PPK itself stays unknown to the attacker in both worlds, and the IKE SA keys recomputed from SKEYSEED' = prf+(PPK, SK_d) cannot be derived from the whole transcript, even by an attacker that knows g^ir and both signing keys. For the network attacker the same holds, and there the protection is doubly redundant because it also does not know g^ir.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value high: RFC 8784 states that SK_d is a function of the PPK and cannot be recovered by a quantum attacker when the PPK has enough entropy; the analysis reproduces that for the RFC 9867 key schedule.
- Stated in the documents: yes. Where the documents say it: s0033, s0035, s0036
- Rests on: prf and prf+ are idealised as one-way functions with independent outputs per label; the real construction takes the keys as slices of one prf+ stream.; The keys are shown secret for a session in which the PPK was confirmed; the PPK reveal at the end of the run makes them recoverable, which the control shows.; SK_d is used only to derive Child SA keys, outside this scope, and is not part of the explicit key test.
- Open: Whether the same holds for the keys of an IKE SA that later performs a CREATE_CHILD_SA rekey, which is outside this scope.
- Next step: Extend the model to CREATE_CHILD_SA to check that the PPK-derived SK_d protects the rekeyed keys.

R05: No PPK-less session under mandatory policy

Under a mandatory PPK policy, IKE SA establishment must fail if the responder has no matching PPK or the value does not match; under an optional policy, if the initiator requires PPK protection, a session without PPK must not be accepted by the initiator as a successful PPK session.

The fail-closed rules work: with a policy that mandates the PPK the responder never completes an IKE SA when the PPK is not confirmed, and an initiator that requires PPK protection never treats a PPK-less session as a PPK-protected one. Both results are about the branches the specification prescribes, and the controls show that the peer which tolerates a missing PPK does complete such a session, so the difference is the policy check and not the model.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: RFC 8784 states that a downgrade by an attacker with a quantum computer is detected when PPKs are mandatory for at least one peer, and that with an optional policy a misconfiguration creates the IKE SA without quantum protection; both are reproduced.
- Stated in the documents: yes. Where the documents say it: s0021, s0023, s0025
- Rests on: The policy branches are modelled from main s0021, s0023 and s0025 with the failure branch of s0021; the retransmission and DoS handling of RFC 8784 section 6 is not modelled.; The mismatch case uses a second

PPK value, standing for a misconfigured pair rather than for an attacker guessing a PPK.; Whether the responder aborts in IKE_AUTH when the confirmed PPK is not listed for the peer identity (main s0029) is not modelled.

- Open: The MAY-continue option of main s0029, and the abort-versus-wait behaviour discussed in RFC 8784 section 6.
- Next step: Model the identity-to-PPK policy of main s0029 to see what the MAY option gives an attacker.

R05.9: PPK policy, as asked in Q3

Under a mandatory PPK policy, IKE SA establishment must fail if the responder has no matching PPK or the value does not match; under an optional policy, if the initiator requires PPK protection, a session without PPK must not be accepted by the initiator as a successful PPK session.

For question Q3: neither the failure answer nor the fallback to plain IKEv2 is treated as a successful PPK session, under the policies the specification states. A mandatory responder stops with AUTHENTICATION_FAILED, and an initiator that requires PPK aborts instead of continuing.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: Reproduces the statement of RFC 8784 that the downgrade is detected when PPKs are mandatory for at least one peer.
- Stated in the documents: yes. Where the documents say it: s0021, s0023, s0025
- Rests on: The same modelling of the policy branches as for R05.; The DoS and retransmission advice of RFC 8784 section 6 is not modelled, so the model says nothing about abusing the abort.
- Open: The DoS consequence of an injected response without USE_PPK_INT (RFC 8784 section 6).
- Next step: Model the retransmission behaviour to check the availability advice of RFC 8784 section 6.

X01: PPK fields hidden under broken DH

An attacker that can decrypt the initial IKE SA (knows g^ir) does not learn the PPK_ID the initiator proposed or the PPK Confirmation value it sent in IKE_INTERMEDIATE.

The IKE_INTERMEDIATE request is protected only by the keys from IKE_SA_INIT, so anyone who can read that exchange reads the PPK_ID and the PPK Confirmation: a post-quantum attacker that knows g^ir reads them off the wire, and a network attacker reads them by answering IKE_SA_INIT itself and so becoming the Diffie-Hellman peer. What leaks is which PPK group is used and a value that lets a guess of the PPK be tested, not the PPK itself.

- Result: Fails (2 attacks); answers what is asked: yes; usable with caveats
- Value medium: RFC 9242 says intermediate data is protected from passive eavesdroppers, and RFC 8784 says a quantum attacker that decrypts the initial IKE SA sees everything except keys protected by the PPK. The main document does not say what the intermediate exchange exposes, so the consequence is implied rather than stated.
- Stated in the documents: unclear. Source text it rests on: s0013, s0016, s0561
- Rests on: The confirmation is modelled as the full prf output rather than its first eight octets.; The network attacker is allowed to answer IKE_SA_INIT with a group element of its choosing, which a validating implementation would reject.; PPK_IDs are treated as not secret, following the design of the notification.
- Open: Whether the exposure of an eight-octet confirmation matters in practice, which depends on PPK entropy and is a computational question.
- Next step: Ask a computational prover or a cryptographic argument what an eight-octet confirmation and a PPK_ID leak per session allow over many sessions.

X02: PPK not recoverable by offline guessing

An attacker that obtains a PPK Confirmation value (by impersonating the responder in IKE_SA_INIT/IKE_INTERMEDIATE) cannot test guesses of a person-chosen PPK against it offline and recover the PPK.

A PPK that people choose, rather than a random 256-bit value, can be recovered by testing guesses against the PPK Confirmation of one exchange, which an attacker obtains by impersonating the responder. With a random PPK of the size RFC 8784 requires this is not feasible, so the result restates the deployment requirement.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value low: RFC 8784 states in its security considerations that an adversary impersonating the server can perform a dictionary search over plausible PPK values even with a conventional computer, and that deriving a PPK from a password or name is not secure.
- Stated in the documents: yes. Where the documents say it: s0013
- Rests on: The dictionary attacker is modelled by the weak secret test, which assumes the attacker can test a guess against a value it saw.; The confirmation is not truncated in the model; the real eight-octet truncation bounds each test rather than preventing it.
- Next step: None: the requirement on PPK entropy is the remedy and it is already stated.

C04: IKE_AUTH protected when Diffie-Hellman breaks

When an attacker obtains the Diffie-Hellman shared secret g^ir from IKE_SA_INIT but not the PPK, it cannot derive the contents of the IKE_AUTH messages or the new IKE SA keys derived from SKEYSEED'.

A post-quantum attacker that knows g^ir but not the PPK obtains neither the new IKE SA keys nor the IKE_AUTH payloads. The reason is that SKEYSEED' is computed from the PPK and the previous SK_d, so an attacker holding only the old shared secret cannot follow the key update.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: RFC 8784 states that a quantum attacker that decrypts the initial IKE SA sees everything except keys protected by the PPK; the analysis reproduces that for IKE_AUTH in RFC 9867.
- Stated in the documents: yes. Where the documents say it: s0033, s0035, s0039
- Rests on: The PPK is assumed secret and of high entropy, as RFC 8784 requires.; prf+ is idealised, and secrecy is shown by an explicit test in which the attacker offers the keys it claims to know.; The attacker is given g^ir as a public function of the transcript, which is stronger than any real quantum attack modelled here.
- Open: Whether the identities and the other IKE_AUTH fields are hidden from such an attacker: the model protects the payload as a whole.
- Next step: Separate the fields of IKE_AUTH in the model to report which of them a post-quantum attacker learns.

U01: DH break, as asked in Q2

When an attacker obtains g^ir but not the PPK, it cannot derive the IKE_AUTH contents or the new IKE SA keys derived from SKEYSEED'.

For question Q2 the answer is the same as for C04: with g^ir leaked and the PPK secret, neither the IKE_AUTH content nor the new IKE SA keys can be derived. The PPK mixing is the last action before IKE_AUTH, so nothing protected with the old keys carries into it.

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

- Value medium: Same statement of RFC 8784 as for C04.
- Stated in the documents: yes. Where the documents say it: s0033, s0035
- Rests on: The same idealisations as C04: a secret PPK of high entropy, idealised prf+, and g^ir public in the post-quantum world.
- Open: The IKE_INTERMEDIATE exchange itself is not protected from such an attacker; see the finding on PPK fields.
- Next step: None for this claim; the exposure of the intermediate exchange is covered by the separate goal on PPK fields.

Appendix B: Known results reproduced

A property that fails as the documents already state, or that they set aside, is not a finding. What follows is of that kind: the analysis reproduces it on the model, which gives the exact conditions.

B.1 A low-entropy PPK is recoverable from one confirmation, as the documents require it not to be

Kind	Weakness of the protocol as specified
Stated in the documents	rfc8784:541
Value	low
Properties	X02, X01 (section 3)
Source text	s0013
Shown by	ProVerif attack query `x02_guess`

What happens. The PPK Confirmation is prf(PPK, Ni|Nr|SPIi|SPIr) truncated to eight octets, and any party that plays the responder of IKE_SA_INIT obtains it. An attacker holding a few plausible PPK values can test them against that value offline and learn whether one of them is the PPK in use.

What the attacker needs. The ability to answer an IKE_SA_INIT request and read the IKE_INTERMEDIATE request that follows, which any on-path attacker has, plus a PPK drawn from a small set of candidates.

What the attacker gains. For a PPK chosen by people the quantum protection of that SA is lost, and the attacker can then impersonate the peers to each other, since the authentication is bound to the PPK as well as to the signatures.

Appendix C: Claims and prover answers

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

target	level	bound	lemma	control	attacks	claims
A03	proved		auth_net	checked		4 proved
A04	proved		inj_net	checked		2 proved
A08	proved		auth_net	checked		2 proved
K01	proved		ppk_sec	checked		4 proved
R05	proved		r05_mand	checked		2 proved
C04	proved		keys_crqc	checked		1 proved
A08.7	proved		auth_net	checked		2 proved
U01	proved		keys_crqc	checked		1 proved
R05.9	proved		r05_mand	checked		2 proved
X01	none				x01_crqc (holds_in_source); x01_net (holds_in_source)	2 attack
X02	none				x02_guess (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
A03	The responder accepts the initiator's AUTH payload in IKE_AUTH only if a matching initiator run sent it for the same Ni, Nr, SPIi, SPIr and the same IKE_INTERMEDIATE transcript, under the network-only attacker.	`network`	proved	auth_net	checked
A03	The matching-run guarantee for the responder holds even for an attacker that knows g^ir and the long term signature keys of both peers, as long as the PPK is secret.	`crqc`	proved	auth_crqc	checked
A03	The initiator accepts the responder's AUTH payload in IKE_AUTH only if a matching responder run sent it for the same Ni, Nr, SPIi, SPIr and the same IKE_INTERMEDIATE transcript, under the network-only attacker.	`network`	proved	authr_net	checked
A03	The matching-run guarantee for the initiator holds even for an attacker that knows g^ir and the long-term signature keys, as long as the PPK is secret.	`crqc`	proved	authr_crqc	checked
A04	Each IKE_AUTH acceptance by the responder in a PPK-protected IKE SA corresponds to a different initiator run: the same IKE_INTERMEDIATE/IKE_AUTH transcript is not accepted twice, under the network-only attacker.	`network`	proved	inj_net	checked
A04	Injectivity of IKE_AUTH acceptance in a PPK-protected IKE SA also holds for an attacker that knows g^ir and the long-term signature keys and does not know the PPK.	`crqc`	proved	inj_crqc	checked
A08	The PPK negotiation the responder accepts (PPK_ID and PPK Confirmation) belongs to this IKE SA's Ni, Nr, SPIi, SPIr, and the peer identity it authenticates in IKE_AUTH is the one that sent that IKE_INTERMEDIATE request.	`network`	proved	auth_net	checked

target	claim	attacker	answer	query	control
A08	The binding between the accepted PPK fields, the IKE_INTERMEDIATE transcript (IntAuth) and the authenticated identity also holds for an attacker that knows g^ir and the long-term signature keys.	`crqc`	proved	auth_crqc	checked
K01	SK_ei, SK_er, SK_ai, SK_ar, SK_pi and SK_pr as recomputed from SKEYSEED' = prf+(PPK, SK_d) are not derivable by the network-only attacker.	`network`	proved	keys_net	checked
K01	The PPK-derived IKE SA keys SK_* stay underivable even for an attacker that knows g^ir and the long-term signature keys.	`crqc`	proved	keys_crqc	checked
K01	The PPK value itself is not derivable by the network-only attacker.	`network`	proved	ppk_sec	checked
K01	The PPK value is not derivable by an attacker that knows g^ir and the long-term signature keys.	`crqc`	proved	ppk_sec	checked
R05	Under a responder policy that mandates PPK, no IKE SA is completed when the proposed PPK is not confirmed (AUTHENTICATION_FAILED instead).	`network`	proved	r05_mand	checked
R05	Under an optional responder policy, an initiator that requires PPK protection never completes a session in which no PPK was negotiated.	`network`	proved	r05_req	checked
C04	With g^ir known and the PPK secret, the new IKE SA keys derived from SKEYSEED' stay hidden, so the IKE_AUTH content protected with them is not readable.	`crqc`	proved	keys_crqc	checked
A08.7	Q1: accepted PPK_ID, PPK Confirmation and IntAuth correspond to this IKE_SA_INIT's nonces and SPIs and to the identity authenticated in IKE_AUTH, under the network-only attacker.	`network`	proved	auth_net	checked
A08.7	Q1: the same binding holds for an attacker that knows g^ir and the long-term signature keys but not the PPK.	`crqc`	proved	auth_crqc	checked
U01	Q2: with g^ir leaked and the PPK secret, the IKE_AUTH content and the new IKE SA keys are not derivable.	`crqc`	proved	keys_crqc	checked
R05.9	Q3: under a mandatory PPK policy no IKE SA is completed when the proposed PPK is not confirmed; the responder answers AUTHENTICATION_FAILED.	`network`	proved	r05_mand	checked
R05.9	Q3: an initiator that requires PPK protection never treats a PPK-less session as a successful PPK session.	`network`	proved	r05_req	checked
X01	An attacker that can read the initial IKE SA learns the PPK_ID and the PPK Confirmation sent in IKE_INTERMEDIATE.	`crqc`	attack (holds_in_source)	x01_crqc	
X01	A network attacker that impersonates the responder in IKE_SA_INIT learns the PPK_ID and the PPK Confirmation.	`network`	attack (holds_in_source)	x01_net	
X02	A person-chosen PPK can be recovered by testing guesses offline against the PPK Confirmation the attacker obtains by impersonating the responder.	`dictionary`	attack (holds_in_source)	x02_guess	

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-20261114T212413, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 52, prover 155 s, wall 17 min, cost 0.170 USD.

Context: the host removed stale parts from the conversation 1 time(s), 468267 characters in all (earlier reasoning, and what showed the model before its latest change). The trajectory file keeps everything.

ProVerif: 143 proofs run, 143 decided, 0 without an answer, 125 s, 27 answers recorded.

What was analysed

Protocol	IKEv2 PPK in IKE_INTERMEDIATE
Version	rfc9867
Analysed	A signed IKE_AUTH after a fixed IKE_INTERMEDIATE PPK negotiation; full use of this RFC's key update and authentication binding; PPK-mandatory and optional policies distinguished. CREATE_CHILD_SA and multiple intermediates excluded.
Not analysed	The symbolic model does not prove computational or post-quantum security of the algorithms; attack traces hold only under explicit assumptions and are not directly equivalent to specification flaws. Every property requires a reachability/positive control.
Source text	586 numbered passages (194124 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	3.1	Creating Initial IKE SA
main	3.1.1	Computing IKE SA Keys
main	4	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
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

Document	Section	Title
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
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

Document	Section	Title
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
rfc9242	3	Intermediate Exchange Details
rfc9242	3.1	Support for Intermediate Exchange Negotiation
rfc9242	3.2	Using Intermediate Exchange
rfc9242	3.3	The IKE_INTERMEDIATE Exchange Protection and Authentication
rfc9242	3.3.1	Protection of IKE_INTERMEDIATE Messages
rfc9242	3.3.2	Authentication of IKE_INTERMEDIATE Exchanges
rfc9242	3.4	Error Handling in the IKE_INTERMEDIATE Exchange

The purpose of the protocol as the source step located it in the documents: fix a single IKE_INTERMEDIATE PPK negotiation followed by a signed IKE_AUTH; use the key update and authentication binding of this RFC in full; distinguish between PPK mandatory and optional policies. Exclude CREATE_CHILD_SA and multiple intermediates.

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
A03	Matching run for IKE_AUTH acceptance	Each of the initiator and the responder, when it accepts the peer's AUTH payload in IKE_AUTH, is assured of a matching run of the peer with the same roles, identities, PPK_ID, Ni, Nr, SPIi, SPIr and IntAuth.	Main says the initial IKE SA, including the IKE_AUTH exchange, is quantum secure even against an active attacker because the PPK is mixed into SK_* and used in authentication; the AUTH payload is the authentication of the IKE SA (rfc7296 s0377) bound to the IKE_INTERMEDIATE content by IntAuth (rfc9242 s0565-s0574).	s0039, s0377, s0565, s0568, s0574
A04	Distinct peer runs per acceptance	Every accepted IKE_AUTH of a given IKE SA establishment corresponds to a different run of the peer: the same IKE_INTERMEDIATE/IKE_AUTH transcript is not accepted twice as a new successful session.	Message IDs must be verified in IKE_INTERMEDIATE (rfc9242 s0555) and IKE_AUTH_MID is part of IntAuth (rfc9242 s0569, s0573); the AUTH payload signs both nonces (rfc7296 s0377). The document does not state injectivity in so many words.	s0377, s0555, s0569, s0573

ID	Name	About	What the documents say	Where
A08	PPK fields bound to this session	The responder's acceptance of PPK_ID and PPK Confirmation, and both peers' acceptance of IntAuth and IKE_AUTH_MID, correspond to the Ni, Nr, SPIi, SPIr of this IKE_SA_INIT and to this IKE_INTERMEDIATE transcript, and agree with the identity authenticated in IKE_AUTH.	PPK Confirmation is prf(PPK, Ni|Nr|SPIi|SPIr) and lets the responder check it has the same PPK for the given PPK_ID (main s0013, s0014); IntAuth binds the IKE_INTERMEDIATE messages into AUTH (rfc9242 s0568-s0574). The responder may only notice during IKE_AUTH that the PPK is not associated with the initiator's identity (main s0029).	s0013, s0014, s0018, s0029, s0568, s0573
K01	Secrecy of PPK-derived IKE SA keys	The PPK and the IKE SA keys recomputed from it (SKEYSEED', SK_d, SK_ei, SK_er, SK_ai, SK_ar, SK_pi, SK_pr) are kept from the attacker.	SKEYSEED' = prf+(PPK, SK_d) and the SK_* keys are recomputed from it before IKE_AUTH (main s0032-s0037); rfc8784 says SK_d is a function of the PPK and, assuming a high-entropy PPK, a quantum attacker cannot recover it (rfc8784:556).	rfc8784:556, s0032, s0033, s0034, s0035, s0036
R05	No PPK-less session under mandatory policy	Under a PPK-mandatory responder policy an IKE SA is never created if no proposed PPK matches (AUTHENTICATION_FAILED instead); under an optional policy an initiator that requires PPK protection does not accept a PPK-less session as a successful PPK-protected session.	Main s0021 (mandatory: AUTHENTICATION_FAILED and abort), s0023-s0025 (optional: no PPK_IDENTITY, and the initiator MUST abort if quantum resistance is a requirement), s0006 and s0027 (Table 1), s0020 (unproposed identity is fatal); rfc8784:619-627 says a downgrade by an attacker with a quantum computer is detected when PPKs are mandatory for at least one peer.	rfc8784:619, s0006, s0020, s0021, s0023, s0025, s0027
C04	IKE_AUTH protected when Diffie-Hellman breaks	The IKE_AUTH content and the IKE SA keys derived from SKEYSEED' stay hidden from an attacker that knows the IKE_SA_INIT shared secret g^ir but not the PPK.	Main s0032-s0036 mixes the PPK into SK_d before IKE_AUTH, and s0039 states the IKE_AUTH exchange is quantum secure even against an active attacker; rfc8784:567 says a quantum attacker that decrypted the initial IKE SA sees everything except keys protected by the PPK.	rfc8784:567, s0032, s0035, s0036, s0039
A08.7	PPK binding, as asked in Q1	Same property as A08, asked by the user as question Q1: accepted PPK_ID, PPK Confirmation, IntAuth and IKE_AUTH_MID belong to this IKE_SA_INIT's nonces and SPIs, to this IKE_INTERMEDIATE transcript and to the identity authenticated in IKE_AUTH.	main s0013, s0014, s0018, s0029; rfc9242 s0568-s0574.	s0013, s0014, s0018, s0029, s0568, s0573
U01	DH break, as asked in Q2	Same property as C04, asked by the user as question Q2: with g^ir leaked and the PPK secret, IKE_AUTH content and the new IKE SA keys are not derivable.	main s0032-s0036, s0039; rfc8784:556, rfc8784:567.	rfc8784:567, s0032, s0035, s0039
R05.9	PPK policy, as asked in Q3	Same property as R05, asked by the user as question Q3: fallback and authentication failure must not be treated as a successful PPK session.	main s0021, s0023, s0025, s0027; rfc8784:619-627, rfc8784:646-650 (with an optional policy and misconfiguration the SA is created without quantum protection).	rfc8784:619, s0021, s0023, s0025, s0027

ID	Name	About	What the documents say	Where
X01	PPK fields hidden under broken DH	the PPK_ID and PPK Confirmation of the initiator's IKE_INTERMEDIATE request, kept from an attacker who can read the initial IKE SA	RFC 9242 states IKE_INTERMEDIATE data is protected from passive eavesdroppers by the Encrypted payload and only authenticated in IKE_AUTH (rfc9242:463), and RFC 8784 states a quantum attacker that decrypts the initial IKE SA sees all information exchanged over it except keys protected by the PPK (rfc8784:567). The main document does not say what the IKE_INTERMEDIATE PPK fields expose.	rfc9242:463, rfc8784:567, s0013, s0016, s0561
X02	PPK not recoverable by offline guessing	the PPK value, chosen by people, kept from an attacker who can mount an offline dictionary attack on the PPK Confirmation	RFC 8784 says a server-impersonating adversary can perform a dictionary search over plausible PPK values with a conventional computer, that a PPK must have at least 256 bits of entropy, and that deriving it from a password or name is not secure (rfc8784:541-554); RFC 7296 says the same of pre-shared keys (rfc7296:6558).	rfc8784:541, rfc7296:6558

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

The security statements of the documents, paragraph by paragraph

The host listed the 46 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: 8. Not taken: 38. 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:479-485	A03, K01, C04, U01, X01, `crqc`	
P02	main:487-488	No goal stated	It only points to RFC 9242 for related security issues; the goals themselves are stated in the paragraphs of that document.
P03	main:490-493	The prover cannot be asked	It states a belief about the strength of symmetric primitives under a CRQC, which a symbolic prover cannot be asked; it is used only to bound the crqc attacker.
P04	rfc7296:6478-6488	Outside the scope	It is about probing for identities and CERTREQ/certificates of unauthenticated peers; certificates and the peer identity probing scenario are not modelled here.
P05	rfc7296:6490-6494	Outside the scope	EAP authentication is excluded from the scope of this task.
P06	rfc7296:6496-6500	Outside the scope	It is about repeated CREATE_CHILD_SA rekeying without Diffie-Hellman, which the task excludes.
P07	rfc7296:6502-6514	The prover cannot be asked	It estimates the cryptographic strength of Diffie-Hellman groups, which a symbolic model cannot express.
P08	rfc7296:6516-6519	The prover cannot be asked	Same subject: the strength and choice of the Diffie-Hellman group and PRF.
P09	rfc7296:6525-6527	No goal stated	It praises the extensible framework and mentions elliptic curve groups; nothing that must hold or can be attacked.
P10	rfc7296:6529-6530	No goal stated	It is an assumption that implementations erase DH exponents after use; in the model exponents are never given to the attacker in the network world, which is the stronger reading.

Paragraphs	Where	Taken up as	Why not
P11	rfc7296:6532-6538	The prover cannot be asked	It is about robustness against DoS from unauthenticated peers, which is availability and outside what the prover can be asked.
P12	rfc7296:6540-6542	The prover cannot be asked	It is a lower bound on the PRF output size, the strength of a primitive.
P13	rfc7296:6544-6549	The prover cannot be asked	It is a requirement on the randomness of the parties, not a property of traces.
P14	rfc7296:6551-6556	The prover cannot be asked	It restricts which algorithms may be negotiated (no NONE integrity, no ENCR_NULL), the strength of primitives.
P15	rfc7296:6558-6563	X02, `dictionary`	
P16	rfc7296:6565-6572	Outside the scope	It is about the NAT_DETECTION notifications and internal address disclosure; NAT traversal is outside the modelled flow.
P17	rfc7296:6578-6580	Outside the scope	It continues the NAT_DETECTION address disclosure discussion.
P18	rfc7296:6582-6606	Outside the scope	It is about EAP methods that do not generate a key, excluded from the scope.
P19	rfc7296:6608-6612	Outside the scope	It is about EAP server authentication requirements, excluded from the scope.
P20	rfc7296:6614-6619	The prover cannot be asked	It is about exhausting reassembly buffers with long messages, an availability issue.
P21	rfc7296:6621-6625	No goal stated	It is deployment advice on admission control and trust anchors, not a property of the protocol's traces.
P22	rfc7296:6631-6633	No goal stated	It says an explicit security policy for peer identity and credentials is essential; it states no property to prove.
P23	rfc7296:6637-6643	Outside the scope	Traffic Selector authorization for Child SAs is excluded from the scope.
P24 to P30	rfc7296:6645-6696	Outside the scope	The rest of Section 5.1 is about Traffic Selector authorization, which the task excludes.
P31	rfc8784:541-554	X02, `dictionary`	
P32	rfc8784:556-565	K01, C04, U01, `crqc`	
P33	rfc8784:567-572	C04, U01, X01, `crqc`	
P34	rfc8784:574-584	Outside the scope	It is about rekeying the IKE SA before sensitive information is sent and childless IKE SAs (RFC 6023), which the task excludes.
P35	rfc8784:586-591	The prover cannot be asked	It is advice on which symmetric algorithms are quantum-safe, the strength of primitives.
P36 to P39	rfc8784:593-617	The prover cannot be asked	It is a list of algorithm key sizes and PRFs that should not be used, the strength of primitives.
P40	rfc8784:619-627	R05, R05.9, `crqc`	
P41	rfc8784:629-644	The prover cannot be asked	It is about a DoS/injection attack that makes the initiator abort and about retransmission timing, availability, which the prover cannot be asked.
P42	rfc8784:646-650	R05, R05.9	
P43	rfc9242:463-470	X01, `crqc`	

Paragraphs	Where	Taken up as	Why not
P44	rfc9242:472-477	No goal stated	It is about IKE fragmentation of large intermediate exchanges, outside the modelled flow (a reference to RFC 7383).
P45	rfc9242:479-499	The prover cannot be asked	It is about a DoS attack by a malicious initiator that transfers data and aborts, an availability and cost property.
P46	rfc9242:501-514	No goal stated	It is about error handling and the trade-off between diagnostics and DoS; it states no property of the exchange that the prover can be asked.

The parties and the protected values, one by one

The host listed the 29 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: 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	Initiator	party	Everything lost by definition	A dishonest initiator holds its own long term signing key and the PPK it shares. With those, impersonating that very identity is not a protocol failure. The task asks no property against a peer that hands over its own credential, and with a single modelled identity nothing further can be asked.
S02	Responder	party	Everything lost by definition	A dishonest responder holds the PPK and its own signing key, so the initiator's assurance about that peer and the secrecy of the PPK-derived keys are lost by definition. No target asks for protection against a peer that is itself corrupt.
S03	IKE SA identity binding	value	Nothing gained	The identity binding is what the parties verify. Holding it gives the attacker no value it does not already compute from the transcript.
S04	PPK_ID and PPK Confirmation	value	`dictionary`	
S05	IKE_SA_INIT nonces and SPIs	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.
S06	IKE_INTERMEDIATE transcript	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.
S07	IKE_AUTH transcript	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.
S08	IntAuth and IKE_AUTH_MID	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.
S09	Message ID	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.
S10	PPK_ID	value	Nothing gained	Nonces, SPIs, Message IDs, the IKE_INTERMEDIATE and IKE_AUTH transcripts, IntAuth and PPK_ID travel on the wire. An attacker that holds them has nothing that the network attacker lacks, and the PPK_ID is not a secret.

ID	Party or value	Kind	Attacker	Why none
S11	PPK Confirmation	value	`dictionary`	
S12	IntAuth	value	Nothing gained	IntAuth, IKE_AUTH_MID and the identities are on the wire (inside the encrypted payload). Holding them adds nothing to what the network attacker has.
S13	IKE_AUTH_MID	value	Nothing gained	IntAuth, IKE_AUTH_MID and the identities are on the wire (inside the encrypted payload). Holding them adds nothing to what the network attacker has.
S14	Identities IDi/IDr	value	Nothing gained	IntAuth, IKE_AUTH_MID and the identities are on the wire (inside the encrypted payload). Holding them adds nothing to what the network attacker has.
S15	PPK	value	`ppk_revealed`	
S16	SKEYSEED'	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S17	SK_d	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S18	SK_ei	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S19	SK_er	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S20	SK_ai	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S21	SK_ar	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S22	SK_pi	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S23	SK_pr	value	Everything lost by definition	An attacker that holds SKEYSEED' or one of the SK_* keys has exactly what the secrecy targets say it must not. The property is lost by definition, and no target asks what happens if a key is handed over.
S24	PPK policy	value	Nothing gained	The PPK policy, the PPK_IDENTITY notification and the AUTHENTICATION_FAILED notification are not secrets. Holding them gives no advantage beyond the network attacker.
S25	PPK_IDENTITY notification	value	Nothing gained	The PPK policy, the PPK_IDENTITY notification and the AUTHENTICATION_FAILED notification are not secrets. Holding them gives no advantage beyond the network attacker.

ID	Party or value	Kind	Attacker	Why none
S26	AUTHENTICATION_FAILED notification	value	Nothing gained	The PPK policy, the PPK_IDENTITY notification and the AUTHENTICATION_FAILED notification are not secrets. Holding them gives no advantage beyond the network attacker.
S27	g^ir	value	`crqc`	
S28	IKE_AUTH content	value	Everything lost by definition	An attacker that holds the IKE_AUTH content or the new IKE SA keys has broken exactly the confidentiality that C04 and K01 ask for. The target asks whether it can derive them, so holding them is the negation, not a further attacker.
S29	New IKE SA keys	value	Everything lost by definition	An attacker that holds the IKE_AUTH content or the new IKE SA keys has broken exactly the confidentiality that C04 and K01 ask for. The target asks whether it can derive them, so holding them is the negation, not a further attacker.

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`	the IKE_SA_INIT shared secret g^ir (so it can read every encrypted message of the initial IKE SA) and the long-term signature keys of both peers, from the start of the session; symmetric primitives stay out of reach	Main s0039 claims quantum security of the IKE_AUTH exchange even against an active attacker, and s0041 says only symmetric primitives are believed unaffected by a CRQC; rfc8784:567 describes a quantum attacker that decrypts the initial IKE SA.	rfc8784:567, s0039, s0041
`dictionary`	several candidate PPK values and the PPK Confirmation value of one IKE_INTERMEDIATE exchange, obtained by impersonating the responder in IKE_SA_INIT and reading the IKE_INTERMEDIATE request; it can test a guess offline	rfc8784:541-554 (dictionary search over plausible PPK values by a server-impersonating adversary) and rfc7296:6558 (low-entropy pre-shared keys).	rfc8784:541, rfc7296:6558
`ppk_revealed`	the PPK itself, as a long-term shared secret that leaked	RFC 8784 states the quantum security of every key rests on the PPK (rfc8784:556); it is used to show that the proofs of K01 and C04 rest on nothing but the PPK.	rfc8784:556

ID	`network`	`crqc`	`dictionary`	`ppk_revealed`
A03	Holds	Holds		
A04	Holds	Holds		
A08	Holds	Holds		
K01	Holds	Holds		
R05	Holds			
C04		Holds		
A08.7	Holds	Holds		

ID	network	crqc	dictionary	ppk_revealed
U01		Holds		
R05.9	Holds			
X01	Fails	Fails		
X02			Fails	

Source and targets

Source: 586 units, 194124 bytes, frozen in source-delivery-01.json. Targets: 7 approved (A03, A04, A08, K01, R05, C04, U01); 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: approve the source package protocol-f7069cbdfce35279198aba406f93

This step as it reports itself: 1 requests (1 failed), 0.000 USD, 84068 tokens in and 5032 out, 31 s.

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
42	A03, A04, A08, A08.7, K01, R05, R05.9, C04, U01, X01, X02	plain_primitives, many_sessions, other	The model uses only hashing, a keyed PRF/KDF, AEAD-style encryption plus MAC, signatures and Diffie-Hellman as a plain key agreement with the single law g^xy = g^yx; the answers do not rest on any algebraic law beyond that. Every property is asked for any number of sessions: both roles are replicated. Further: the PPK Confirmation and the AUTH payload are truncated/serialised on the wire in the protocol, which a symbolic model cannot express, and the post-quantum attacker is modelled by making g^ir a public function of the transcript and publishing the long-term signature keys instead of modelling a quantum attack on the primitives.

The analyst's closing note

Model built: one run of IKE_SA_INIT, one IKE_INTERMEDIATE carrying PPK_IDENTITY_KEY, the PPK key update (SKEYSEED' = prf+(PPK, SK_d)) and signed IKE_AUTH with RFC 9242 IntAuth, with a policy branch for a mandatory/optional responder and for an initiator that requires PPK. Two attacker worlds run side by side in the same file: the network-only attacker, and the post-quantum one modelled by making g^ir a public function of the transcript and publishing both signing keys.

Eleven targets, all answered (22 recorded answers, all re-verified on the final text). Proved: both directions of IKE_AUTH correspondence for the network attacker and for the attacker that has broken Diffie-Hellman and the signatures (A03); injectivity of IKE_AUTH acceptance in both worlds (A04); the PPK_ID/confirmation-to-session binding and the identity binding (A08, A08.7, partly: one identity only); the PPK and the PPK-derived SK_* keys unreachable for both attackers (K01); neither the IKE_AUTH content nor the new keys derivable when g^ir is known (C04, U01); the fail-closed policy rules of section 3.1 (R05, R05.9). Each proof comes with a control query that ProVerif falsifies.

Two attacks, both named in the documents for an attacker with at least what mine needs, so reported rather than presented as new: the PPK_ID and the PPK Confirmation of IKE_INTERMEDIATE are readable whenever the initial IKE SA is readable, which a post-quantum attacker has, and a network attacker gets the same by answering

IKE_SA_INIT itself (X01, novelty unclear); and a person-chosen PPK falls to offline guessing against that confirmation (X02, recorded in RFC 8784 section 6).

What blocks more: the model idealises the primitives, gives the post-quantum attacker g^ir instead of attacking the primitives, does not model Key Exchange payload validation, Message ID counters, several IKE_INTERMEDIATE exchanges, several PPK identities or the identity-to-PPK policy of section 3.1, and cannot speak about availability; the PPK Confirmation is modelled as the full prf output rather than its first eight octets. Tamarin with a bounded state and an explicit byte layout would settle the Message ID and truncation questions.