Security analysis of CMS composite KEM encryption profile

draft-ietf-lamps-cms-composite-kem-04 · ProVerif+ Agent report

Documents	Composite ML-KEM for use in Cryptographic Message Syntax (CMS) (protocol definition); Using Key Encapsulation Mechanism (KEM) Algorithms in the Cryptographic Message Syntax (CMS) (supporting)
Properties examined	13: 8 hold, 5 fail (5 attacks)
Findings	None

Contents

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

1 Protocol and scope

The profile carries a content-authenticated-encryption key from an originator to one recipient inside a CMS AuthEnvelopedData, using one of the composite ML-KEM algorithms registered in section 3 of the draft. The originator holds the recipient's certified composite public key, an ML-KEM public key and a traditional public key, makes the content key itself, encapsulates to both components and sends a KEMRecipientInfo with the wrapped key and the encrypted content. The recipient holds the matching composite private key, an ML-KEM decapsulation key and the traditional private key.

When the exchange ends the recipient holds the originator's content key and the originator's content, intact, for the entry it was addressed by, but it has no information about which party sent that entry. The originator learns nothing back: the profile has no response, no key confirmation and no session to keep.

The parties:

- **Originator**: makes the content key, encapsulates to the recipient, wraps. Holds: recipient's composite public key (pkML-KEM, pktrad) from its certificate; the content.
- **Recipient**: decapsulates with its composite private key, unwraps, decrypts. Holds: composite private key (skML-KEM, sktrad); its certificate with rid.

Figure 1 shows the exchange as the documents specify it.

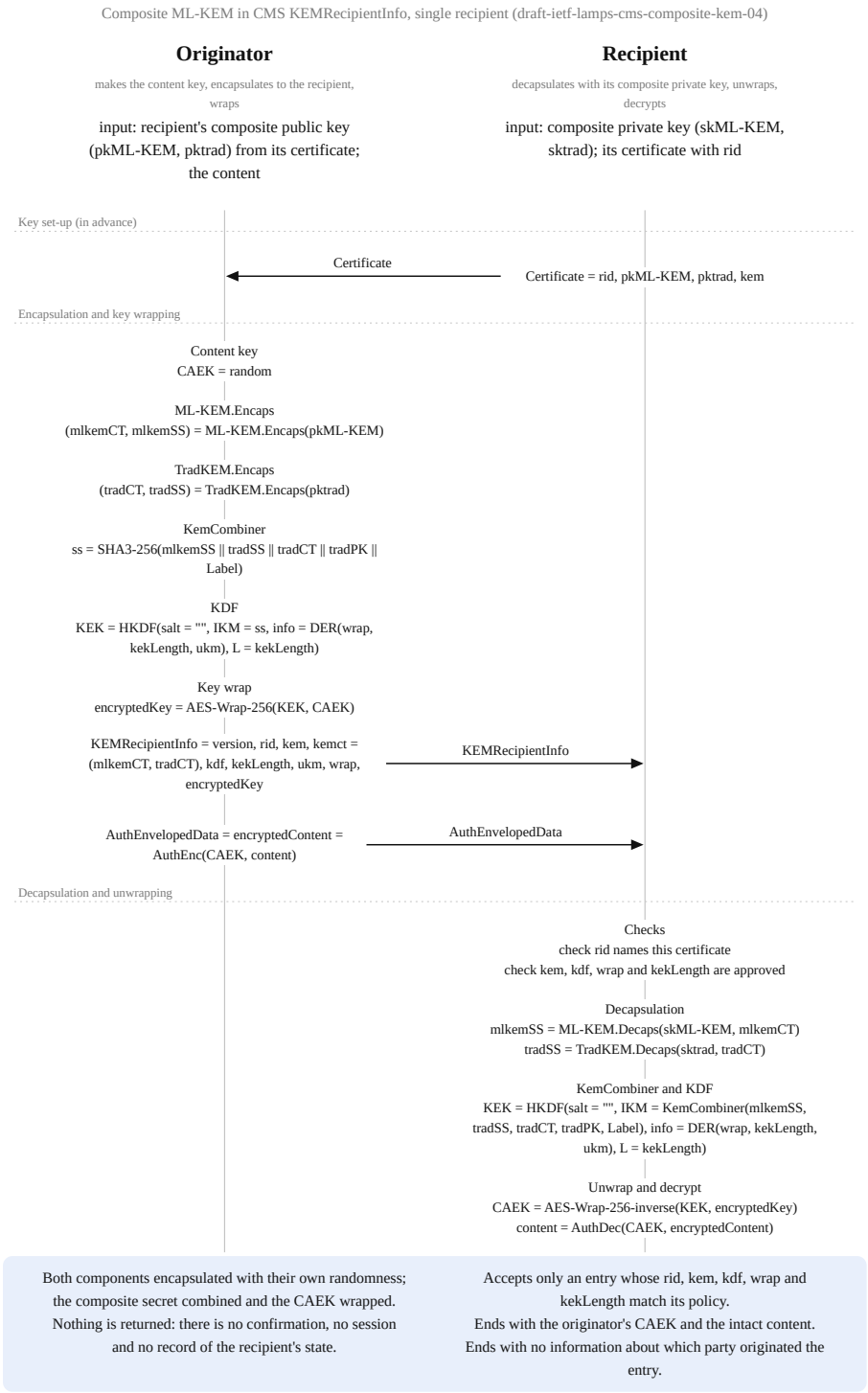


Figure 1. The protocol as specified.

The scope of the analysis is in Appendix D: what was analysed, what was left out, and the sections of the documents it is built from.

2 Findings

No finding. The analysis found nothing that the documents do not already state or set aside. The table in section 3 has the result for every property.

Within the scope analysed the profile gives what it is meant to give: the pairwise key-encryption key, the content key and the content stay out of reach of the network attacker, the key-encryption key is bound to the ukm, the ciphertexts and the suite that produced it, one broken component leaves the secret intact, and a disclosed message key reaches only that message. There is no finding: every failure the model reproduces is stated in the documents, namely that the recipient can attribute an accepted entry to no party, that a later disclosure of the composite private key opens completed messages, and that losing both component keys loses every message protected with the key pair. Of the thirteen targets, the proofs cover nine and the remaining claims are attacks that hold in the source. The proofs rest on idealised primitives: the composite KEM, the combiner, the key derivation, the key wrap and the content encryption are constructors with their inverse laws and no algebra, so component security, collision resistance and cipher strengths are assumed. Tamarin+ Agent, which models the components and their algebra as they are, would be needed to carry the results to the algorithms themselves.

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
K01	**Keys and private key kept secret.** Every secret (receiving combination) for which a script is called inKEMPrivate key, byKEMSharing a secret birth pair.KEK、CEK/CAEK, message authentication key) the value cannot be derived from the protocol traffic by the attacker who does not hold the corresponding leak; GroupingKEMWhen the private key is leaked, all information in the recorded script is jeopardized.	recipient (disclosure at any time during or after the protected session)	Attack holds (stated in the documents)	medium	yes
K02	**Content confidentiality under theCAEK.** Yes.CEK/CAEKThe encrypted content can only be read by the intended recipient; Do not hold the recipient's private key orCEK/CAEKThe attackers could not resume the content.	no compromise	Holds	medium	yes
K04	**KEKseparation across derivation contexts.** For different wrap algorithms, differentkekLength, different km or different combinationsKEMKey pair.KEKDifferent from one another; It's a context.KEKIt's not a different context.KEKUse.	no compromise	Holds	low	yes
A03	**Recipient agreement on KEK and identity.** When the recipient accepts KEMRecipientInfo and gives birth to KEK, there is a matching sponsor operating, encapsulating the recipient ' s public key and producing the same shared secret as KEK and matching the recipient ' s identity.	no compromise	Holds	low	no
A08	**KEMRecipientInfofields bound into theKEK.** Receiving only Kem, kdf, wrap marking,kekLengthSame as km and approved context, and it's tied in.KEKIt's only accepted when it's derived.KEMRecipientInfo。	no compromise	Holds	low	yes

ID	Property	Compromise assumed	Result	Value	In the documents
D01	**Content integrity without origin authentication.** The recipient's acceptance of the certified content is true of the sponsor and is not supported by an unacknowledged message authentication key orCAEK(a) The aggressor ' s modification;AEADIntegrity is not interpreted as authentication of the identity of the person who sent it arbitrarily.	no compromise	Attack holds (stated in the documents)	medium	yes
C08	**Hybrid security with one component broken.** In GroupKEMWhen one of the points fails (the private key is leaked) and the other one is perfect, it becomes the sharing of secrets and the sharing of secrets.KEKThe perpetrators of the attack remain confidential.	recipient (break before or during the protected session)	Holds	medium	yes
C08.8	**CEKsecrecy under partial or full leakage.** Leaked in traditional weight private keys,ML-KEMIf the private key is leaked or both, check.CEK/CAEKWhether or not the attackers remain confidential.	Reporter (during or after session)	Attack holds (stated in the documents)	medium	yes
A08.9	**Combiner binds label, ciphertexts, keys.** Of the KEMRecipientInfo accepted by the recipient, kem, kdf, wrap identification, kekLength, ukm is consistent with the recipient public key/certificate and entered as the context derived from KEK.	no compromise	Holds	low	yes
D01.10	**No sender authentication fromAEAD.** The recipient's acceptance of the protected content corresponds to the actual transmission of the authorized sender (the party holding the corresponding key) and has not been tampered with;AEADIntegrity is not interpreted as authentication of the identity of the person who sent it arbitrarily.	no compromise	Attack holds (stated in the documents)	medium	yes
X01	**Per-recipient encapsulation freshness.** No two KEMRecipientInfo entries of the originator share a KEM shared secret or encapsulation randomness: if two sent entries carry the same composite shared secret, they are the same entry.	no compromise	Holds	low	yes
X02	**Leakage confined to one message.** Disclosure of the KEK or of the CEK/CAEK of one message does not disclose the content of any other message; the content of another message is learned only when the recipient's composite private key is disclosed.	The key-encryption key, the content-encryption key and the content-authenticated-encryption key of one message of a session, obtained by the attacker during that session (for example by reading them from a party's memory).	Holds	medium	yes
X03	**Forward secrecy of past messages.** The content and the CEK/CAEK of a completed session stay secret when the recipient's composite private key is disclosed only after the session has ended.	The recipient's whole Composite ML-KEM private key, i.e. both component private keys, disclosed to the attacker at any time during or after the protected session.	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, X02, X03.

The questions asked, and the analyst's answers:

Question	Answer	Rests on
Q1: When a single fraction is leakedCEKConfidentiality and double disclosure controls	With one component of a composite key disclosed and the other intact, the content key of that recipient's entries stays secret, in both directions; only when both component keys are disclosed does the content key and the content fall, which the control run shows.	C08, C08.8
Q2: Grouper Public Key/Cypher/ AlgorithmIDContext bound	The suite label and the two ciphertexts are bound into the composite secret and through it into the key-encryption key, so an entry made under one suite, or with different ciphertexts, never yields another entry's key-encryption key; the binding of the ML-KEM ciphertext and public key rests on decapsulation alone, as the referenced combiner draft says, and the combiner is idealised in the model.	A08.9, A08, K04, A03
Q3: Don't putAEADCompleteness is interpreted as arbitrary sender identification	The recipient gets integrity but no sender authentication: a party holding the recipient's certificate can make an entry the recipient accepts with content of its own choosing, which the model reproduces and RFC 9629 states.	D01, D01.10

4 Limits of this analysis

A result here is a result on a model of the protocol: it says what the specified messages and checks give or fail to give, with cryptography taken as ideal, and nothing about an implementation.

What this analysis does not cover, as the analyst states it:

- The composite KEM is idealised: ML-KEM and the traditional component are abstract encapsulation functions whose shared secret is a function of the public key and fresh coins, so the chosen-ciphertext security of either component is assumed, not proved.
- The combiner is a free injective constructor instead of SHA3-256, so collisions, ciphertext second-preimage resistance and the binding hierarchy are assumed away; following the referenced draft, the ML-KEM ciphertext and public key are not explicit combiner inputs and are bound only through decapsulation, which the model reproduces symbolically.
- HKDF, AES key wrap and the content encryption are ideal constructors with their single inverse law; their strength and the strength of the component algorithms are outside the model.
- The recipient's composite public key is assumed authentic: certificate content, validation and substitution of the public key are not modelled.
- The associated data of the content encryption is a constant, so which CMS fields are authenticated by the encryption itself is not examined.
- One content type (authenticated-enveloped-data) and one recipient per entry are modelled; signed attributes, S/MIME capabilities, several recipients and the content-encryption profile are left out.
- The model has no value that changes across messages, so replay of a delivered entry, acceptance order and retransmission are neither proved nor disproved.
- Random-number quality, timing and other side channels, and the probability of guessing, are outside what a symbolic prover can be asked.
- The traditional component is not modelled as Diffie-Hellman: the model gives the attacker less than the algebraic laws of X25519 or ECDH, so an attack found would still be an attack while the proofs are proofs against a weaker attacker.

- The private-key disclosure is modelled as both component keys handed over after the exchange; partial disclosure of stored key material is modelled only as one whole component.

What the host records:

- The answers changed after the reading was written for: K04, A03, A08, A08.9, D01, X01.

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.

K01: Keys and private key kept secret

Every secret to name a script (receiving combination)KEMPrivate key, byKEMSharing a secret birth pair.KEK、CEK/CAEK, message authentication key) the value cannot be derived from the protocol traffic by the attacker who does not hold the corresponding leak; GroupingKEMWhen the private key is leaked, all information in the recorded script is jeopardized.

An attacker on the network reads every field of an entry and obtains neither the pairwise key-encryption key, nor the content key, nor the recipient's private key. The one case in which it obtains them is the one the documents state: once the recipient's composite private key is disclosed, it decapsulates the entry's two ciphertexts, derives the key-encryption key and recovers the content key and the content, for every message protected with that key.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value medium: The secrecy proofs hold with the idealisation listed; the disclosure result is what the documents state, now shown mechanically with the exact steps that give the content key.
- Stated in the documents: yes. Where the documents say it: main:494, rfc9629:568, s0067
- Rests on: The composite KEM, the combiner, HKDF, AES key wrap and the content encryption are ideal constructors.; The disclosure is modelled as both component private keys handed over; the message-authentication key of the authenticated-data content type has no separate counterpart because in AuthEnvelopedData the content-authenticated-encryption key carries it.; Each secrecy proof has its control, the same query without the disclosure excluded, which fails.
- Next step: Nothing further for the network attacker; the algebra of the components would need Tamarin+ Agent.

K02: Content confidentiality under the CAEK

Yes.CEK/CAEKThe encrypted content can only be read by the intended recipient; Do not hold the recipient's private key orCEK/CAEKThe attackers could not resume the content.

The content of an entry is not read by an attacker that has the network alone: it holds neither the recipient's private key nor the content key, and unwrapping fails without the key-encryption key the recipient derives. Content falls only when the recipient's composite private key, or the message's own content key, is disclosed.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A confidentiality proof for the central value with the idealisation listed; the documents require it, so it is not a new result.
- Stated in the documents: yes. Where the documents say it: main:490, s0067, s0078, s0087
- Rests on: The content encryption is an ideal authenticated encryption with one inverse law; no cipher properties are modelled.; The associated data is a constant, so the fields the encryption itself protects are not examined.
- Next step: Tamarin+ Agent would settle the strength of the idealised primitives.

K04: KEK separation across derivation contexts

For different wrap algorithms, differentkekLength, different km or different combinationsKEMKey pair.KEKDifferent from one another; It's a context.KEKIt's not a different context.KEKUse.

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

Two entries that end up with the same key-encryption key are the same entry, so different ukm values, different suites and different composite key pairs never give the same key-encryption key and none can be carried over to another context. The ukm carried in an entry is exactly the value the key-encryption key was derived from, and an entry presented with a changed ukm does not yield that key.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value low: It follows from treating the key derivation and the combiner as injective constructors, and one of the four context fields is not exercised.
- Stated in the documents: yes. Where the documents say it: rfc9629:378, s0116, s0117, s0124, s0034
- Rests on: Only the ukm is varied; the key-wrap algorithm and key length are the profile's constants, so their part of the separation follows from the profile rather than being exercised.; The key derivation and the combiner are injective constructors, which is what makes equal keys imply equal inputs.; Both claims have their control, which fails once the two events are no longer tied to one entry.
- Open: Only the ukm varies in the model: wrap and kekLength are the profile's single mandated values, and separation across key pairs is shown only across the two modelled suites.
- Next step: A model with several key-wrap algorithms and key lengths would exercise the rest of the separation.

A03: Recipient agreement on KEK and identity

When the recipient accepts KEMRecipientInfo and gives birth to KEK, there is a matching sponsor running, encapsulating the recipient's public key and producing the same shared secret as KEK and matching the recipient's identity.

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

When the recipient accepts a delivered entry it arrives at the same key-encryption key, content key and content as the originator, under the recipient identity and suite that the entry names, so the entry is not confused with another recipient's or another suite's. This says nothing about who sent the entry: the correspondence is with an entry, not with a party.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: The agreement follows from the derivation being deterministic given the entry, and the documents state no authentication goal, so this is my operational reading of the purpose rather than a stated property.
- Stated in the documents: no. Source text it rests on: s0002, s0078, s0102
- Rests on: The claim is about a delivered entry, identified by its ciphertexts; it does not say that a genuine originator stands behind it.; The identity binding is structural: each recipient compares the recipient identifier with its own and each suite uses its own label.; Its control, the same correspondence required of an entry the originator never sent, fails.
- Next step: An entry accepted by a party other than the addressed recipient would need a second suite sharing keys; the model shows the addressed-recipient case only.

A08: KEMRecipientInfo fields bound into the KEK

Receiver only kem, kdf, wrap marking,kekLengthSame as km and approved context, and it's tied in.KEKIt's only accepted when it's derived.KEMRecipientInfo。

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

The context values the recipient accepts are the ones the key-encryption key was derived from: an entry whose ukm or whose ciphertexts were changed does not yield the originator's key-encryption key, and an entry carrying another suite's identifier is accepted only by that suite's recipient. The recipient refuses anything whose identifiers do not match its approved set.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value low: It follows from the key derivation taking the received context fields as inputs, with only the ukm exercised.
- Stated in the documents: yes. Where the documents say it: rfc9629:378, main:368, s0106, s0116, s0117, s0124
- Rests on: The identifier checks are structural comparisons with the profile's constants; a recipient configured to accept several suites is not modelled.; The binding is shown for the ukm, the one context field that varies per message.; Both claims have their control, which fails once the two events are no longer tied to one entry.
- Open: The kem, kdf and wrap identifiers and kekLength are single values in this profile, so the model checks them but cannot vary them; the ukm is the context field whose binding is exercised.
- Next step: Several key-wrap algorithms and key lengths in one model would settle the rest.

A08.9: Combiner binds label, ciphertexts, keys

It's accepted by the recipientKEMRecipientInfo, kem, kdf, wrap marking,kekLength,ukm is consistent with the recipient public key/certificate and asKEKThe context of the derivative input.

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

The suite label and the two ciphertexts are inputs to the composite secret, so an entry made under one suite, or with different ciphertexts, never yields another entry's key-encryption key. The binding of the ML-KEM ciphertext and public key comes only from decapsulation, not from an explicit combiner input, which is the design decision the referenced draft records.

- Result: Holds; answers what is asked: partly; usable with caveats
- Value low: The binding follows from the combiner's inputs being distinct constructor arguments, and the documents state the binding property themselves.
- Stated in the documents: yes. Where the documents say it: draft-ietf-lamps-pq-composite-kem:2588, s0012, s0041, s0066
- Rests on: The combiner is a free injective constructor rather than a hash, so the binding results rest on injectivity, not on collision resistance.; The traditional public key used at decapsulation is the recipient's own; the draft's requirement that a decapsulator obtain it out of band is assumed satisfied.; Both claims have their control, which fails once the two entries are no longer tied by the same key.
- Open: Collisions and ciphertext second-preimage resistance are assumed away by the idealised combiner.; The ML-KEM ciphertext and public key are bound only through decapsulation, as the referenced draft states.
- Next step: Tamarin+ Agent would need the algebraic properties of the hash and of the traditional key exchange to go further.

D01: Content integrity without origin authentication

The recipient's acceptance of the certified content is the real transmission to the sponsor and is not supported by an unacknowledged message authentication key orCAEK(a) The aggressor ' s modification;AEADIntegrity is not interpreted as authentication of the identity of the person who sent it arbitrarily.

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

For an entry the originator sent, the content the recipient accepts is the originator's own: an attacker on the network cannot substitute content into an entry that decrypts. Integrity ends there, however, because an entry that no originator sent is accepted just as readily, so the recipient cannot tell an honest message from one the attacker made up.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value medium: The integrity half holds with the idealisation listed, and the missing origin authentication is reproduced mechanically with its exact condition, a certificate the attacker can read.
- Stated in the documents: yes. Where the documents say it: rfc9629:563, s0067, s0002
- Rests on: The associated data of the content encryption is a constant, so the fields the encryption itself protects are not examined.; Whether the originator is a party the recipient trusts at all is outside the model.
- Next step: Nothing further; the absence of origin authentication belongs to CMS with a KEM recipient, not to the composite construction.

D01.10: No sender authentication from AEAD

The protected content accepted by the receiving party corresponds to the actual transmission of the permitted sender (the party holding the corresponding key) and has not been tampered with; AEAD Integrity is not interpreted as authentication of the identity of the person who sent the information arbitrarily.

The recipient cannot establish which party sent an accepted entry: anyone who holds the recipient's certificate encapsulates to its public key, derives its own key-encryption key and sends content the recipient accepts. Authentication of the originator would have to come from CMS itself, for example a signature, not from the key transport.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable
- Value medium: A documented property of KEM recipient information, shown here on this profile's own message flow; it is the condition under which the profile is used rather than a defect of the composite construction.
- Stated in the documents: yes. Where the documents say it: rfc9629:563, s0002
- Rests on: The recipient's certificate is public by design; the model publishes both recipients' public keys.; The content encryption is idealised, so the result concerns the key transport only.
- Next step: Nothing further for this profile; a profile that adds signatures would be a different analysis.

C08: Hybrid security with one component broken

GroupingKEMWhen one of the points fails (the private key is leaked) and the other one is perfect, it becomes the sharing of secrets and the sharing of secrets.KEKThe perpetrators of the attack remain confidential.

With the ML-KEM key of one composite key or the traditional key of another disclosed, the other component of each key intact, the key-encryption key and content key of those recipients' entries stay secret: the attacker computes one component secret but not the other, and the combiner is one way. Both directions of the hybrid property hold, and only disclosure of both component keys of the same key pair lets the secret fall.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A central guarantee of the profile, proved with the idealisation of the components and the combiner listed; the documents state it, so it confirms rather than discovers.
- Stated in the documents: yes. Where the documents say it: draft-ietf-lamps-pq-composite-kem:2588, s0066, s0075
- Rests on: The two single-component disclosures are modelled on the two modelled key pairs, so each claim tests one broken component with the other component of the same key pair intact.; The combiner, the key derivation and the components are idealised; the result is the symbolic counterpart of the hybrid argument, not the argument itself.; Each claim has its control, which fails once the whole composite key is disclosed.
- Next step: Tamarin+ Agent for the component algebra and the second-preimage assumptions.

C08.8: CEK secrecy under partial or full leakage

In the traditional mass private key leak,ML-KEMIf the private key is leaked or both, check.CEK/CAEKWhether or not the attackers remain confidential.

One broken component, in either direction, leaves the content key of that recipient's entries unreachable, while disclosure of both component keys of one composite key gives the content key and the content of every entry protected with that key. The single-leak and the double-leak case therefore come out differently, as the question expected.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable with caveats
- Value medium: It answers the question with both the verified case and the documented loss, at the cost of the idealisation listed.
- Stated in the documents: yes. Where the documents say it: main:494, rfc9629:568, s0066, s0067
- Rests on: The single-component disclosures are placed on the two modelled key pairs, so neither single leak combines with the other into a full key.; The composite KEM and its combiner are idealised.
- Next step: A model with partial disclosure of stored key material would cover a leak of part of one component's storage.

X01: Per-recipient encapsulation freshness

No two KEMRecipientInfo entries of the originator share a KEM shared secret or encapsulation randomness: if two sent entries carry the same composite shared secret, they are the same entry.

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

No two entries of the originator share a key-encryption key or a content key: entries that carry the same composite shared secret are the same entry, so the encapsulation randomness and the shared secret are never reused for another entry or another recipient.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value low: It follows from the coins being fresh per session and the constructors being injective; the requirement is stated in the documents.
- Stated in the documents: yes. Where the documents say it: rfc9629:575, s0069, s0072
- Rests on: Freshness is structural: each originator session makes its own coins, so the result rests on the constructors being injective.; The model has one content key per entry, as the profile does, and no case of deliberate reuse.; Its control, two unconstrained entries expected to share a key, fails.
- Next step: Nothing further; a second session of the same recipient would not change the argument.

X02: Leakage confined to one message

Disclosure of the KEK or of the CEK/CAEK of one message does not disclose the content of any other message; the content of another message is learned only when the recipient's composite private key is disclosed.

An attacker that obtains the key-encryption key and the content key of one message decrypts that message and nothing else: the content of every other message still needs its own key or the recipient's composite private key. The isolation holds because each message gets its own encapsulation and its own content key.

- Result: Holds; answers what is asked: yes; usable with caveats
- Value medium: A stated guarantee shown mechanically with its exact condition, namely one message's keys only.
- Stated in the documents: yes. Where the documents say it: rfc9629:569, s0067, s0116
- Rests on: The disclosure of one message's keys is modelled by a session that publishes its key-encryption key and content key.; Each message has fresh coins and a fresh content key, which is what the isolation rests on.
- Next step: Tamarin+ Agent would be needed if the algebra of the components mattered for the isolation.

X03: Forward secrecy of past messages

The content and the CEK/CAEK of a completed session stay secret when the recipient's composite private key is disclosed only after the session has ended.

A session that has already finished is opened by a disclosure of the recipient's composite private key that happens afterwards: the stored ciphertexts are decapsulated and the content key and the content follow. The profile gives no forward secrecy, and the documents state that the private key compromises all messages protected with it.

- Result: Attack holds (stated in the documents); answers what is asked: yes; usable
- Value medium: The absence of forward secrecy is stated in the documents; the model shows it on this profile with the exact step that opens a completed session.
- Stated in the documents: yes. Where the documents say it: main:494, s0067
- Rests on: The disclosure is modelled in a second phase, after the exchange, as both component private keys.; The ciphertexts of the entry are assumed kept, which any party on the network can do.
- Next step: Nothing further: the profile has no ratchet or rekeying that could restore secrecy.

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
K01	proved		kek_secret	checked	ctl_cek (holds_in_source)	3 proved, 1 attack
K02	proved		m_secret	checked		1 proved
K04	proved		kek_inj	checked		4 proved
A03	proved		agree	checked		2 proved
A08	proved		ukm_bound	checked		4 proved
D01	proved		agree	checked	forge (holds_in_source)	2 proved, 1 attack
C08	proved		one_mlkem	checked		2 proved
C08.8	proved		one_mlkem	checked	ctl_cek (holds_in_source)	2 proved, 1 attack
A08.9	proved		kek_inj	checked		4 proved
D01.10	reachable		run		forge (holds_in_source)	1 attack
X01	proved		kek_inj	checked		2 proved
X02	proved		confine	checked		1 proved
X03	reachable		run		ctl_cek (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
K01	The pairwise KEK of an originator entry is not obtained by an attacker that has only the network, during or after the session (the control, the same query without the phase restriction, fails once the private key is disclosed).	`network`	proved	kek_secret	checked
K01	The CEK/CAEK of an originator entry, which in AuthEnvelopedData also carries the message authentication, is not obtained by an attacker that has only the network.	`network`	proved	cek_secret	checked
K01	The recipient's composite private key is not derived from the traffic; the attacker reaches it only when it is disclosed (control: without that exclusion the query fails).	`network`	proved	sk_secret	checked
K01	Disclosure of the recipient's composite private key gives the attacker the CEK/CAEK of an entry protected with that key, and with it the content: the consequence the documents state for that disclosure.	`composite_key_leak`	attack (holds_in_source)	ctl_cek (the control of cek_secret)	
K02	The content of an originator entry, encrypted under the CEK/CAEK, is not obtained by an attacker that holds neither the recipient's private key nor the CEK/CAEK.	`network`	proved	m_secret	checked
K04	Two originator entries that yield the same KEK are the same entry: a different ukm, a different suite identifier or a different composite key pair never give the same KEK, so a KEK cannot be carried over from one derivation context to another (control: two unconstrained entries do not share a KEK).	`network`	proved	kek_inj	checked

target	claim	attacker	answer	query	control
K04	The KEK is derived from the ukm carried in the KEMRecipientInfo, so an entry presented with a different ukm does not yield the originator's KEK: the accepted context and the derived KEK belong together (control: without the condition that the two events concern one entry, a different ukm is accepted).	`network`	proved	ukm_bound	checked
K04	Two originator entries that yield the same KEK are the same entry: a different ukm, a different suite identifier or a different composite key pair never give the same KEK, so a KEK cannot be carried over from one derivation context to another.	`network`	proved	kek_inj	checked
K04	The KEK is derived from the ukm carried in the KEMRecipientInfo, so an entry presented with a different ukm does not yield the originator's KEK: the accepted context and the derived KEK belong together.	`network`	proved	ukm_bound	checked
A03	When the recipient accepts a delivered KEMRecipientInfo it arrives at the originator's KEK, CEK and content for that entry, and the recipient identity and suite identifier it accepted are the ones that entry names (control: an entry that no originator sent is accepted with no such matching send).	`network`	proved	agree	checked
A03	When the recipient accepts a delivered KEMRecipientInfo it arrives at the originator's KEK, CEK and content for that entry, and the recipient identity and suite identifier it accepted are the ones that entry names.	`network`	proved	agree	checked
A08	The ukm carried in the entry is exactly the value bound into the KEK derivation: an entry whose ukm was altered does not yield the accepted KEK, so the recipient never accepts a context other than the one the KEK was derived from (control: without the condition that the two events concern one entry, a different ukm is accepted).	`network`	proved	ukm_bound	checked
A08	The ciphertexts (kemct) and the suite identifier carried in the entry are the values bound into the KEK: an entry with replaced ciphertexts or another suite's identifier is not accepted together with the originator's KEK (control: without the condition that the two events concern one entry, other ciphertexts are accepted under the same ukm).	`network`	proved	ct_bound	checked
A08	The ukm carried in the entry is exactly the value bound into the KEK derivation: an entry whose ukm was altered does not yield the accepted KEK, so the recipient never accepts a context other than the one the KEK was derived from.	`network`	proved	ukm_bound	checked
A08	The ciphertexts (kemct) and the suite identifier carried in the entry are the values bound into the KEK: an entry with replaced ciphertexts or another suite's identifier is not accepted together with the originator's KEK.	`network`	proved	ct_bound	checked
D01	The recipient accepts an entry that no originator sent, carrying content the attacker chose: the integrity AuthEnvelopedData provides gives no origin authentication.	`network`	attack (holds_in_source)	forge	
D01	For an entry delivered from the originator, the content the recipient accepts is the originator's own content: an attacker on the network cannot substitute a different content into an entry that decrypts (control: an entry that no originator sent is accepted, so the condition on the send is what carries the claim).	`network`	proved	agree	checked

target	claim	attacker	answer	query	control
D01	For an entry delivered from the originator, the content the recipient accepts is the originator's own content: an attacker on the network cannot substitute a different content into an entry that decrypts.	`network`	proved	agree	checked
C08	With the ML-KEM decapsulation key of a composite key disclosed and the traditional component of that same key intact, the KEK and CEK of that recipient's entries stay secret (control: without the full-disclosure exclusion the query fails).	`mlkem_key_leak`	proved	one_mlkem	checked
C08	With the private key of the traditional component of a composite key disclosed and the ML-KEM key of that same key intact, the KEK and CEK of that recipient's entries stay secret.	`trad_key_leak`	proved	one_trad	checked
C08.8	One broken component, the ML-KEM decapsulation key, does not disclose the CEK/CAEK of the recipient's entries while the traditional component of that key remains intact.	`mlkem_key_leak`	proved	one_mlkem	checked
C08.8	One broken component, the traditional private key, does not disclose the CEK/CAEK of the recipient's entries while the ML-KEM key of that same key remains intact.	`trad_key_leak`	proved	one_trad	checked
C08.8	When both component private keys of the composite key are disclosed, the CEK/CAEK of an entry protected with that key is recovered.	`composite_key_leak`	attack (holds_in_source)	ctl_cek (the control of cek_secret)	
A08.9	The per-suite Label is an input to the combiner: two entries whose KEK coincides carry the same suite identifier and the same recipient, so a ciphertext made under one suite never yields another suite's KEK (control: two unconstrained entries do not share a KEK).	`network`	proved	kek_inj	checked
A08.9	The traditional ciphertext and, through ML-KEM decapsulation, the ML-KEM ciphertext carried in the entry are the ones the composite secret is computed from: an entry with replaced ciphertexts does not yield the accepted KEK (control: without the condition that the two events concern one entry, other ciphertexts are accepted).	`network`	proved	ct_bound	checked
A08.9	The per-suite Label is an input to the combiner: two entries whose KEK coincides carry the same suite identifier and the same recipient, so a ciphertext made under one suite never yields another suite's KEK.	`network`	proved	kek_inj	checked
A08.9	The traditional ciphertext and, through ML-KEM decapsulation, the ML-KEM ciphertext carried in the entry are the ones the composite secret is computed from: an entry with replaced ciphertexts does not yield the accepted KEK.	`network`	proved	ct_bound	checked
D01.10	The recipient cannot establish which party originated an accepted entry: any party holding the recipient's certificate can make an entry the recipient accepts.	`network`	attack (holds_in_source)	forge	
X01	No two KEMRecipientInfo entries of the originator share a KEK or a CEK: entries carrying the same composite shared secret are the same entry, so per-recipient encapsulation randomness and the shared secret are never reused (control: two unconstrained entries do not share a KEK or a CEK).	`network`	proved	kek_inj	checked
X01	No two KEMRecipientInfo entries of the originator share a KEK or a CEK: entries carrying the same composite shared secret are the same entry, so per-recipient encapsulation randomness and the shared secret are never reused.	`network`	proved	kek_inj	checked

target	claim	attacker	answer	query	control
X02	The KEK and CEK of one message, disclosed to the attacker, do not disclose the content of any other message; the content of another message is reached only when the recipient's composite private key is disclosed. The attacker holds one message's KEK and CEK beyond the network.	`network`	proved	confine	checked
X03	The content and the CEK of a session that has already completed are recovered once the recipient's composite private key is disclosed afterwards: there is no forward secrecy.	`composite_key_leak`	attack (holds_in_source)	ctl_cek (the control of cek_secret)	

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-20261114T233225, 2026-11-14. Status: finished. Model: deepseek-v4-1-flash-261017. Steps: 46, prover 4 s, wall 16 min, cost 0.137 USD.

ProVerif: 71 proofs run, 71 decided, 0 without an answer, 3 s, 43 answers recorded.

What was analysed

Protocol	CMS composite KEM encryption profile
Version	draft-ietf-lamps-cms-composite-kem-04
Analysed	(a) A combination of ML-KEM + traditional weights explicitly registered in the fixed draft; Single recipient AuthEnvelopedData; Receiving public key is certified, independent mass randomity and standard grouper input/domain security is maintained; The algorithmic computational nature is not included in the symbol conclusion. Packages where the script does not exist shall not be selected.
Not analysed	Symbolic models do not support the calculation of algorithms as safe or as resistant to quantum security; The trajectory of the attack is established only under clear assumptions and is not directly equivalent to a normative loophole. Each nature must be subject to accessibility/proportional controls.
Source text	124 numbered passages (22283 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	2	Use of Composite ML-KEM in the CMS
main	2.1	RecipientInfo Conventions
main	2.2	Underlying Components
main	2.2.1	Use of the HKDF-Based Key Derivation Function
main	2.3	Certificate Conventions
main	2.4	SMIME Capabilities Attribute Conventions
main	3	Identifiers
main	4	Security Considerations
rfc9629	2	KEM Processing Overview
rfc9629	3	KEM Recipient Information
rfc9629	5	Key Derivation

The process of the protocol as the source step listed it in the documents: a combination of ML-KEM + traditional weights explicitly registered in the fixed draft; Single recipient AuthEnvelopedData; Receiving public key is certified, independent mass randomity and standard grouper input/domain security is maintained; The algorithmic computational nature is not included in the symbol conclusion. Packages where the script does not exist shall not be selected.

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
K01	Keys and private key kept secret	The recipient's Composite ML-KEM private key, the pairwise KEK, the CEK/CAEK (which in AuthEnvelopedData also carries the message authentication) kept from an attacker who has only the network; and the stated consequence that disclosure of the private key compromises every message protected with it.	main s0067 makes implementations protect exactly these values and states the disclosure consequences; rfc9629 section 7 says the same and adds that KEK/CEK/CAEK compromise is limited to one message.	main:490, main:494, rfc9629:568, s0067, s0078, s0079, s0109, s0123
K02	Content confidentiality under the CAEK	The content and the CEK/CAEK protecting it, kept from an attacker who holds neither the recipient's decapsulation key nor the CEK/CAEK.	s0002/s0005: the composite KEM transfers the CEK/CAEK from originator to recipient; s0078 and s0087: the CEK is generated by the originator and every recipient decrypts the content with it; s0067 lists the CEK/CAEK among the values to protect.	main:490, s0002, s0005, s0067, s0078, s0087
K04	KEK separation across derivation contexts	The pairwise KEK: KEKs derived for different wrap algorithms, different kekLength, different ukm or different composite key pairs must be different, so a KEK of one context cannot serve as the KEK of another.	s0116/s0117 build the KDF info input from the KEMRecipientInfo elements (wrap, kekLength, ukm); s0124 requires that all KDF inputs influence the output; s0034 requires L to be confirmed consistent with the wrap key size.	rfc9629:378, s0014, s0034, s0116, s0117, s0119, s0120, s0121, s0124
A03	Recipient agreement on KEK and identity	The recipient: when it accepts a delivered KEMRecipientInfo its KEK, CEK and content are the originator's for that entry, and the recipient identity in rid is the one whose certificate was used.	The documents state no explicit authentication goal; the requirement follows from the purpose, transfer of the CEK to the intended recipient (s0002, s0005, s0078 with s0084/s0089), and from rid naming the recipient's certificate and key (s0102).	s0002, s0005, s0078, s0084, s0089, s0102
A08	KEMRecipientInfo fields bound into the KEK	The kem, kdf and wrap identifiers, kekLength and ukm that the recipient accepts with a KEMRecipientInfo, and that these exact values are the ones bound into the KEK derivation.	s0106: implementations MUST confirm kekLength is consistent with the wrap algorithm; s0116/s0117: the info input of the KDF is built from these fields and s0124 requires all inputs to influence the output; s0014/s0020 mandate HKDF-SHA256 and AES-Wrap-256; s0034 requires confirming L; s0019 leaves ukm free.	main:296, main:368, rfc9629:378, s0012, s0014, s0019, s0020, s0034, s0106, s0116, s0117, s0124
D01	Content integrity without origin authentication	The authenticated content the recipient accepts: not modified by an attacker on the network, while the recipient gets no assurance about which party originated it.	rfc9629 section 7 states that KEM algorithms do not provide data origin authentication, so the content is delivered with integrity from an unknown source; main s0067 states that key disclosure could allow modification of the authenticated content.	rfc9629:563, s0002, s0067
C08	Hybrid security with one component broken	The composite shared secret, the resulting KEK and the content, when the attacker holds the private key of exactly one composite component and the other component is intact.	main s0066 makes the security considerations of the composite KEM draft apply, and that draft states that the combined secret stays secret while one component is broken; main s0075 warns about key-pair reuse.	s0066, s0075

ID	Name	About	What the documents say	Where
C08.8	CEK secrecy under partial or full leakage	The CEK/CAEK and the KEK when one or both private keys of the composite components are disclosed; the user question asks for the single-leak and the double-leak case separately.	Same basis as C08: main s0066 defers to the composite KEM draft, whose combiner statement covers one broken component; main s0067 and rfc9629 section 7 state the full private-key disclosure consequence.	s0066, s0067, s0075
A08.9	Combiner binds label, ciphertexts, keys	The composite shared secret produced by the KEM combiner: it must bind the suite Label, the traditional ciphertext and traditional public key, and through ML-KEM decapsulation the ML-KEM ciphertext and public key.	s0012 requires kem to hold one of the Composite ML-KEM OIDs of section 3 and s0044..s0059 register a distinct OID per suite; the combiner itself is specified in the referenced composite KEM draft (s0041), which main s0066 makes applicable.	s0012, s0041, s0048, s0053, s0066
D01.10	No sender authentication from AEAD	The originator's identity, which the recipient cannot establish from an accepted AuthEnvelopedData even though the content is intact.	rfc9629 section 7 line 563: KEM algorithms do not provide data origin authentication, so the content is delivered with integrity from an unknown source.	rfc9629:563, s0002, s0067
X01	Per-recipient encapsulation freshness	The originator's per-recipient encapsulation, its fresh randomness and the composite shared secret: nothing of one entry may be reused for another.	rfc9629 section 7 (lines 574-587) requires a fresh KEM shared secret, including any random value the KEM uses, for each recipient, and forbids reuse for any other purpose; main s0069/s0072 require independent component randomness.	rfc9629:575
X02	Leakage confined to one message	The content of other messages, when one message's KEK or CEK/CAEK reaches the attacker.	rfc9629 section 7 (lines 568-572): compromise of the KEK, the CEK or the CAEK may result in disclosure of the encrypted content of a single message, while compromise of the KEM private key may disclose all contents protected with it.	rfc9629:569
X03	Forward secrecy of past messages	The content and the CEK/CAEK of a completed session, against a disclosure of the recipient's composite private key that happens after the session ended.	main s0067 states that disclosure of the Composite ML-KEM private key could result in the compromise of all messages protected with that key, which is the absence of this property; rfc9629 section 7 lines 568-570 says the same.	main:494

Added from the documents by the analyst: X01, X02, X03.

The security statements of the documents, paragraph by paragraph

The host listed the 84 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: 48. Not taken: 36. Which paragraphs are listed follows from the layout of the standard; where each went is the analyst's judgement.

Paragraphs	Where	Taken up as	Why not
P01	draft-ietf-lamps-pq-composite-kem:2531-2534	C08, K01, `composite_key_leak`	
P02 to P06	draft-ietf-lamps-pq-composite-kem:2538-2578	C08	

Paragraphs	Where	Taken up as	Why not
P07 to P09	draft-ietf-lamps-pq-composite-kem:2582-2586	A08.9	
P10	draft-ietf-lamps-pq-composite-kem:2588-2596	C08, A08.9	
P11	draft-ietf-lamps-pq-composite-kem:2598-2602	A08.9	
P12 to P18	draft-ietf-lamps-pq-composite-kem:2604-2628	A08.9	
P19	draft-ietf-lamps-pq-composite-kem:2632-2633	The prover cannot be asked	It introduces a glossary of computational notions; the introduction itself states no property of the protocol.
P20	draft-ietf-lamps-pq-composite-kem:2635-2649	A08.9	
P21	draft-ietf-lamps-pq-composite-kem:2653-2655	C08	
P22	draft-ietf-lamps-pq-composite-kem:2657-2662	C08	
P23 to P26	draft-ietf-lamps-pq-composite-kem:2666-2692	A08.9	
P27	draft-ietf-lamps-pq-composite-kem:2694-2697	A08.9	
P28 to P31	draft-ietf-lamps-pq-composite-kem:2701-2728	The prover cannot be asked	Ciphertext second pre-image resistance is a computational property about finding two ciphertexts that decapsulate to the same secret; a symbolic model makes constructors injective and cannot be asked it.
P32	draft-ietf-lamps-pq-composite-kem:2731-2735	A08.9	
P33	draft-ietf-lamps-pq-composite-kem:2737-2741	A08.9	
P34	draft-ietf-lamps-pq-composite-kem:2745-2749	No goal stated	It warns against applying this combiner to other component algorithms: a design restriction, not a property of this exchange.
P35	draft-ietf-lamps-pq-composite-kem:2753-2757	C08	
P36	draft-ietf-lamps-pq-composite-kem:2759-2764	No goal stated	It compares single-algorithm key reuse with the hybrid setting; it states no property of this exchange.

Paragraphs	Where	Taken up as	Why not
P37 to P40	draft-ietf-lamps-pq-composite-kem:2766-2791	C08, A08.9	
P41	draft-ietf-lamps-pq-composite-kem:2795-2806	Outside the scope	Guidance to certification authorities about revocation checks on component keys; certificate management is outside the task.
P42 to P43	draft-ietf-lamps-pq-composite-kem:2810-2827	No goal stated	Policy for deprecating algorithms in composite certificates; it states no property of the message flow that a prover could be asked.
P44	main:486-488	C08, K01, D01	
P45	main:490-501	K01, K02, D01, `composite_key_leak`, `kek_leak`	
P46	main:503-504	No goal stated	A pointer to the key-management guidance of SP 800-57, stating nothing itself.
P47 to P50	main:506-532	The prover cannot be asked	Weak pseudorandom generators and the information encapsulation randomness gives about other outputs are probability and generator-quality questions; a symbolic prover has no probabilistic attacker.
P51	main:534-536	C08, A08.9	
P52	main:538-545	The prover cannot be asked	Side channels through timing or intermediate calculations are outside what a symbolic prover can be asked.
P53	main:547-550	A08.9	
P54	rfc5083:308-310	K02, D01	
P55	rfc5083:312-317	The prover cannot be asked	A statement about the strength of counter-mode encryption compared with CBC; a computational cipher property.
P56	rfc5083:325-328	K02	
P57	rfc5083:330-337	X02	
P58	rfc5083:339-348	The prover cannot be asked	Random generation of CAEKs, padding and nonces and the quality of the generator are not properties a symbolic prover can be asked.
P59	rfc5083:350-355	Outside the scope	The message-digest attribute of signed attributes and content tracking: signed attributes are not part of the modelled profile.
P60	rfc5083:357-369	Outside the scope	Unsolicited encrypted traffic and plaintext analysis by gateways: an operational concern outside the modelled exchange.
P61	rfc5652:2837-2839	No goal stated	It describes what CMS provides in general; no property of this profile.
P62	rfc5652:2841-2842	Outside the scope	Protection of the signer's private key and masquerade of signed data: the signature part of CMS is outside this encryption profile.
P63	rfc5652:2844-2849	K01, K02, `composite_key_leak`, `kek_leak`	
P64	rfc5652:2851-2855	D01, K01	
P65	rfc5652:2864-2872	D01, D01.10	
P66	rfc5652:2874-2877	D01.10	
P67	rfc5652:2879-2889	The prover cannot be asked	Random generation of CEKs, MAC keys, IVs and padding; generator quality is outside a symbolic model.

Paragraphs	Where	Taken up as	Why not
P68	rfc5652:2891-2903	The prover cannot be asked	The effective security is that of the weaker of the two algorithms: a statement about cryptographic strength, not about traces.
P69	rfc5652:2905-2911	No goal stated	Algorithms weaken with time and implementations should be modular: advice, not a property of the exchange.
P70	rfc5652:2917-2927	Outside the scope	The countersignature unsigned attribute; CMS attributes are outside this encryption profile.
P71	rfc5869:355-360	No goal stated	An introduction to HKDF's own security considerations that refers to another document; it states no property of this protocol.
P72	rfc5869:362-371	The prover cannot be asked	The strength of the underlying hash and the modelling choices behind HKDF are computational assumptions a symbolic prover cannot be asked about.
P73	rfc9629:542-543	K01, K02, D01	
P74	rfc9629:545-554	The prover cannot be asked	IND-CCA2 security and the Fujisaki-Okamoto transform are computational properties; the symbolic model idealises the KEM instead.
P75	rfc9629:556-556	The prover cannot be asked	The KDF should offer at least the security level of the KEM: a statement about strengths.
P76	rfc9629:558-561	The prover cannot be asked	The choice and size of the key-encryption algorithm against the KEM's level: a statement about strengths.
P77	rfc9629:563-566	D01, D01.10	
P78	rfc9629:568-572	K01, K02, X02, `composite_key_leak`, `kek_leak`	
P79	rfc9629:574-587	X01	
P80	rfc9629:589-598	The prover cannot be asked	Random generation of CEKs, CAEKs and MAC keys and generator quality are not askable symbolically.
P81 to P83	rfc9629:600-620	The prover cannot be asked	The effective security of a key-encryption, content-encryption or MAC algorithm pairing: statements about cryptographic strength.
P84	rfc9629:622-629	No goal stated	Algorithms weaken over time and implementations should be modular: advice to implementers.

The catalogue of properties, entry by entry

The properties were drafted from a catalogue of security and privacy properties. The draft left out 35 of its entries, and the analyst went through each of them: examined here as a property, or left out with the reason. Examined: 12. Left out: 23. 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
K03	Indistinguishability of confidentiality/secretity	K01, K02	
A01	(Aliveness)	A03	
A02	Weak Consistency Certification	A03, D01.10	
A04	Single-shot Consistency Authentication	D01.10	
A05	Secret Key Authentication	K01, K02	

Entry	Property of the catalogue	Examined as	Why not
A06	Visible Key Confirmation	Not applicable	The profile has no key-confirmation message: the originator never receives anything back, so nothing could confirm that the recipient holds the key.
A07	Identity - Certificate - Key binding	A03, A08	
R01	Certification/challenging novelty	Not applicable	The profile has no challenge, counter, timestamp or ticket; nothing is authenticated by freshness.
R02	Message Resume	D01.10	
R03	Operations at most once	Not applicable	The profile performs no business operation that could be charged, consumed or allocated twice.
R04	Message acceptance order/window security	Not applicable	There is no sequence number, counter or acceptance window in this profile.
R05	Anti-degrading/security strategy maintained	A08, A08.9	
C01	Forward confidentiality (FS)	X03	
C02	Anti-key disclosure fake (KCI)	Not applicable	The originator holds no long-term credential at all, so there is no peer key whose disclosure could be used to impersonate it.
C03	Known key security/dissemination quarantine	X02	
C04	Temporary secret/partial state leaks resilience	C08	
C05	Security restored after leak (PCS)	Not applicable	There is no ratchet, no rekeying and no long-lived session: each message is a fresh one-shot encapsulation.
C06	Enhanced authentication after disclosure of a certificate of registration	Not applicable	No password and no enhanced authentication mechanism exists in this profile.
C07	Anti-offline dictionary attack	Not applicable	No password or other low-entropy secret is used, so there is no offline dictionary attack surface.
Z01	Authorization to execute correctness	Not applicable	The profile carries no authorization decision: the recipient simply decrypts what it can.
Z02	Delegation does not extend	Not applicable	There is no delegation or proxy mechanism.
Z03	Authorisation and identity/meeting/operation binding	Not applicable	There is no authorization token to bind to a user, service or operation.
Z04	Avoidance and expiry	Outside the scope	Revocation and certificate validity are certificate-management concerns that the task leaves out; the draft discusses them only as CA guidance.
Z05	Cross Tenant/Interservice Safety	Not applicable	There is no multi-tenant or cross-service structure in the profile.
S01	Encryption uniqueness	X01	
S02	Re-transmit/key weight installation security	Not applicable	No key is ever installed or reinstalled: every message is independent.
S03	Anti-roll and state monotony	Not applicable	The profile keeps no versioned or cached state that could be rolled back.
S04	Consistency in security context across phases	Not applicable	The profile has one phase only: one originator, one entry, one delivery.
S05	Termination and old state lapses	Not applicable	There is no session termination; nothing outlives the message.

Entry	Property of the catalogue	Examined as	Why not
P01	Identity anonymity	Not applicable	The recipient is named in rid and by its certificate in the clear; the profile has no anonymous mode.
P02	Interconnectedness of sessions	Not applicable	Messages carry the recipient's identifier and are not designed to be unlinkable; nothing in the profile aims at it.
P03	User/evidence Existence Hide	Not applicable	There is no lookup or discovery exchange whose answer could enumerate users or credentials.
V01	Conditional progress/ error recovery	The prover cannot be asked	Progress and error recovery are liveness questions, which a trace-based prover cannot be asked.
V02	No unexpected death locks.	The prover cannot be asked	Deadlock freedom is not a property ProVerif can be asked; it cannot state that a run always completes.
V03	Abuse of restricted resources/anti-magnification	The prover cannot be asked	Resource amplification needs a cost model; a symbolic prover counts events and nothing else.

The parties and the protected values, one by one

The host listed the 18 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: 7. Without: 11. 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	originator	party	Nothing gained	The originator holds no long-term secret: an attacker in its place can already send the recipient any content over the network, which is what D01.10 states.
S02	recipient	party	`composite_key_leak`	
S03	Composite ML-KEM private key	value	`composite_key_leak`	
S04	pairwise KEK	value	`kek_leak`	
S05	CEK/CAEK	value	`kek_leak`	
S06	message-authentication key	value	`kek_leak`	
S07	content	value	Nothing gained	Knowing the plaintext of one message yields no key and nothing about any other message; the properties of the targets remain askable and unaffected.
S08	context fields	value	Nothing gained	wrap, kekLength and ukm travel in the KEMRecipientInfo itself and are public to the network attacker already.
S09	pairwise shared secret	value	`kek_leak`	
S10	recipient identity	value	Nothing gained	The recipient identity is carried in rid and in a public certificate; the network attacker has it already.
S11	kem OID	value	Nothing gained	The KEM OID is a public algorithm identifier carried in the message.
S12	kdf OID	value	Nothing gained	The KDF OID is a public algorithm identifier carried in the message.
S13	wrap OID	value	Nothing gained	The wrap OID is a public algorithm identifier carried in the message.
S14	kekLength	value	Nothing gained	kekLength is a public integer carried in the message.
S15	ukm	value	Nothing gained	The ukm is public by design: s0019 places no requirement on it and it is transmitted in the clear.

ID	Party or value	Kind	Attacker	Why none
S16	authenticated content	value	Nothing gained	The authenticated content is the message payload; holding it gives no key and nothing about other messages.
S17	CAEK	value	`kek_leak`	
S18	recipient public key	value	Nothing gained	The recipient's composite public key is published in its certificate and used by the originator; the network attacker has it already.

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		
`composite_key_leak`	The recipient's whole Composite ML-KEM private key, i.e. both component private keys, disclosed to the attacker at any time during or after the protected session.	main s0067 states that disclosure of the Composite ML-KEM private key could result in the compromise of all messages protected with that key, so the property has to be checked under that disclosure.	main:494, s0067
`mlkem_key_leak`	Only the ML-KEM decapsulation key of the recipient, the traditional component's private key still intact.	main s0066 makes the composite KEM draft's security considerations apply, and that draft analyses the case where one component algorithm is completely broken; user question 1 asks for this case.	s0066, s0075
`trad_key_leak`	Only the private key of the traditional component (X25519 or ECDH), the ML-KEM decapsulation key still intact.	The symmetric case of the one-broken-component analysis that main s0066 makes applicable; user question 1 asks for it.	s0066, s0075
`kek_leak`	The key-encryption key, the content-encryption key and the content-authenticated-encryption key of one message of a session, obtained by the attacker during that session (for example by reading them from a party's memory).	rfc9629 section 7 lists the KEK among the values implementations must protect and says its compromise may disclose the content of a single message; main s0067 says disclosure of the key-encryption key, the content-encryption key or the CAEK could compromise the associated encrypted content.	rfc9629:569, main:496, s0067

ID	`network`	`composite_key_leak`	`mlkem_key_leak`	`trad_key_leak`	`kek_leak`
K01	Holds	Fails			
K02	Holds				
K04	Holds				
A03	Holds				
A08	Holds				
D01	Fails				
C08			Holds	Holds	
C08.8		Fails	Holds	Holds	
A08.9	Holds				

ID	network	composite_key_leak	mlkem_key_leak	trad_key_leak	kek_leak
D01.10	Fails				
X01	Holds				
X02	Holds				
X03		Fails			

Notes on the prover and the model

What the analyst named for targets:

step	targets	features	reason
31	K01, K02, K04, A03, A08, A08.9, D01, D01.10, C08, C08.8, X01, X02, X03	plain_primitives, many_sessions, long_flow	Every answer rests on ideal primitives: the composite KEM is two abstract KEMs whose shared secrets are functions of the public key and fresh coins, the combiner is a free injective constructor, and HKDF, AES key wrap and the AEAD are ideal constructors with their single inverse law. No exclusive-or and no Diffie-Hellman algebra is used, so the traditional component is treated as an opaque KEM and the model gives the attacker less than a real X25519 or ECDH implementation. The properties are asked over any number of originator and recipient sessions of two suites, with a two-phase run for the private-key disclosure.

The analyst's closing note

The CMS composite KEM profile was modelled from the source units and all thirteen targets answered. The model has one originator and two recipients of two different composite suites (ML-KEM768-X25519, ML-KEM1024-ECDH-P384); the composite KEM is two abstract components with their own coins, the combiner follows the referenced draft (both component secrets, tradCT, tradPK, per-suite Label), the key-encryption key comes from HKDF over wrap, kekLength and ukm, and the content key is wrapped and used under an ideal content encryption. A second phase discloses the composite private key, one session publishes its own key-encryption and content keys, and the two single-component leaks are placed on the two different key pairs so each one-component claim is tested with the other component intact. Proved, each with a control that fails: the key-encryption key, the content key and the content are secret from the network attacker; the recipient ends with the originator's keys, content, identity and suite for a delivered entry; the ukm and the ciphertexts are the values bound into the key-encryption key; the hybrid property holds with one component broken in either direction; entries never share a key; and a leaked message key reaches only that message. Attacks that hold in the source: the recipient accepts an entry no originator sent (recorded, RFC 9629 section 7), a completed session is opened by a later private-key disclosure (recorded, main section 4), and losing both component keys loses every message protected with the key pair. No new finding. What blocks more: the KEM components, the combiner, the key derivation and the cipher strengths are idealised and the traditional component carries no Diffie-Hellman algebra, so carrying the results to the algorithms needs Tamarin+ Agent; no value changes across messages here, so no bounded model or state review applies; and the wrap algorithm and key length are profile constants, so only the ukm was exercised as a context field.