

Beyond Mosca’s Heuristic: Supported Is Not Protected in Post-Quantum Migration

Abdoul Ahad Fall 

Université Paris 8 (Vincennes–Saint-Denis), Saint-Denis, France
CLICKNCRYPT, 7 rue de la Croix Martre, 91120 Palaiseau, France
lahad@clickncrypt.fr
ORCID: 0009-0001-0111-3053

Abstract—Post-quantum (PQ) migration is usually tracked per asset, with Mosca’s inequality and a share of “migrated” systems. For a flow of sensitive data, the relevant questions are different: is every session that carries it keyed by a quantum-safe secret, what evidence supports that claim, and what can an assessor establish from outside? We propose an assessment method whose unit is the data path and whose verdicts are tied to evidence. Key lineage is an algebra on break dates: a derived key falls when all its inputs fall, and a key falls when any of its copies falls. This covers TLS 1.3 resumption, key transport and keys wrapped several times. A verdict is a pair of strong-Kleene evaluations, one taking claims at face value and one admitting only claims that meet an evidence policy; the reachable pairs form four ordered labels. We characterize what an external probe can establish. A probe of the server decides a hop only if the server enforces PQ or cannot negotiate it; enforcement must be shown by a negative test paired with a positive control; a server that resumes sessions without a fresh key exchange leaves the layer undecided; and behind a load balancer the confidence of a negative test follows a sampling bound only under an independence assumption that a default balancer violates. We implement the probe and validate its inference rules on 22 configurations of five server stacks built on three TLS libraries, including confounders and a heterogeneous load balancer. The same nominal configuration yields different states across implementations, so declared configuration cannot replace runtime evidence. A protocol for an internet-scale enforcement study accompanies the paper.

Index Terms—post-quantum cryptography, migration readiness, TLS 1.3, hybrid key exchange, key lineage, active measurement, Mosca’s inequality, financial infrastructures

I. INTRODUCTION

ML-KEM is standardized [1], the design of hybrid key exchange in TLS 1.3 is described in RFC 9954 [2], its ML-KEM hybrid groups are specified in RFC 10024 [3], and hybrid key exchange is documented for SSH [4], and adoption is fast where both ends are controlled by a few vendors: Cloudflare reports that the share of human HTTPS traffic it serves with PQ key agreement grew from 29% to 52% during 2025 [5]. Public authorities now publish target dates. NIST IR 8547, at the time of writing an initial public draft, proposes to deprecate quantum-vulnerable public-key algorithms at the 112-bit level after 2030 and to disallow them after 2035 [6]. The EU roadmap sets milestones in 2026, 2030 and 2035 [7], and the G7 Cyber Expert Group proposes 2035 for the financial sector, with critical systems around 2030–2032, in a roadmap that it presents as non-binding [8].

PCI DSS 4.0 already requires an annually reviewed inventory of cipher suites and protocols [9].

Organizations and the supervisors of shared infrastructures must answer an operational question: *is this flow protected against a recorder who will later own a cryptographically relevant quantum computer (CRQC), what supports the claim, and when will it hold?* Which strategy to follow (hybrid or pure PQ key establishment, TLS 1.3 hybrids or KEMTLS, sector-specific timelines) is the subject of a recent systematization of hybrid migration strategies [10]. This paper takes the strategy as given and asks how to verify, flow by flow, that its deployment actually protects data. Per-asset use of Mosca’s inequality [11], [12] misses five facts. (i) Data follows a path, and one unprotected hop exposes it. (ii) Confidentiality, authentication and long-lived signatures have different deadlines; this paper formalizes confidentiality and records authentication as a separate observed state. (iii) Negotiating a PQ group is not refusing classical key exchange, and a key can be inherited or copied. (iv) Completion depends on counterparties. (v) Assessors do not see inside other organizations.

Contributions:

- **Key lineage (Sec. V).** A break-date algebra, relative to what the adversary can record, in which derivation is conjunctive for the attacker (max) and copying is disjunctive (min). It classifies pre-shared keys, resumed sessions, transported keys and multiply-wrapped keys correctly.
- **Verdicts with evidence (Sec. VI).** Hop-AND/Layer-OR verdicts over explicit adversary recording classes. *Hop-AND* means that a path is protected only if *every* hop it crosses is protected (a conjunction over hops); *Layer-OR* means that a hop is protected as soon as *at least one* of the protection layers covering it (per-hop TLS, an end-to-end envelope, encryption at rest) is protected (a disjunction over layers). Verdicts are computed as a pair of Kleene evaluations (face value, evidence policy). Evidence has a level, a negative test with a positive control, a scope and a date. Exposure is decided per hop with per-family CRQC dates.
- **What a probe can establish (Sec. VII).** An identifiability result for server-side probing that accounts for session resumption, the conditions under which a refusal counts as enforcement, and a sampling bound for load-balanced

services together with its limits.

- **Implementation and validation (Sec. VIII).** `pqprobe`, validated on OpenSSL, nginx, HAProxy, Go and rustls (three TLS libraries), with confounders and a load-balancer experiment, and a protocol for an internet-scale study.
- **Composition (Appendix A).** Interfaces that compose dates and verdicts across organizations, including layers that cross organizations, with an explicit account of what they disclose.

Exposure-aware ordering of migration tasks is out of scope. The formal results are elementary. Their purpose is to make the assessment rules precise and checkable; they relate to series-parallel structure functions and bottleneck paths, which we cite as such.

II. BACKGROUND

TLS 1.3 negotiation: A client offers groups and key shares; the server selects a group, possibly after a `HelloRetryRequest`. TLS 1.3 is now specified by RFC 9846, which obsoletes RFC 8446 without changing the protocol version [13]. Whose preference wins is an implementation choice. Because the transcript is authenticated, an active adversary cannot change the group chosen by two PQ-capable peers before the CRQC exists. The residual risk of a server that still accepts classical groups is its *peer population*: any classical-only or misconfigured peer obtains a classical session. In a recent preprint, Han argues, with experiments on seven configurations of two protocols including SSH, that delivered PQ protection is a relation between an artifact and its peer population, and that common audit surfaces miss its loss [14]. Sessions can be resumed from a pre-shared key with a fresh key exchange (`psk_dhe_ke`) or without one (`psk_ke`); in the latter case the new keys descend from the earlier handshake [13]. OpenSSL disables `psk_ke` by default.

Hybrids and combiners: A hybrid KEM is secure if either component is, provided the combiner is robust [15]–[17]; NIST SP 800-227 gives recommendations for KEMs and their combination [18].

Profiles: ANSSI and BSI recommend hybrid key establishment during the transition [19]–[21], whereas CNSA 2.0 targets PQ-only algorithms (ML-KEM-1024, ML-DSA-87) for national security systems [22]. Security and compliance are therefore different predicates. In payments, BIS Project Leap tested PQ signatures on liquidity transfers in an operational payment system, alongside the classical ones, and reported that such hybridization had not been anticipated in the system’s original cryptographic design [23].

Mosca and z : Mosca’s inequality $x + y > z$ turns the CRQC date z into a deadline [11]. Expert elicitations give intervals that refer to a single cryptanalytic target, RSA-2048 [24], and do not provide dates per family. We nevertheless allow one date z_ϕ per algorithm family ϕ , as a modelling choice for assessors who expect different timelines; for instance, resource estimates for elliptic-curve discrete logarithms

at 256 bits need fewer logical qubits than for RSA-2048 [25]. The dates used in our examples are hypothetical.

III. RELATED WORK

Migration strategies, frameworks and inventories: A systematization of knowledge on hybrid strategies for the PQ transition [10] reviews hybrid key establishment in TLS 1.3, QUIC and KEMTLS, the regulatory and standardization landscape, and sector-specific migration roadmaps, and identifies open gaps, among them organization-specific migration frameworks and the evaluation of cryptographic discovery tools. It addresses which migration strategy to adopt; the present work is complementary and addresses how to verify its outcome on a flow. Institutional frameworks [26]–[29] structure discovery, prioritization and execution. CBOMs [30], [31], the XIPHER methodology [32], [33] and infrastructure digital twins [34] provide assets, dependencies and evidence records. None defines when a flow is protected or what evidence demonstrates it.

Measurement: Loizou and Ghadafi measure PQ group support across 4,665 UK organizations and observe no PQ certificate signatures [35]. Delgado separates passive, active-probe, certificate and registry evidence, shows that active probing reveals hybrid capability that single sessions hide, and measures capability on 1,000 public targets [36]; its stress scenarios already cover mTLS, PSK resumption and `HelloRetryRequest`, and it treats unknown outcomes as first-class. Both measure what servers *support*. Han argues that PQ delivery cannot be gated by one peer [14]. Wickramasinghe et al. measure PQ key exchange longitudinally on one million domains over three rounds and find adoption driven by a few infrastructure providers rather than by policy timelines [37]; like the other studies, they measure what is negotiated, not what is refused. Cloudflare publishes adoption statistics [5].

Scanners and test frameworks: `testssl.sh` and `SSLyze` enumerate protocol versions, ciphers and groups, including ML-KEM hybrids [38], [39]; `tlsfuzzer` and `TLS-Attacker` drive arbitrary handshakes and negative test cases [40], [41]; `ZMap` and `zgrab2` support internet-wide scans [42]. Offering only classical groups and observing a refusal is therefore not new as a technique. What is new here is not the observation of resumption, `HelloRetryRequest` or unknowns, which Delgado’s framework also covers [36], but the use of refusal as a *decision criterion*, tied to key lineage: when a refusal demonstrates enforcement, what it says about a hop whose client is unknown, which resumption behaviour voids it, and how much confidence it carries behind a load balancer.

Risk models and formal tools: Graph and layer-based risk scores [43]–[47] aggregate per asset or connection; weighted aggregation lets a protected hop compensate an unprotected one, which harvest-now-decrypt-later does not allow. Loebenberg et al. formalize migration [48]. Our verdict is a monotone series-parallel structure function as in reliability and fault trees [49], [50], with the AND/OR form of attack trees [51]–[53], and our dates are bottleneck values [54].

IV. THREAT MODEL AND ASSUMPTIONS

A data item d needs confidentiality for x_d years and follows a path of hops between components of one or several organizations. A *recorder* records the hops of a set of classes $\mathcal{A} \subseteq \{\text{internet, interconnect, internal, storage}\}$. From date z_ϕ an adversary can recover secrets protected only by classical algorithms of family $\phi \in \{\text{ECC, RSA, ECDH}\}$ (elliptic-curve algorithms including ECDH, RSA, and finite-field Diffie–Hellman). Before z_ϕ classical authentication holds. A plaintext node keeps d confidential iff it is trusted. Side channels and implementation flaws are out of scope.

Assumption 1. (A1) A key derived by a KDF from several inputs is pseudorandom if one input is. This requires a robust combiner, which is not automatic [15]; we assume the deployed designs, such as the TLS 1.3 hybrid key schedule [2], have been shown robust [16]. (A2) A layer whose session key is pseudorandom protects its payload against a quantum adversary; we assume 256-bit symmetric primitives, which the probe records but does not grade. (A3) A classical exchange or transport of family ϕ is recoverable from its transcript from z_ϕ on. (A4) A key wrapped or transported under a key W is recoverable once W is, provided the adversary recorded the wrapped copy.

These assumptions define a model; the statements below are made in this model and are not game-based reductions.

V. KEY LINEAGE

Definition 1 (Lineage and break date). A key K has an *origin* and a set of *copies*. The origin is fresh randomness, a PQ or hybrid KEM secret, an ephemeral classical secret of family ϕ (an (EC)DH exchange or a key transport), a static classical key pair of family ϕ , or a KDF of input keys $I_1, \dots, I_r$. A copy is a key W under which K was transported, wrapped or stored, or an offline ceremony (for instance key components entered by custodians). Each ephemeral classical origin and each copy has a set of *carriers*: the places where its key-exchange transcript or its wrapped value (the ciphertext) appears, each given by a recording class (internet, interconnect, internal, storage) and by the key of the layer, if any, that protects it there. A static key pair has no carrier: its public key is public, so its private key is recoverable at z_ϕ whatever is recorded.

The *break date* $\beta_{\mathcal{A}}(K)$ of K is the earliest date from which an adversary that records the classes in $\mathcal{A}$ and owns a CRQC can recover K ; $\beta_{\mathcal{A}}(K) = \infty$ means that K can never be recovered by this adversary. Let the *carrier date* $\gamma_{\mathcal{A}}(S)$ of a set S of carriers be the earliest date at which the recorder both holds and can read the value: the minimum, over carriers of S whose class is in $\mathcal{A}$, of the break date of the protecting key ($-\infty$ for a carrier in the clear), and ∞ if no carrier is recorded. The break date is the earlier of two dates: the *origin break date* $\beta_{o,\mathcal{A}}(K)$, the earliest date at which K can be recovered by attacking the way it was created; and, for each copy under W

with carriers S_W , the date at which both W and the wrapped value are available:

$$\beta_{\mathcal{A}}(K) = \min\left(\beta_{o,\mathcal{A}}(K), \min_{W \in \text{copies}(K)} \max(\beta_{\mathcal{A}}(W), \gamma_{\mathcal{A}}(S_W))\right).$$

The origin break date is ∞ for fresh randomness and PQ secrets; z_ϕ for a static key pair of family ϕ ; $\max(z_\phi, \gamma_{\mathcal{A}}(S))$ for an ephemeral classical secret with transcript carriers S ; and $\max_i \beta_{\mathcal{A}}(I_i)$ for a KDF of inputs $I_1, \dots, I_r$. An offline ceremony counts as a copy with break date ∞ . Carriers make nesting explicit: a key wrapped under RSA and carried only inside a PQ tunnel has $\gamma = \infty$ for a recorder of that tunnel, and a multi-hop layer has one carrier per hop it crosses. Copies may form cycles, for instance two key-encryption keys backed up under each other; the equations then have several solutions, and $\beta_{\mathcal{A}}$ is the *greatest* one. K is *quantum-safe* for $\mathcal{A}$ iff $\beta_{\mathcal{A}}(K) = \infty$; we drop the index when $\mathcal{A}$ contains every class.

By (A1) the attacker must recover *every* input of a KDF, hence the maximum; by (A4) recovering *any* readable copy suffices, hence the minimum; reading a copy requires both its wrapping key and its carrier, hence the inner maximum. On an acyclic graph the solution is unique. On a cyclic one, the least solution would declare two keys wrapped only under each other recoverable although the adversary has no way in. The greatest solution is obtained by starting every key at ∞ and applying the equations until nothing changes, which terminates because the graph is finite and dates take finitely many values ($z_\phi, \pm\infty$); it is the earliest date reachable by a finite derivation, which the artifact checks against forward derivation on 3,000 random cyclic graphs. The following cases, which a “PQ component” rule misclassifies, follow directly.

- A data key wrapped under an RSA key and under an ML-KEM key breaks at z_{RSA} .
- An HSM key loaded by RSA remote key loading breaks at z_{RSA} ; one loaded from offline components does not (Fig. 1).
- A TLS 1.3 session resumed with `psk_ke` after an X25519 handshake breaks at z_{ECC} , even if the client offered only PQ groups. With `psk_dhe_ke` and a hybrid share its traffic keys are safe, but its 0-RTT early data, encrypted under keys derived from the resumption secret alone, still breaks at z_{ECC} .

$\beta_{\mathcal{A}}$ is non-decreasing in every z_ϕ and can only decrease when $\mathcal{A}$ grows; K is safe for $\mathcal{A}$ iff its origin is safe and every copy is safe or unreadable, where a KDF origin is safe iff some input is.

VI. VERDICTS

A. Layers and endpoint states

A hop e is covered by layers $L(e)$: per-hop TLS, SSH or IPsec, an envelope spanning several hops or organizations, or encryption at rest. For a negotiated layer and an endpoint we use externally observable states, ordered from weakest to strongest as $N < \text{En} < \text{Pref}^* < \text{Pref} < \text{Enf}$:

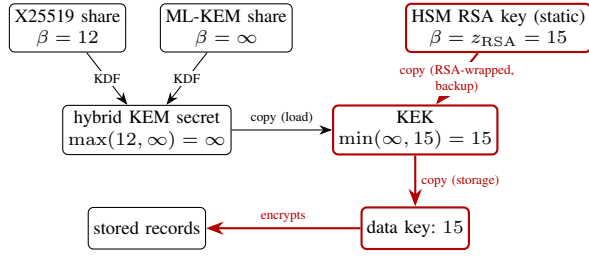


Fig. 1. Lineage of the storage keys in scenario S4 of the worked example. Arrows go from a key to the keys derived from it (KDF) or copied under it. A KDF breaks when *all* its inputs do (maximum), a key when *any* readable copy does (minimum): loading the KEK through a hybrid KEM does not help while the RSA-wrapped copy exists. In S5 a new KEK without that copy, and new data keys, have $\beta = \infty$.

- **N (None):** the endpoint cannot negotiate any PQ or hybrid group;
- **En (Enabled):** the endpoint can negotiate a PQ or hybrid group, but selects it only when the peer leaves no other choice or decides the order;
- **Pref (Preferred):** the endpoint selects a PQ or hybrid group whenever the peer lists both a PQ and a classical group, even if the peer sent a key share only for the classical one (the server then sends a HelloRetryRequest); an endpoint that does so only when the peer also sends a PQ key share is in the intermediate state Pref^* , between En and Pref;
- **Enf (Enforced):** the endpoint refuses peers that offer only classical options, including older protocol versions such as TLS 1.2.

Each state implies the capabilities of the states below it; the order expresses how much the endpoint, on its own, guarantees a PQ session. The definitions are phrased for the server, which selects the group. For a client, N means that it offers no PQ group, En that it offers one among others, Pref that it lists a PQ group first and sends its key share, and Enf that it offers only PQ groups and only TLS 1.3. When neither endpoint enforces, the two states do not determine the outcome, which depends on each implementation’s selection rule (Section VIII). Capable but disabled software is indistinguishable from N from outside. Whether removing the classical component of a hybrid is progress depends on the profile, so it is not a state.

The *layer value* $\nu_{\mathcal{A}}(\ell)$ answers one question: is *every* session that layer ℓ can establish keyed by a secret that is quantum-safe for $\mathcal{A}$? It takes three values: 1 means yes; 0 means no, some session that the configuration admits is keyed by a secret with a finite break date; and $\perp$ (“unknown”) means that the available information does not decide, typically because the state of one endpoint is not known.

Definition 2 (Layer value). Consider a negotiated layer with endpoints u, v that establishes sessions. Let $\mathcal{K}$ be the set of keys protecting traffic in the sessions it admits other than fresh classical handshakes: the keys of fresh PQ sessions, and the traffic and early-data keys of *every* resumed session, whatever handshake its ticket descends from, including tickets issued

before migration. Its value $\nu_{\mathcal{A}}(\ell) \in \{0, \perp, 1\}$ is given by the first rule that applies: (1) 0 if some key in $\mathcal{K}$ is not quantum-safe for $\mathcal{A}$; (2) 1 if u or v is in Enf , since no fresh classical session can then be established; (3) if u or v is in N , every fresh session is classical: 0 if the keys of these classical sessions are not quantum-safe for $\mathcal{A}$, and 1 otherwise (for instance when their exchange is never recorded); (4) if the configurations of both endpoints are known: 1 if the pair negotiates a PQ group in its sessions or if its classical session keys are quantum-safe for $\mathcal{A}$, and 0 otherwise; (5) $\perp$ in all remaining cases. A static layer has $\nu_{\mathcal{A}} = 1$ iff its key is quantum-safe for $\mathcal{A}$.

A layer whose endpoints cannot agree, for instance one in Enf and the other in N , establishes no session and carries no data. This is an availability failure, which the confidentiality verdict does not grade; we report such layers separately and exclude them from ν .

Proposition 3 (Hop-AND / Layer-OR). Assume (A1)–(A4), that every layer value is known (0 or 1), and (A5) joint realizability: the layers of a hop are negotiated independently, so that if each layer of a hop admits a weak session, the configuration admits carrying an item of d in weak sessions of all of them at once. Call the flow d protected against $\mathcal{A}$ if no item of d , carried in any session that the configuration of the path admits, can be recovered by a recorder of classes $\mathcal{A}$. Then d is protected against $\mathcal{A}$ iff every hop of a class in $\mathcal{A}$ has a layer with $\nu_{\mathcal{A}} = 1$ and every plaintext node of d is trusted.

Proof sketch. ($\Leftarrow$) Pick one layer with $\nu_{\mathcal{A}} = 1$ per recordable hop. Every recorded occurrence of d lies inside one of them, under keys that are quantum-safe for $\mathcal{A}$ by Definition 1; by (A2) the model grants the adversary nothing about a payload under such keys, and occurrences on unrecorded hops are not in its view. ($\Rightarrow$) Suppose every layer of a recordable hop has $\nu_{\mathcal{A}} = 0$. By rules (1), (3) and (4) of Definition 2, which all refer to quantum safety for $\mathcal{A}$, each layer admits a session whose key has a finite break date for $\mathcal{A}$; by (A5) the configuration admits carrying an item of d in such sessions on all layers of the hop at once, and by (A3)–(A4) the recorder recovers that item. An untrusted plaintext node leaks d by assumption. Only ($\Rightarrow$) uses (A5): without it the condition remains sufficient, and a verdict based on it can only err toward reporting exposure. $\square$

The “iff” holds because protection quantifies over every session the configuration admits; it does not claim that a given item is exposed, only that some admitted session exposes one.

B. Evidence and the verdict pair

An evidence record for a layer has a positive level (vendor < static < lab < interop < runtime), a *negative test* flag, a *scope* and a *date*. A negative test counts only if a positive control with otherwise identical parameters received a Server-Hello selecting a PQ group and the refusal is a handshake-family alert (Section VII). The scope records which listeners, names and backends were probed and with what confidence.

An evidence policy Π fixes the minimum level, requires the negative test for negotiated layers, and requires the scope to cover the layer's backends and the date to be within a freshness window τ of the assessment.

Definition 4 (Verdict). Let $\nu_\Pi(\ell) = \nu_A(\ell)$ if $\nu_A(\ell) \neq \perp$, and otherwise 1 if the evidence of ℓ satisfies Π and $\perp$ if it does not. A plaintext node contributes 1 to both evaluations if it is trusted, 0 if it is not, and $\perp$ if its trust is unknown. Evaluate the path in strong Kleene logic [55] with values ordered $0 < \perp < 1$, as the minimum over recordable hops and plaintext nodes of the maximum over each hop's layers, once with ν_A (value a) and once with ν_Π (value b). The verdict is the pair (a, b) . *Demonstrated* thus means demonstrated in this model, under Π , for the scope and date of the evidence; it is not a proof that the flow is secure in all conditions.

Proposition 5. *The pair takes only the values $(0, 0)$, $(\perp, \perp)$, $(1, \perp)$ and $(1, 1)$, which we call *Exposed*, *Unknown*, *Claimed* and *Demonstrated*. They form a chain under the componentwise order, and the path verdict equals the chain-minimum over hops of the chain-maximum over layers. Each component is exact with respect to completions: it is 1 (resp. 0) iff every replacement of the unknowns by 0 or 1 yields 1 (resp. 0).*

Proof. Per layer, ν_Π differs from ν_A only by turning a 1 into $\perp$, so per layer the pair is one of the four values. Componentwise min and max preserve this set, since the four pairs are totally ordered componentwise, so lattice operations on pairs coincide with chain operations. Exactness of each component holds because the formula is monotone and negation-free, so Kleene evaluation equals the evaluations at the all-0 and all-1 completions. $\square$

Thus “Unknown below Claimed” is not a policy choice: Claimed means that the claims would protect the flow but the evidence policy does not yet confirm it, while Unknown means that even the claims leave the outcome open. A hop with one unknown layer and one demonstrated layer is Demonstrated; an unknown layer is never treated as exposed.

Decision rule: Only Demonstrated counts as complete. Claimed triggers evidence collection, chiefly a controlled negative test or a refresh of stale evidence. Unknown triggers data collection, typically about the client side. Exposed triggers migration work.

C. Time

Let T_ℓ be the date from which layer ℓ has $\nu_A = 1$, let b_ℓ be the break date of the keys it used before T_ℓ , and for a hop let $T_e = \min_\ell T_\ell$ and $b_e = \max_\ell b_\ell$. Time 0 is the assessment date. As $\mathcal{K}$ includes long-lived and resumed sessions, T_ℓ is the cut-over date plus the longest session or ticket lifetime.

Proposition 6 (Exposure). *Traffic of d sent from time 0 on is exposed iff some recordable hop has $T_e > 0$ and $b_e < T_e + x_d$. Legacy traffic sent before 0 is exposed iff some recordable hop has $b_e < \min(T_e, 0) + x_d$ for the keys then in use.*

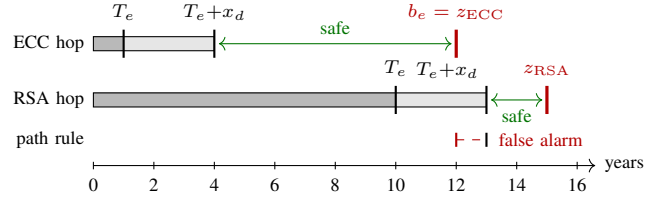


Fig. 2. Per-hop exposure (Proposition 6) with $x_d = 3$. Dark: traffic recorded before the hop migrates at T_e ; light: the same traffic while it must stay confidential, until $T_e + x_d$. A hop is exposed iff its break date b_e falls inside that window. Neither hop is exposed, but a path rule comparing the earliest break date (12) with the latest $T_e + x_d$ (13) reports exposure.

Proof. Traffic sent at t on hop e is recorded unprotected iff $t < T_e$, and is readable while sensitive iff $b_e < t + x_d$. The supremum of $t + x_d$ over $[0, T_e)$ is $T_e + x_d$, and over $t < \min(T_e, 0)$ (sent before 0 and before the hop was protected) it is $\min(T_e, 0) + x_d$. The path is exposed iff some recordable hop is. $\square$

With a single date z , this reduces to $z < \max_e T_e + x_d$, Mosca’s inequality on paths. The condition must be checked per hop: a path-level formula that combines the earliest break date with the latest migration date reports exposure where there is none, for instance for an ECC hop migrated at year 1 and an RSA hop migrated at year 10 with $x_d = 3$, $z_{\text{ECC}} = 12$ and $z_{\text{RSA}} = 15$: the earliest break date (12) is below the latest migration date plus x_d (13), yet neither hop satisfies $b_e < T_e + x_d$ ($12 < 4$ and $15 < 13$ are both false; Fig. 2).

Profiles: Compliance is evaluated separately from ν_A . A *hybrid-required* profile accepts only hybrid groups. A *PQ-only* profile accepts only ML-KEM-1024 with ML-DSA-87 authentication.

VII. WHAT A PROBE CAN ESTABLISH

An external probe acts as a client of the server end of a hop. The client end of a real hop is not observed.

Whether the keys in $\mathcal{K}$ are quantum-safe is not fully observable from outside: a server that resumes sessions without a fresh key exchange (`psk_ke`), or accepts 0-RTT early data, protects traffic under keys derived from earlier handshakes, possibly classical ones made before its migration, and whether such earlier tickets remain valid depends on ticket-key rotation, which is invisible to a client.

Proposition 7 (Identifiability). *Consider a negotiated layer that establishes sessions. From server-side observations and the recording classes, its value is determined only in two cases. If the server is in $\mathbf{N}$, rule (3) applies: $\nu_A = 0$ when the transcripts of the layer’s exchanges are readable by $\mathcal{A}$, as when the layer runs unprotected over a hop of a class in $\mathcal{A}$, and $\nu_A = 1$ otherwise. If the server is in Enf , refuses resumption without a fresh PQ key exchange, refuses 0-RTT early data, and exports no copy of its session keys, $\nu_A = 1$. In every other case $\nu_A = \perp$ until the client side, or the server’s resumption policy, is known.*

Proof. With a server in N , the layer establishes sessions only with clients not in Enf , every fresh session is classical, and rule (3) decides by Definition 1: a classical exchange is recoverable for $\mathcal{A}$ iff some carrier of its transcript is readable by $\mathcal{A}$. With a server in En , $Pref^*$ or $Pref$, a PQ-only client negotiates a PQ group and a classical-only client a classical one, so both values are consistent with the observations. With a server in Enf that accepts `psk_ke` or early data, a resumed session may carry traffic under keys descending from a pre-migration classical handshake, so rule (1) cannot be decided from outside. Otherwise rule (2) applies. $\square$

The probe observes whether a server *selects* `psk_ke` when a client offers it alongside `psk_dhe_ke`, and whether it accepts early data on resumption; with our client it cannot exclude that a server accepts `psk_ke` when it is the only mode offered, and no client can see whether session keys are copied out of the endpoint, for instance to a decryption appliance for monitoring; a Demonstrated verdict records both as stated assumptions of its evidence. Server-side observations therefore yield Demonstrated, Exposed or Unknown for a hop; they never justify Claimed on their own. Within an organization, where both ends are known, the client side is read from configuration or observed directly.

Attributing a refusal: A failed classical-only handshake can be caused by many things: a required client certificate, an unknown name, an access list, a different backend. A refusal counts as PQ enforcement only if (i) a positive control, identical except that it offers PQ groups, receives a ServerHello selecting a PQ group (the handshake may still fail afterwards, for instance for lack of a client certificate), and (ii) the failure is a TLS alert of the handshake-failure family (`handshake_failure`, `protocol_version`, `insufficient_security`). Transport resets, timeouts and other alerts are *inconclusive*. A refusal demonstrates enforcement only for the classical options offered: the probe therefore offers, in one ClientHello, all 13 classical TLS 1.3 groups known to OpenSSL 3.5, and runs a separate TLS 1.2 test. Fig. 3 summarizes the resulting decision procedure.

Proposition 8 (Sampling). *Suppose each connection to a service reaches a backend drawn independently of the others, and backends that accept classical-only peers receive a fraction π of connections. Let N be the number of negative handshakes, i.e. connections in which the probe offers only classical options, sent to the service, and let $\delta \in (0, 1)$ be the largest probability, accepted by the assessor, of wrongly reporting Enf . A probe that reports Enf only if all of its N negative handshakes are refused reports Enf wrongly with probability $(1 - \pi)^N$. Hence, under this independence assumption, $N \geq \ln(1/\delta) / \ln(1/(1 - \pi))$ negative handshakes bound by δ the probability of reporting Enf for a service in which a fraction π or more of connections can be served classically.*

Proof. Each negative handshake independently avoids every non-enforcing backend with probability $1 - \pi$, so all N of them do with probability $(1 - \pi)^N$, which is the probability of a wrong Enf report. Requiring this probability to be at most

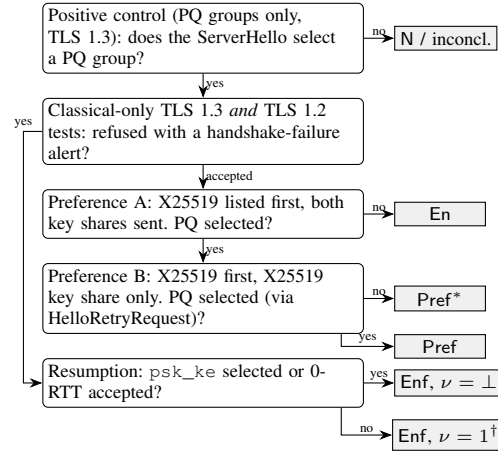


Fig. 3. How the probe infers a server state. Each test is repeated R times; differing outcomes give *heterogeneous*, resets and other alerts *inconclusive*; a failed positive control gives N only with a handshake-failure alert. $1^\dagger$: assuming `psk_ke` offered alone is refused (Proposition 7).

the accepted error δ , i.e. $(1 - \pi)^N \leq \delta$, and solving for N gives the bound. $\square$

Real balancers violate independence: some choose among candidates by current load, and sticky sessions send repeated connections from one client to one backend, in which case repetition adds no information. The bound is therefore optimistic; repeats should be spread over time and, where possible, over source addresses, and the evidence record should state the balancing assumption. The scope of a Demonstrated verdict is the triple (listener and name, π , δ), together with this assumption and the date. Outcomes that differ across repeats are reported as *heterogeneous* and the weakest state is kept.

VIII. EVALUATION

The evaluation validates the inference rules on controlled ground truth. It is not a measurement of deployed systems; that study is specified in the artifact (Section IX).

Probe: `pprobe` drives an OpenSSL 3.5 client and implements the procedure of Fig. 3. It also tests pure ML-KEM-1024 and PQ and classical-only signature algorithms, and, when classical handshakes are accepted, resumes a classical session with a PQ-only client that allows `psk_ke`. It reads the selected group and the presence of a key share from every ServerHello. Each step can be repeated R times; a run uses $9R$ handshakes plus at most six for the resumption and 0-RTT tests, and records the negotiated cipher suite.

A. Multi-implementation validation

We built OpenSSL 3.5.4, nginx 1.29.1 and HAProxy 3.2.0 against it, and wrote minimal servers with Go 1.24.7 `crypto/tls` and `rustls` 0.23.45 with `aws-lc-rs`: five server stacks on three TLS libraries, since nginx and HAProxy use OpenSSL. We probed 22 configurations, each five times (Table I). They include two confounders: a server that requires client certificates and a listener that resets every connection.

ID	Implementation	Configuration	Expected	Inferred	KEX	Cl. 1.3 / 1.2	Auth	Resum.	ν	Profile
O-A	OpenSSL 3.5.4	hybrid only, TLS 1.3 only	Enf	Enf		R / R	N	no	1 [†]	hybrid
O-B	OpenSSL 3.5.4	X25519, then hybrid (server order)	En	En		A / A	N	no	⊥	—
O-C	OpenSSL 3.5.4	classical only	N	N		A / A	N	—	0	—
O-D	OpenSSL 3.5.4	hybrid, then X25519 (server order, :)	Pref*	Pref*		A / A	N	no	⊥	—
O-D2	OpenSSL 3.5.4	hybrid, then X25519 (server order, /)	Pref	Pref		A / A	N	no	⊥	—
O-E	OpenSSL 3.5.4	hybrid only, ML-DSA-65 certificate	Enf	Enf		R / R	Enf	no	1 [†]	hybrid
O-F	OpenSSL 3.5.4	as O-D, <code>psk_ke</code> allowed and preferred	Pref*	Pref*		A / A	N	sel., inh.	⊥	—
O-H	OpenSSL 3.5.4	as O-A, <code>psk_ke</code> allowed and preferred	Enf	Enf		R / R	N	sel.	⊥	hybrid
O-I	OpenSSL 3.5.4	as O-A, 0-RTT early data accepted	Enf	Enf		R / R	N	0-RTT	⊥	hybrid
O-G	OpenSSL 3.5.4	ML-KEM-1024 only, ML-DSA-87 certificate	Enf	Enf		R / R	Enf	no	1 [†]	PQ-only
N-1	nginx 1.29.1	<code>ssl_ecdh_curve</code> X25519MLKEM768, TLS 1.3	Enf	Enf		R / R	N	no	1 [†]	hybrid
N-2	nginx 1.29.1	<code>ssl_ecdh_curve</code> X25519MLKEM768:X25519	—	En		A / A	N	no	⊥	—
N-3	nginx 1.29.1	as N-2, <code>ssl_prefer_server_ciphers</code> on	—	Pref*		A / A	N	no	⊥	—
H-1	HAProxy 3.2.0	<code>curves</code> X25519MLKEM768, min TLS 1.3	Enf	Enf		R / R	N	no	1 [†]	hybrid
H-2	HAProxy 3.2.0	<code>curves</code> X25519MLKEM768:X25519	—	Pref*		A / A	N	no	⊥	—
G-1	Go 1.24.7	<code>curves</code> {X25519MLKEM768}, TLS 1.3	Enf	Enf		R / R	N	no	1 [†]	hybrid
G-2	Go 1.24.7	<code>curves</code> {X25519MLKEM768, X25519, P-256}	—	Pref		A / A	N	no	⊥	—
G-3	Go 1.24.7	<code>curves</code> {X25519, P-256, X25519MLKEM768}	—	Pref		A / A	N	no	⊥	—
R-1	rustls 0.23.45	<code>kx groups</code> {X25519MLKEM768}, TLS 1.3	Enf	Enf		R / R	N	no	1 [†]	hybrid
R-2	rustls 0.23.45	<code>kx groups</code> {X25519MLKEM768, X25519, P-256}	—	En		A / A	N	no	⊥	—
X-mTLS	OpenSSL 3.5.4	hybrid only, client certificate required	Enf	Enf		R / R	N	n/t	⊥	hybrid
X-ACL	raw TCP	every connection reset	inconcl.	inconcl.		I / I	—	—	⊥	—

TABLE I

INFERRED SERVER STATES ON 22 CONFIGURATIONS OF FIVE SERVER STACKS BUILT ON THREE TLS LIBRARIES (5 RUNS EACH, ALL IDENTICAL). “EXPECTED” IS THE STATE IMPLIED BY THE CONFIGURATION WHERE IT IS UNAMBIGUOUS; “—” MARKS CONFIGURATIONS WHOSE STATE DEPENDS ON THE IMPLEMENTATION’S SELECTION RULE. CL. 1.3 / 1.2: OUTCOME OF THE CLASSICAL-ONLY TLS 1.3 AND TLS 1.2 NEGATIVE TESTS (R = REFUSED, A = ACCEPTED, I = INCONCLUSIVE). RESUM.: AFTER A PQ HANDSHAKE, THE SERVER SELECTS RESUMPTION WITHOUT A FRESH KEY EXCHANGE (SEL.), A PQ-ONLY CLIENT THEN INHERITS A CLASSICAL HANDSHAKE (INH.), OR THE SERVER ACCEPTS 0-RTT EARLY DATA (0-RTT); N/T = NOT TESTABLE WITHOUT CLIENT CREDENTIALS. ν : LAYER VALUE DETERMINED FROM THE SERVER SIDE FOR A RECORDER OF THE PROBED HOP (PROPOSITION 7); 1[†] = UNDER THE STATED ASSUMPTION THAT `psk_ke` OFFERED ALONE IS REFUSED.

Inference: In all 16 configurations with an unambiguous expectation, the inferred state matched it, and all five runs of every configuration were identical. Refusals in enforcing configurations were `handshake_failure` (classical TLS 1.3) and `protocol_version` (TLS 1.2) alerts. The mTLS server is correctly reported as `Enf`: in TLS 1.3 the group is refused before any certificate is requested, and the positive control receives a `ServerHello` selecting the hybrid group before the handshake fails for lack of a client certificate. Its resumption policy cannot be tested without client credentials, so its layer value stays unknown from outside. The resetting listener is reported as inconclusive, not as enforcing.

Configuration is not state: The same nominal group list produced different states. With the list “X25519MLKEM768, X25519”, nginx was in `En`, since the client’s order decided, and HAProxy in `Pref`. The libraries are the same; the default differs: HAProxy applies server preference, nginx does not, and enabling `ssl_prefer_server_ciphers` put nginx in the same state as HAProxy (N-3). With OpenSSL, even server preference yields only `Pref*`: groups separated by : form one tuple, within which the server takes the first group for which the client sent a key share, so a client that sends only an X25519 share obtains X25519. Separating them with / (O-D2) makes the server send a `HelloRetryRequest` for the hybrid group, i.e. `Pref`. Go selected the hybrid group, through a `HelloRetryRequest` if needed, even when configured with a classical-first list (G-3), whereas rustls with a PQ-first list followed the client’s order (`En`). An assessment that reads only the group list would mislabel most of these configurations. Go

also negotiated AES-128-GCM where the others chose AES-256-GCM, which a policy following (A2) should flag. This is the reason why Definition 4 demonstrates only from runtime evidence with a negative test.

Lineage: The probe parses each `ServerHello`: a resumed session without a `key_share` extension was established with `psk_ke`. In O-F, a client offering only ML-KEM-based groups was served such a session after an earlier X25519 handshake. In O-H, an enforcing server selected `psk_ke` after a hybrid handshake: it would be reported as `Enf`, and Demonstrated, by a probe that tested only negotiation, although sessions resumed from pre-migration tickets would inherit classical keys (scenario S3r below). The probe therefore leaves its layer value unknown. O-I, an enforcing server that accepts 0-RTT, is treated in the same way, since its early data is protected by keys derived from the resumption secret alone. These are known properties of TLS 1.3 resumption [13], not new findings; no other server selected `psk_ke` or accepted early data. They validate the lineage rule and Proposition 7.

B. Load balancer

An HAProxy TCP load balancer distributed connections with its `random` algorithm between an enforcing backend (as O-A) and a non-enforcing one (as O-D), so $\pi = 1/2$. Each run of the probe makes $2R$ negative handshakes. Under independence, a run reports `Enf` with probability 2^{-2R} , and is not flagged as heterogeneous with probability $4 \cdot 2^{-2R}$ (both negative tests homogeneous). Table II pools two campaigns of 200 runs per value of R .

R	False Enf	Indep.	Heterog.	Pref	Non-het.
1	28.0%	25.0%	—	72.0%	400 / 400
2	7.3%	6.3%	76.0%	16.8%	96 / 100
4	1.0%	0.4%	96.5%	2.5%	14 / 6.3

TABLE II

PROBE OUTCOMES BEHIND A LOAD BALANCER WITH ONE ENFORCING AND ONE NON-ENFORCING BACKEND, HAPROXY RANDOM (TWO CAMPAIGNS, 400 RUNS PER ROW). INDEP.: FALSE-Enf RATE $(1 - \pi)^{2R}$ EXPECTED UNDER INDEPENDENCE, WITH $\pi = 1/2$; IT IS NOT AN UPPER BOUND. NON-HET.: RUNS NOT FLAGGED HETEROGENEOUS, OBSERVED / EXPECTED UNDER INDEPENDENCE.

For $R = 1$ and $R = 2$ the counts are consistent with the independence model (one-sided binomial $p = 0.09$ and $p = 0.23$ for false Enf). For $R = 4$ they are not: 14 runs were not flagged as heterogeneous where 6.3 were expected ($p \approx 0.005$), and 4 runs reported a false Enf where 1.6 were expected ($p \approx 0.07$). Per campaign, $R = 4$ gave 3 false Enf and 9 non-heterogeneous runs in the first, and 1 and 5 in the second. Our hypothesis was HAProxy’s random algorithm, which by default draws two candidates and keeps the less loaded, so that successive choices depend on the connections in progress. Two controls at $R = 4$ (200 runs each) support it. With random(1), a single uniform draw, the counts matched independence: no false Enf (0.8 expected) and 3 non-heterogeneous runs (3.1 expected). With roundrobin, which alternates deterministically, every run was flagged heterogeneous. Repetition thus exposes heterogeneity, but the false-Enf rate depends on the balancing algorithm, and Proposition 8 gives its value only under independence: a probe of a load-balanced name should never be reported as Demonstrated without its sampling scope and balancing assumption.

C. Worked example: an interbank credit transfer

We model a stylized ISO 20022-style flow [56] from Bank A through a central infrastructure (CSM) to Bank B over mTLS, with internal hops at each organization, two interconnect hops, a trusted processing node and a storage hop at the CSM. The storage data key is wrapped under a key-encryption key (KEK) loaded into the HSM. All values are hypothetical and describe no real operator. We take $z_{\text{ECC}} = 12$, $z_{\text{RSA}} = 15$, and a policy requiring runtime evidence with a controlled negative test, at most one year old (Table III).

S1 satisfies an “interconnections migrated” milestone but leaves the flow exposed to an insider recorder. S2 is only claimed: both ends of the internal hops prefer the hybrid group, so the pair negotiates it and traffic is protected at face value (hence the readable date of 15, set by the storage key), but neither end refuses classical peers and no negative test can demonstrate it. S4 shows the copy semantics (Fig. 1). Destroying that copy would not help either against a recorder who already captured it: by Definition 1, the data key wrapped under the old KEK still breaks at z_{RSA} . Only S5, which generates a new KEK and new data keys and re-encrypts the data, is demonstrated under every recording class, for data written from now on; ciphertexts recorded before the re-encryption remain readable from z_{RSA} . These private,

Scenario	External	+Internal	+Storage	Readable
S0 all classical	Ex	Ex	Ex	12
S1 interconnects enforced	De	Ex	Ex	12
S2 + internal hops hybrid-preferred	De	Cl	Ex	15
S3 all hops enforced; KEK loaded by RSA	De	De	Ex	15
S3r as S3, one interconnect resumes pre-migration tickets with <code>psk_ke</code>	Ex	Ex	Ex	12
S4 KEK re-loaded by hybrid KEM, RSA-wrapped copy kept in backup	De	De	Ex	15
S5 new KEK and data keys, data re-encrypted	De	De	De	∞

TABLE III

VERDICTS OF THE INTERBANK FLOW: EX = EXPOSED, UN = UNKNOWN, CL = CLAIMED, DE = DEMONSTRATED (PROPOSITION 5). “READABLE” IS THE DATE (YEARS) FROM WHICH TRAFFIC RECORDED TODAY BY A RECORDER OF ALL CLASSES CAN BE READ, TAKING CLAIMS AT FACE VALUE.

mutually authenticated interconnects are out of reach of an external probe: each participant runs it from inside with its credentials and submits the evidence to the coordinator.

IX. DISCUSSION AND LIMITATIONS

No field measurement yet: All results are from controlled testbeds and a stylized example. The artifact contains `pqscan` and a protocol for an internet-scale study of support, preference and enforcement (rate limits, scanner identification, opt-out, aggregate publication, prior ethics and data-protection review following the Menlo Report [57]). Private interconnects are out of reach of such a study by design.

Probe limits: The probe uses one client implementation, sees only what a peer can negotiate, and cannot see backends that its connections do not reach; a server may also treat probe clients differently. Evidence is a dated sample; drift [14] motivates the freshness window.

Composition: Composition across organizations, and what interfaces disclose, is treated in Appendix A.

Model limits: Assumptions (A1)–(A5) exclude implementation flaws and key compromise outside lineage. Profiles simplify evolving guidance such as IR 8547. The CRQC dates are elicited and enter only through comparisons, so results should be read for ranges of z_ϕ .

X. CONCLUSION

Whether a flow is protected against a future quantum adversary depends on the lineage of its keys, on what the endpoints of each hop refuse, and on the evidence for both. Key lineage makes the first precise, including copies and resumption. The verdict pair separates what claims imply from what evidence demonstrates, with exact semantics for unknowns. The probe results delimit what can be established from outside and show that declared configuration does not predict negotiated state. For practitioners, this suggests per-flow verdicts instead of migrated-asset percentages, controlled negative tests with a stated scope and date, ticket-key rotation at cut-over, and re-keying with re-encryption of data whose keys had classically wrapped copies.

REFERENCES

- [1] NIST, “Module-lattice-based key-encapsulation mechanism standard,” NIST, Tech. Rep. FIPS 203, 2024.
- [2] D. Stebila, S. Fluhrer, and S. Gueron, “Hybrid key exchange in TLS 1.3,” RFC 9954 (Informational), Jul. 2026.
- [3] K. Kwiatkowski, P. Kampanakis, B. E. Westerbaan, and D. Stebila, “Post-quantum traditional (PQ/T) hybrid key agreement mechanisms for TLS 1.3,” RFC 10024 (Proposed Standard), Aug. 2026.
- [4] P. Kampanakis, D. Stebila, and T. Hansen, “Post-quantum/traditional hybrid key exchange with the module-lattice-based key-encapsulation mechanism for use in SSH,” RFC 10042 (Informational), Aug. 2026.
- [5] Cloudflare, “The 2025 Cloudflare Radar year in review,” Cloudflare blog, <https://blog.cloudflare.com/radar-2025-year-in-review/>, Dec. 2025.
- [6] D. Moody, R. Perlner, A. Regenscheid, A. Robinson, and D. Cooper, “Transition to post-quantum cryptography standards,” NIST, Tech. Rep. IR 8547 (Initial Public Draft), Nov. 2024.
- [7] NIS Cooperation Group, “A coordinated implementation roadmap for the transition to post-quantum cryptography, version 1.1,” European Commission, Jun. 2025.
- [8] G7 Cyber Expert Group, “Advancing a coordinated roadmap for the transition to post-quantum cryptography in the financial sector,” Statement and roadmap, U.S. Treasury publication, Jan. 2026.
- [9] PCI Security Standards Council, “Payment card industry data security standard, version 4.0.1,” Requirement 12.3.3, Jun. 2024.
- [10] A. A. Fall, “SoK: Systematizing hybrid strategies for the transition to post-quantum cryptography,” Cryptology ePrint Archive, Paper 2025/2052, 2025. [Online]. Available: <https://eprint.iacr.org/2025/2052>
- [11] M. Mosca, “Cybersecurity in an era with quantum computers: Will we be ready?” *IEEE Security & Privacy*, vol. 16, no. 5, pp. 38–41, 2018.
- [12] M. Mosca and J. Mulholland, “A methodology for quantum risk assessment,” Global Risk Institute, 2017.
- [13] E. Rescorla, “The transport layer security (TLS) protocol version 1.3,” RFC 9846 (Standards Track; obsoletes RFC 8446), 2026.
- [14] Y. Han, “Nothing breaks: No single peer can soundly gate post-quantum delivery,” arXiv:2609.07849, Sep. 2026.
- [15] F. Giacon, F. Heuer, and B. Poettering, “KEM combiners,” in *Public-Key Cryptography (PKC 2018)*, ser. LNCS, vol. 10769. Springer, 2018, pp. 190–218.
- [16] N. Bindel, J. Brendel, M. Fischlin, B. Goncalves, and D. Stebila, “Hybrid key encapsulation mechanisms and authenticated key exchange,” in *Post-Quantum Cryptography (PQCrypto 2019)*, ser. LNCS, vol. 11505. Springer, 2019, pp. 206–226.
- [17] B. Dowling, T. B. Hansen, and K. G. Paterson, “Many a mickle makes a muckle: A framework for provably quantum-secure hybrid key exchange,” in *Post-Quantum Cryptography (PQCrypto 2020)*, ser. LNCS, vol. 12100. Springer, 2020, pp. 483–502.
- [18] NIST, “Recommendations for key-encapsulation mechanisms,” NIST, Tech. Rep. SP 800-227, Sep. 2025.
- [19] ANSSI, “ANSSI views on the post-quantum cryptography transition,” Position paper, 2022.
- [20] —, “ANSSI views on the post-quantum cryptography transition (2023 follow up),” Position paper, 2023.
- [21] Bundesamt für Sicherheit in der Informationstechnik, “Cryptographic mechanisms: Recommendations and key lengths,” BSI, Tech. Rep. TR-02102-1, Version 2026-01, 2026.
- [22] National Security Agency, “Announcing the commercial national security algorithm suite 2.0,” Cybersecurity Advisory, 2022.
- [23] BIS Innovation Hub, “Project Leap phase 2: Quantum-proofing payment systems,” Technical report, with Banca d’Italia, Banque de France, Deutsche Bundesbank, Nexi-Colt and Swift, Dec. 2025.
- [24] M. Mosca and M. Piani, “Quantum threat timeline report 2025,” Global Risk Institute and evolutionQ, 2026, report dated 9 March 2026.
- [25] M. Roetteler, M. Naehrig, K. M. Svore, and K. Lauter, “Quantum resource estimates for computing elliptic curve discrete logarithms,” in *Advances in Cryptology – ASIACRYPT 2017*, ser. LNCS, vol. 10625. Springer, 2017, pp. 241–270.
- [26] NIST NCCoE, “Migration to post-quantum cryptography,” NIST, Tech. Rep. SP 1800-38, 2023.
- [27] ETSI TC CYBER, “CYBER; quantum-safe cryptography (QSC); a repeatable framework for quantum-safe migrations,” ETSI, Tech. Rep. TR 104 016 V1.1.1, Oct. 2024.
- [28] TNO and CWI and AIVD, “The PQC migration handbook: Guidelines for migrating to post-quantum cryptography,” Second edition, Dec. 2024.
- [29] E. Nzetchuen, M. Stöttinger, A. Wiesmaier, and E. Yeşilbek, “A study of PQC migration frameworks,” in *MAGiCS 2026*, ser. CCIS, vol. 3002. Springer, 2026, pp. 61–81.
- [30] B. Hess, “The anatomy of cryptography bills of materials: Standardization and practice in CycloneDX,” in *Migration and Agility in Cryptographic Systems (MAGiCS 2026)*, ser. CCIS, vol. 3002. Springer, 2026, pp. 1–19.
- [31] E. Hirsch and K. Raab, “Architecture-derived CBOMs for cryptographic migration: A security-aware architecture tradeoff method,” in *MAGiCS 2026*, ser. CCIS, vol. 3002. Springer, 2026, pp. 20–39.
- [32] J. Dev, “Cryptography at scale: Building a practical cryptographic inventory for risk, compliance, and post-quantum readiness,” Talk, USENIX Security ’26, 2026.
- [33] —, “Output schema based on cryptographic bill of materials (CBOM) for post-quantum cryptography usage,” IETF Internet-Draft draft-dev-xpiper-cbom-extension-00 (individual, work in progress), 2026.
- [34] D. Herzinger, C. Näther, and S.-L. Gazdag, “Supporting cryptographic migration with an IT infrastructure digital twin,” in *MAGiCS 2026*, ser. CCIS, vol. 3002. Springer, 2026, pp. 40–60.
- [35] K. Loizou and E. Ghadafi, “Measuring post-quantum TLS deployment across UK internet sectors,” arXiv:2608.02147, 2026.
- [36] J. L. Delgado, “Observability for post-quantum TLS readiness: A multi-surface evidence framework,” arXiv:2605.02978, May 2026.
- [37] N. Wickramasinghe, F. Li, S. Jha, and A. Shaghaghi, “Mind the gap: Policy vs reality in post-quantum TLS deployment,” arXiv:2607.29005, 2026.
- [38] D. Wetter *et al.*, “testssl.sh,” Software, <https://github.com/testssl/testssl.sh>, 2026.
- [39] A. Diquet *et al.*, “SSLyze,” Software, <https://github.com/nabla-c0d3/sslyze>, 2026.
- [40] H. Kario *et al.*, “tlsfuzzer,” Software, <https://github.com/tlsfuzzer/tlsfuzzer>, 2026.
- [41] J. Somorovsky, “Systematic fuzzing and testing of TLS libraries,” in *Proc. ACM CCS*, 2016.
- [42] Z. Durumeric, E. Wustrow, and J. A. Halderman, “ZMap: Fast internet-wide scanning and its security applications,” in *USENIX Security Symposium*, 2013.
- [43] K. Hasan, L. Simpson, M. A. R. Bae, C. Islam, Z. Rahman, W. Armstrong, P. Gauravaram, and M. McKague, “A framework for migrating to post-quantum cryptography: Security dependency analysis and case studies,” *IEEE Access*, vol. 12, pp. 23 427–23 450, 2024.
- [44] A. F. Herrera, M. Gadimaliyev, Y. Meng, and L. Wang, “A graph-based framework for cryptographic risk modeling in the post-quantum transition: TLS 1.3 and TLCP analysis,” *Information and Software Technology*, vol. 195, p. 108099, 2026.
- [45] R. Erlemann, C. C. Morris, and S. Sathe, “Full-stack knowledge graph and LLM framework for post-quantum cyber readiness,” arXiv:2601.03504, Jan. 2026.
- [46] Y.-L. Hyoung, S.-M. Jeong, D.-E. Lim, D.-Y. Park, S.-B. Cho, J.-H. Kim, H.-J. Kim, and H.-J. Seo, “A layered risk scoring model for TLS connections against quantum threats,” Cryptology ePrint Archive, Paper 2026/1174, Jun. 2026.
- [47] S. Krušniuskas, Š. Grigaliūnas, R. Brūzgienė, and M. Cayir, “A quantum-adjusted risk model for enterprise infrastructure across data in transit, in use, and at rest,” *Electronics*, vol. 15, no. 12, p. 2546, 2026.
- [48] D. Loebenberger, E. Hirsch, S.-L. Gazdag, D. Herzinger, C. Näther, and J.-P. Steghöfer, “A formal model and lower-bound intuition for cryptographic migration,” in *MAGiCS 2026*, ser. CCIS, vol. 3002. Springer, 2026, pp. 82–101.
- [49] R. E. Barlow and F. Proschan, *Statistical Theory of Reliability and Life Testing: Probability Models*. Holt, Rinehart and Winston, 1975.
- [50] W. E. Vesely, F. F. Goldberg, N. H. Roberts, and D. F. Haasl, “Fault tree handbook,” U.S. Nuclear Regulatory Commission, Tech. Rep. NUREG-0492, 1981.
- [51] B. Schneier, “Attack trees,” *Dr. Dobbs’s Journal*, vol. 24, no. 12, pp. 21–29, 1999.
- [52] S. Mauw and M. Oostdijk, “Foundations of attack trees,” in *Information Security and Cryptology (ICISC 2005)*, ser. LNCS, vol. 3935. Springer, 2006, pp. 186–198.
- [53] B. Kordy, L. Piètre-Cambacédès, and P. Schweitzer, “DAG-based attack and defense modeling: Don’t miss the forest for the attack trees,” *Computer Science Review*, vol. 13–14, pp. 1–38, 2014.

- [54] M. Pollack, “The maximum capacity through a network,” *Operations Research*, vol. 8, no. 5, pp. 733–736, 1960.
- [55] S. C. Kleene, *Introduction to Metamathematics*. North-Holland, 1952.
- [56] ISO, “ISO 20022 financial services – universal financial industry message scheme,” International Organization for Standardization, 2013.
- [57] D. Dittrich and E. Kenneally, “The Menlo report: Ethical principles guiding information and communication technology research,” U.S. Department of Homeland Security, Tech. Rep., 2012.
- [58] J. E. Kelley and M. R. Walker, “Critical-path planning and scheduling,” in *Proc. Eastern Joint IRE-AIEE-ACM Computer Conference*, 1959, pp. 160–173.
- [59] European Parliament and Council, “Regulation (EU) 2022/2554 on digital operational resilience for the financial sector (DORA),” Official Journal of the European Union, 2022.
- [60] —, “Directive (EU) 2022/2555 on measures for a high common level of cybersecurity across the union (NIS2),” Official Journal of the European Union, 2022.

APPENDIX

Completion dates across organizations follow a critical-path recurrence when the state-level dependency graph is acyclic [58]. Verdicts and dates compose through interfaces.

Proposition 9. *No function of the multiset of an organization’s edge dates composes path dates exactly; in particular migrated-asset percentages do not.*

Proof. Edges $p \rightarrow a$, $a \rightarrow q$, $a \rightarrow r$ with dates $(0, 5, 0)$ or $(0, 0, 5)$ have equal multisets but give path dates 5 and 0 for a flow from p to q . $\square$

Let Λ be the declared layers that cross organizational boundaries, such as an end-to-end message envelope between two banks across the CSM. Treat edges and plaintext nodes uniformly as *elements*, each with a local date m (max–min over local layers), a local verdict w (Proposition 5) and the set $C \subseteq \Lambda$ of crossing layers covering it. We assume, as in the flat model, that all organizations evaluate with the same recording classes $\mathcal{A}$, policy Π and dates z_ϕ , and that the cover of an element does not depend on the route through it. The *flat* date of a walk is the maximum over its elements of $\min(m, \min_{\lambda \in C} T_\lambda)$, its flat verdict the minimum of $\max(w, \max_{\lambda \in C} w_\lambda)$ (Layer-OR between local and crossing layers, Hop-AND along the walk); a flow’s date and verdict are the worst over the walks it may take. For ports (p, q) , an element is *relevant* if it lies on some p – q walk, i.e. it is reachable from p and reaches q ; walks may revisit elements, which changes neither set nor result. Let M_C and W_C be the maximum date and the minimum verdict over relevant elements with cover C . Taking all walks makes the interface a worst case over internal routing: it is exact when the flow may take any walk between the ports, and conservative when routing is fixed, in which case the same construction applies to the declared routes only.

Theorem 10 (Interface). *Given the dates T_λ and verdicts w_λ of the crossing layers, an organization contributes, for a flow that may take any walk between its ports, the date $\max_C \min(M_C, \min_{\lambda \in C} T_\lambda)$ and the verdict $\min_C \max(W_C, \max_{\lambda \in C} w_\lambda)$ to a flow through (p, q) . The*

composite date and verdict of a flow crossing several organizations are the maximum date and the minimum verdict of the contributions and of the links between organizations.

Proof. If the flow may take any p – q walk, the worst flat date over walks is the maximum of $\min(m, \min_{\lambda \in C} T_\lambda)$ over relevant elements: every relevant element lies on some walk, and every walk consists of relevant elements. Grouping relevant elements by cover, $\min(\cdot, t)$ is non-decreasing in m for fixed $t = \min_{\lambda \in C} T_\lambda$, so the maximum within a group is attained at M_C ; this gives the stated date, and the verdict follows dually with $\max$, $\min$ and W_C . For a chain of organizations joined by links, a walk of the flow is a concatenation of per-organization walks and links, so the relevant elements of the chain are the union of the per-organization relevant elements and the links, and the maximum (minimum) over a union is the maximum (minimum) of the parts. With fixed routes, the same argument applies with relevant elements restricted to the declared routes. $\square$

We checked the implementation against the flattened computation on 10,000 random compositions; this verifies the code, not the theorem.

Exposure through interfaces: Dates alone do not decide exposure when CRQC dates differ by family: Proposition 6 compares b_e and T_e hop by hop, and the ECC/RSA example of Section VI applies unchanged across organizations. The interface must therefore carry, for each cover C and each local break date b (in practice, each algorithm family), the date $M_{C,b}$, the maximum of m over elements with cover C whose pre-migration keys break at b . Forward traffic with lifetime x is then exposed iff some entry has $T = \min(M_{C,b}, \min_{\lambda \in C} T_\lambda) > 0$ and $\max(b, \max_{\lambda \in C} b_\lambda) < T + x$, where b_λ is the break date of the crossing layer λ ; this follows from Proposition 6 because the condition is monotone in m . On 10,000 random compositions (local dates uniform on $\{0, \dots, 10\}$ years, local and envelope break dates uniform on $\{12, 15, \infty\}$, envelope coverage 0.4, lifetime uniform on $\{1, \dots, 10\}$), this refined interface decided exposure exactly, whereas a date-only interface combined with the earliest break date erred in 6.2% of them; the rate depends on this distribution.

What an interface discloses: An entry reveals, per port pair, at most $2^{|\Lambda|}$ pairs (M_C, W_C) , and one date per family for exposure, not edge labels. It still reveals that some walk not covered by an envelope stays classical until $M_\emptyset$, information that operators treat as security-sensitive and that is shared, in the European financial sector, within the ICT risk-management and supervisory relations framed by DORA and NIS2 [59], [60]. Disclosing only threshold bits $[M_C \leq D]$ answers deadline questions exactly, but bits for several deadlines reconstruct the date to a bucket, and a negative bit is precisely the sensitive fact. Interfaces should therefore go to the coordinator over a confidential channel, or be replaced by zero-knowledge proofs over committed graphs, which we leave to future work.