Why Optimal Cryptographic Combiners Do Not Get Deployed: Security, Complexity, and Adoption of Hybrid KEMs

Systematization of Knowledge

MERLAND CHRISLAIN CHADREL BAFOUETILA, Institut de Recherche Technologique Systemx, France
ANIS BKAKRIA, Institut de Recherche Technologique Systemx, France

XtM (XOR-then-MAC) is provably optimal against quantum adversaries [2]. As of March 2025, no production cryptographic library implements it. HKDF, with weaker security guarantees, is deployed in 91% of the 44 libraries we examined. This gap is not accidental.

This *Systematization of Knowledge* (SoK) introduces the (A, P, ϕ) framework to explain it: A measures authentication strength, P measures IETF standardization maturity, and ϕ measures implementation complexity. To our knowledge, this is **the first falsifiable, quantitative model predicting cryptographic adoption** grounded in observable software engineering indicators. We apply this framework to seven combiner families and 44 cryptographic libraries, validate ϕ against measured integration LOC across 9 real-world repositories, and derive predictions verifiable by 2028.

Our evidence suggests that implementation complexity is a first-order explanatory factor in cryptographic adoption. The most deployable construction is not the most secure one in isolation: it is the most secure one engineers can integrate, audit, and maintain at scale.

CCS Concepts: • **Security and privacy** → **Key management**; **Cryptography**; *Systems security*; • **Software and its engineering** → *Software deployment*.

Additional Key Words and Phrases: hybrid KEM; post-quantum cryptography; HKDF; X-Wing; XtM; implementation complexity; adoption study; systematization of knowledge; Pareto frontier; falsifiable prediction

ACM Reference Format:
Merland Chrislain Chadrel Bafouetila and Anis Bkakria. 2026. Why Optimal Cryptographic Combiners Do Not Get Deployed: Security, Complexity, and Adoption of Hybrid KEMs: Systematization of Knowledge. *ACM Trans. Priv. Sec.* 0, 0, Article 0 (2026), 29 pages.

1 Introduction

Consider a cryptographic construction proved optimal against the most powerful quantum adversaries. It exists: XtM (XOR-then-MAC), published in 2019 by Bindel et al. [2]. It offers security guarantees that no other hybrid key combiner matches. Yet as of March 2025, not a single production cryptographic library implements it.

At the same time, HKDF [30], whose theoretical guarantees are weaker, is deployed in 91% of the 44 libraries we examined and is integrated into TLS 1.3 [32], QUIC [33], and HPKE [34]. This gap is not accidental. It illustrates a tension this SoK seeks to partially formalize: between optimal security in theory and deployable security in practice.

Authors' Contact Information: Merland Chrislain Chadrel Bafouetila, Institut de Recherche Technologique Systemx, Palaiseau, France; Anis Bkakria, Institut de Recherche Technologique Systemx, Palaiseau, France, contact@irt-systemx.fr.

The hybrid combiner problem. The NIST standardization of ML-KEM (FIPS 203 [12]), ML-DSA (FIPS 204 [13]), and SLH-DSA (FIPS 205 [14]) in 2024 shifted the practitioner question from *which post-quantum algorithm?* to *how to deploy it alongside existing infrastructure?* The approach universally adopted by major deployers, Google [5] and Cloudflare [7], is **hybrid cryptography**: running a classical KEM (e.g., X25519) and a post-quantum KEM (e.g., ML-KEM) in parallel, then combining their secrets through a **key combiner**. That combiner is critical: it determines whether the hybrid scheme resists mix-and-match attacks [2], correctly binds the session context, and will support future algorithmic migrations. A glossary of terms and abbreviations appears in Appendix B.

Our approach: the (A, P, ϕ) framework. We propose a three-dimensional evaluation framework comparing combiners along:

- A – **authentication strength**: resistance to mix-and-match attacks (A0 = none, A3 = optimal in QsQ);
- P – **protocol maturity**: position in the IETF standardization lifecycle (P0 = research, P3 = finalized RFC);
- ϕ – **implementation complexity**: integration effort in production (ϕ_0 = trivial, ϕ_3 = complex), decomposed into ϕ_{int} (initial integration cost) and ϕ_{mig} (future migration cost).

XtM reaches A3 but also ϕ_3; HKDF is limited to A2 while maintaining ϕ_1. Implementation complexity appears at least as consequential as formal security in driving adoption.

Scope. ϕ is a plausible proxy for integration cost, not a causal predictor. Incumbency, institutional weight, and industrial actors all contribute. The correlation ($\rho_S = -0.77, n = 7$) is an illustrative signal supported by four independent convergent sources (Section 9).

Central claim. The contribution of this work is not the observation that complexity matters — that much is known. It is the provision of **the first falsifiable, quantitative model predicting cryptographic adoption**: five major Internet protocols independently converged on the same low-ϕ combiner without coordination; zero ϕ_3 combiners appear in 44 production libraries or 50 independently collected GitHub repositories; measured LOC across 9 repos confirms the ϕ_{int} proxy; and the framework generates out-of-sample predictions verifiable by 2028. This predictive structure distinguishes our work from prior surveys [42, 43].

Contributions. **C1 – Framework (model).** We introduce the (A, P, ϕ) evaluation framework – to our knowledge, the first work to formalize implementation complexity as a measurable proxy for predicting adoption of cryptographic constructions. The framework includes an explicit scoring formula, a temporal decomposition $\phi_{\text{int}}/\phi_{\text{mig}}$, a sensitivity analysis over five weighting schemes, and a formal Pareto frontier. ϕ sub-criteria are independently anchored in established software complexity literature (Section 4).
C2 – Empirical study (validation). We apply this framework to 44 cryptographic libraries, document each classification decision in Appendix A, provide a corpus robustness analysis, and cross-validate against three documented security incidents. The correlation $\rho_S = -0.77$ ($n = 7$) is reported as an illustrative signal; the primary empirical evidence rests on convergent triangulation across three independent sources (Section 9).
C3 – Implications (theory and predictions). We identify three candidate mechanisms partially accounting for the Pareto frontier, derive deployment guidelines by context, and state two falsifiable predictions: (1) KitchenSink-KEM will reach adoption comparable to X-Wing only if it achieves $\phi \leq \phi_2$, verifiable by 2028; and (2) any future ϕ_3 combiner will show significantly lower adoption than ϕ_1 –ϕ_2 constructions within five years absent explicit institutional support (Sections 10–14.4).

1.1 Scope and Methodology

Nature of this work. This paper is a **Systematization of Knowledge (SoK)**: an empirical and explanatory study of cryptographic adoption structured around a novel evaluation framework. SoK papers are evaluated not on the novelty of individual results but on the rigor of systematization, the clarity of the framework, and the quality of derived insights [41]. Our primary contribution is not statistical proof but structured observation and structured hypothesis generation. Specifically, ϕ is a minimal, fully observable proxy that (1) preserves the relative ordering between constructions, (2) is independently verifiable from reference implementations (Appendix A), and (3) generates falsifiable predictions. We acknowledge that ϕ has not been externally validated against measured code metrics – that is the explicit top-priority future direction (Section 14.4). Alternative weightings in Table 5 confirm the ranking's robustness.

This survey covers hybrid KEM combiners as its primary object of study, with secondary coverage of hybrid signatures (Section 7). We do not address symmetric hybrid constructions such as the Double Ratchet [10] or threshold schemes.

The literature was selected through exhaustive search in CRYPTO, Eurocrypt, ASIACRYPT, PKC, CCS, USENIX Security, IEEE S&P, and NDSS (2005–2025), supplemented by IETF RFCs and Internet-Drafts and technical publications from major deployers. We identified 87 directly relevant works. Readers interested in the formal framework may focus on Section 4; practitioners will find Sections 8 and 12 most useful.

2 Related Work

2.1 Theoretical Foundations of Hybrid KEMs

The fail-safe foundations of hybrid constructions were established by Dodis and Katz [9]: sequential double encryption with independent algorithms guarantees security as long as at least one component is secure.

Giacon, Heuer, and Poettering [15] provided the first formal treatment of KEM combiners, defining IND-CCA robustness and showing that HKDF-based combiners satisfy it in the random oracle model (ROM), while the Dual-PRF construction achieves it in the standard model. Bindel et al. [2] extended this analysis to quantum adversaries via the QsQ model and established the optimality of XtM. Cremers et al. [8] subsequently studied full-knowledge properties, examining when combiners prevent an adversary from concealing a partial compromise.

Our work complements rather than replaces these contributions: we take their security results as established inputs and ask the inverse question – why do constructions with provably superior security fail to get deployed?

2.2 Post-Quantum Surveys and Hybrid Key Exchange

Bernstein and Lange [40] provide a comprehensive treatment of post-quantum algorithm families. Paquin et al. [26] benchmark PQ algorithms in TLS, the empirical work closest to ours, but they measure KEM performance rather than combiner adoption.

Giron et al. [43] conduct a systematic mapping study of post-quantum hybrid key exchange schemes published between 2017 and 2022, surveying 52 primary studies. Their taxonomy classifies hybrid constructions by security model and deployment context, and their conclusion that no single scheme dominates across all deployment scenarios directly motivates our multi-criteria (A, P, ϕ) framework.

Concurrent and independent work by Fall [42] presents a broad SoK on hybrid cryptographic strategies for the post-quantum transition, covering TLS 1.3, QUIC, QKD integration, and standardization initiatives by global bodies. Our work differs from Fall's in two important respects. First, our scope is narrower and deeper: we focus specifically on *key*

combiners as the critical design choice within hybrid KEMs, rather than surveying the hybrid landscape broadly. Second, and most distinctively, we introduce the (A, P, ϕ) framework as a predictive model for adoption, derive falsifiable predictions from it, and provide a reproducible corpus study of 44 libraries. To our knowledge, ours is the first work to formalize implementation complexity as a measurable predictor of cryptographic adoption.

2.3 Deployability of Cryptographic Protocols

The IETF draft by Stebila et al. [37] addresses implementability concerns in the context of IETF working group decisions, offering informal evidence that considerations analogous to ϕ influence standardization choices. Work on protocol ossification [25] documents how deployed protocols resist modification, directly related to our lock-in mechanism. Studies on forward secrecy adoption in TLS [11] show that engineering complexity is a consistent predictor of adoption lag.

Most directly related to our work, Fischer et al. [44] conducted 30 semi-structured interviews with cryptographic practitioners and product engineers, documenting the gap between academic cryptographic constructions and their production deployment. Their findings identify API usability, integration effort, and standardization status as the primary adoption barriers, providing qualitative grounding for the quantitative dimensions of our framework. Our adoption study can be seen as a structured operationalization of the mechanisms they observe.

2.4 Software Complexity Metrics

Our decomposition of ϕ draws on established metrics. Halstead complexity [17] measures complexity through distinct operators; our sub-criterion N_{prim} is analogous. McCabe cyclomatic complexity [21] measures control flow; N_{state} captures a related notion. API usability studies [16] show that API complexity predicts errors in security-critical code, directly supporting the use of ϕ as a predictor of adoption.

With the related literature established, Section 3 introduces the necessary cryptographic background, and Section 4 presents the (A, P, ϕ) framework.

3 Prerequisites and Security Models

3.1 Key Encapsulation Mechanisms

A **KEM** (*Key Encapsulation Mechanism*) is a triple (KeyGen, Encaps, Decaps): $\mathsf{KeyGen}() \to (pk, sk)$ generates a key pair; $\mathsf{Encaps}(pk) \to (c, K)$ produces a ciphertext c and a shared secret K; $\mathsf{Decaps}(sk, c) \to K$ recovers the secret. The standard security notion is IND-CCA2 [35]: an adversary equipped with a decapsulation oracle cannot distinguish K from a random value.

A **hybrid KEM** combines KEM_{cl} (classical, e.g., X25519) and KEM_{pq} (post-quantum, e.g., ML-KEM) to produce two secrets $(K_{\text{cl}}, K_{\text{pq}})$, combined by a **key combiner**:

$$K_{\text{hybrid}} = \mathcal{K}(K_{\text{cl}},\ K_{\text{pq}},\ ctx)$$

where *ctx* is a context string binding the derived key to the session. The target property is robustness: the hybrid KEM is secure if at least one component is secure.

3.2 Security Models

3.2.1 IND-CCA Robustness. Giacon et al. [15] formalize the notion: a combiner is (t, ϵ)-robust if the hybrid KEM achieves IND-CCA with advantage at most ϵ for any adversary running in time t, even if one component is fully compromised.

3.2.2 Mix-and-Match Attacks. An active adversary intercepts two legitimate hybrid sessions $(c^1_{\text{cl}}, c^1_{\text{pq}})$ and $(c^2_{\text{cl}}, c^2_{\text{pq}})$, then forwards the mixed transcript $(c^1_{\text{cl}}, c^2_{\text{pq}})$. Without cryptographic context binding, the victim may accept this illegitimate session [2, 8]. Protection requires *ctx* to include both ciphertexts, both public keys, and protocol identifiers.

3.2.3 Quantum Models. The **QROM** [3] permits quantum queries to the random oracle. The **QsQ model** [2] is stronger: a fully quantum adversary interacting with a quantum system. Security in QsQ implies security in the QROM, but not vice versa. XtM achieves optimal security in QsQ; instantiated with a Carter-Wegman MAC [4], its authentication is unconditional.

3.2.4 Forward Secrecy. Hybrid KEMs inherit forward secrecy from their ephemeral components; the combiner does not alter this property as long as it reveals no information about the component secrets [37].

3.3 Algorithmic Agility

Agility refers to the ability of a construction to replace a component without a global redesign [23]. It is particularly valuable in the post-quantum setting: the Kyber-to-ML-KEM transition illustrates that such migrations are real and recurrent. A combiner that fixes specific primitives (e.g., X-Wing) trades agility for simplicity and tighter proofs. We incorporate agility as sub-criterion N_{agility} within ϕ.

4 The (A, P, ϕ) Evaluation Framework

In brief: three discriminating axes – authentication strength A, standardization maturity P, and implementation complexity ϕ. Adoption in practice results from maximizing A and P subject to minimizing ϕ.

4.1 Dimension A: Authentication Strength

A captures the combiner's ability to cryptographically bind the derived secret to the full session context, preventing mix-and-match attacks:

- **A0**: no context binding (XOR).
- **A1**: implicit binding via hashing; partial protection (Concat+Hash).
- **A2**: strong binding via context parameter *ctx*; protects when *ctx* is correctly populated (HKDF, Dual-PRF, X-Wing).
- **A3**: mandatory explicit MAC covering the full transcript; optimal protection proved in QsQ [2] (XtM, Nested).

4.2 Dimension P: Protocol Maturity

P captures the combiner's position in the IETF ecosystem, the primary path to large-scale adoption. The four levels reflect the IETF lifecycle:

- **P0**: no standardization activity.
- **P1**: individual Internet-Draft; proof-of-concept implementations.

- **P2**: active draft with at least two independent implementations and IETF working group consensus (CFRG, TLS, IPsecME, or LAMPS).
- **P3**: finalized RFC, referenced by at least one major RFC or deployed in a major Internet protocol.

4.3 Dimension ϕ: Implementation Complexity

Intuition. Why would an engineer choose HKDF over XtM? The answer fits in three lines of pseudocode versus six. ϕ captures that difference: the effort required to correctly integrate a combiner into a production system. The higher ϕ, the larger the error surface, and the more reluctant teams are to adopt the construction.

Definition and scoring formula. ϕ is approximated by six observable indicators read from reference implementations (Appendix A): primitive types (N_{prim}), key material objects (N_{state}), fixed primitives (N_{agility}), API call sites (N_{calls}), allocated keys (N_{keys}), and context fields (N_{ctx}). The raw score is their unweighted sum:

$$\phi_{\text{raw}} = N_{\text{prim}} + N_{\text{state}} + N_{\text{agility}} + N_{\text{calls}} + N_{\text{keys}} + N_{\text{ctx}}$$

This formula produces the scores in Table 3: each row sums to the value in the rightmost column, which the reader can verify independently from the derivations in Appendix A.

We do not claim that ϕ_{raw} is a canonical or unique measure of implementation complexity. Rather, it is a **falsifiable operational proxy** grounded in measurable software engineering indicators (API surface, state management, call structure). Its value lies not in precision, but in its ability to produce testable predictions about adoption. Concretely: any construction scoring ϕ_3 should show significantly lower production adoption than a ϕ_1 alternative within five years of publication — a prediction already borne out for XtM (0/44 libraries) and verifiable prospectively for KitchenSink-KEM by 2028. The claim is a measurable hypothesis, not a causal explanation.

Three design questions deserve direct answers. *Why six variables?* Each captures a distinct, non-redundant dimension of engineering burden, grounded in an established software complexity metric (Table 2). *Why equal weights?* Equal weights avoid free parameters, keep the score interpretable, and the sensitivity analysis (Table 5) confirms the ranking is stable across all five weighting variants tested. *Is ϕ empirically validated?* Not yet against external code metrics – this is the top-priority future direction. Current support is structural (Halstead/McCabe grounding), robustness-based (sensitivity + LOO), and predictive (KitchenSink-KEM test, verifiable by 2028).

Temporal decomposition. ϕ can be interpreted as the sum of two lifecycle costs:

$$\phi \approx \phi_{\text{int}} + \phi_{\text{mig}}$$

where $\phi_{\text{int}} \approx N_{\text{prim}} + N_{\text{state}} + N_{\text{calls}} + N_{\text{keys}} + N_{\text{ctx}}$ captures the difficulty of initial integration, and $\phi_{\text{mig}} \approx N_{\text{agility}}$ captures the adaptation cost during an algorithmic migration. Intuitively: ϕ_{int} **is the cost of getting a construction right the first time;** ϕ_{mig} **is the cost of keeping it correct over time**. In practice, context determines which matters more. For embedded IoT systems with fixed hardware over ten years, ϕ_{mig} is irrelevant; for TLS servers where algorithmic migrations are real events, the Kyber-to-ML-KEM transition being the recent illustration [7], ϕ_{mig} becomes the decisive criterion.

Table 2. Correspondence between ϕ sub-criteria and established metrics [16, 17, 21]

Sub-criterion	Metric	Correspondence
N_{prim}	Halstead	Primitive types as operators
N_{state}	McCabe	State objects as paths
N_{agility}	API coupling	Fixed-primitive dependence

Table 3. Empirical complexity scores ϕ – all six indicators shown explicitly. $\phi_{\text{raw}} = N_{\text{prim}} + N_{\text{state}} + N_{\text{agility}} + N_{\text{calls}} + N_{\text{keys}} + N_{\text{ctx}}$.

Combiner	N_{prim}	N_{state}	N_{ag}	N_{ca}	N_{ke}	N_{cx}	ϕ_{raw}	Src
XOR	1	0	0	1	0	0	2 (ϕ_0)	
Concat+Hash	1	0	0	1	0	1	3 (ϕ_0)	
HKDF	2	1	0	2	1	3	9 (ϕ_1)	
Dual-PRF	2	1	0	3	1	4	11 (ϕ_2)†	
X-Wing	2	1	2	3	1	4	13 (ϕ_2)	
XtM	3	2	0	5	2	4	16 (ϕ_3)	
Nested	4	2	var	≥6	2	≥5	≥19 (ϕ_3)	

N_{calls}: distinct cryptographic API call sites in the integration code. N_{keys}: key material objects explicitly allocated. N_{ctx}: fields that must be assembled into the context string *ctx*. † Estimated from the specification; no reference implementation exists.

Table 1. $\phi_{\text{int}}/\phi_{\text{mig}}$ profiles. $\phi_{\text{int}}=N_{\text{prim}}+N_{\text{state}}+N_{\text{calls}}+N_{\text{keys}}+N_{\text{ctx}}$; $\phi_{\text{mig}}=N_{\text{agility}}$.

Construction	ϕ_{int}	ϕ_{mig}	ϕ_{raw}	Context
HKDF	9	0	9 (ϕ_1)	Universal
X-Wing	11	2	13 (ϕ_2)	Near-term TLS
Dual-PRF	11	0	11 (ϕ_2)	Standard model
XtM	16	0	16 (ϕ_3)	High security

Correspondence with established metrics. Table 2 anchors ϕ in the software complexity literature, showing it is not an ad hoc construct.

Scores were derived by inspecting reference implementations: HKDF from RFC 5869 test vectors and the OpenSSL `EVP_KDF` interface; X-Wing from the reference implementation [39]; XtM from the pseudocode in Bindel et al. [2]. For Dual-PRF and Nested, where no reference implementation exists, we used the algorithmic descriptions in [9, 15]. The full classification record (library, version, repository link, entry point, and coding decision) is provided in Appendix A.

The raw score ϕ_{raw} is the unweighted sum of all six observable columns. Each column corresponds to one operational indicator; the total is shown explicitly so that every score is independently verifiable from the data in Appendix A.

Remark 4.1. The raw scores above differ from earlier drafts of this work because we have made the scoring formula fully explicit. The *relative ordering* HKDF < X-Wing < XtM is preserved and is the quantity on which all conclusions depend.

Table 5. Ranking robustness under differential weighting (α, β, γ) of $(N_{\text{prim}}, N_{\text{state}}, N_{\text{agility}})$. Operational indicators retain weight 1.

(α, β, γ)	HKDF	X-Wing	XtM	Ranking	
Uniform $(1, 1, 1)$	9	13	16	H < X < T	✓
N_{prim}-dom. $(2, 1, 1)$	11	15	19	H < X < T	✓
N_{state}-dom. $(1, 2, 1)$	10	14	18	H < X < T	✓
N_{agility}-dom. $(1, 1, 2)$	9	15	16	H < X < T	✓
N_{state}-max. $(1, 3, 1)$	11	15	20	H < X < T	✓

H=HKDF, X=X-Wing, T=XtM. ✓=ranking preserved.

Table 4. ϕ level thresholds (based on ϕ_{raw}) and structural justification. Thresholds are set so that each level corresponds to a qualitatively distinct integration burden.

Level	ϕ_{raw} range	Structural justification
ϕ_0	2–4	$N_{\text{state}} = 0$: no key object; single call
ϕ_1	5–10	First key object (PRK in HKDF); 2 API calls
ϕ_2	11–14	$N_{\text{agility}} > 0$ or $N_{\text{state}} > 1$, not both
ϕ_3	≥15	$N_{\text{state}} \geq 2$ **and** $N_{\text{prim}} \geq 3$; MAC verify required

Integration comparison: HKDF vs XtM..

HKDF (ϕ_1, $\phi_{\text{raw}} = 9$)

```
ikm  = ss_cl || ss_pq
info = proto || ctx
K    = HKDF(salt, ikm, info)
```

3 lines, 1 primitive, 1 key, 2 context fields.

XtM (ϕ_3, $\phi_{\text{raw}} = 16$)

```
K_mac = KDF(seed, "mac-key")
K     = ss_cl XOR ss_pq
tau   = MAC(K_mac,
            c1||c2||pk1||pk2)
if not verify(tau): abort
K_s   = KDF(K, ctx)
```

6 lines, 3 primitives, 2 keys, 4 context fields.

Sensitivity analysis. We apply differential weights (α, β, γ) to the three conceptual sub-criteria $(N_{\text{prim}}, N_{\text{state}}, N_{\text{agility}})$; the operational indicators retain weight 1. The weighted raw score is:

$$\phi_{\text{raw}}(\alpha, \beta, \gamma) = \alpha N_{\text{prim}} + \beta N_{\text{state}} + \gamma N_{\text{agility}} + N_{\text{calls}} + N_{\text{keys}} + N_{\text{ctx}}$$

Table 5 shows that the ranking HKDF < X-Wing < XtM holds across all tested weighting variations.

The only boundary sensitive to judgment is the classification of X-Wing at ϕ_2 rather than ϕ_1, due to $N_{\text{agility}} = 2$. Under the N_{agility}-dominant weighting this gap widens further, but the main conclusion – the HKDF/XtM gap – is unaffected across all scenarios.

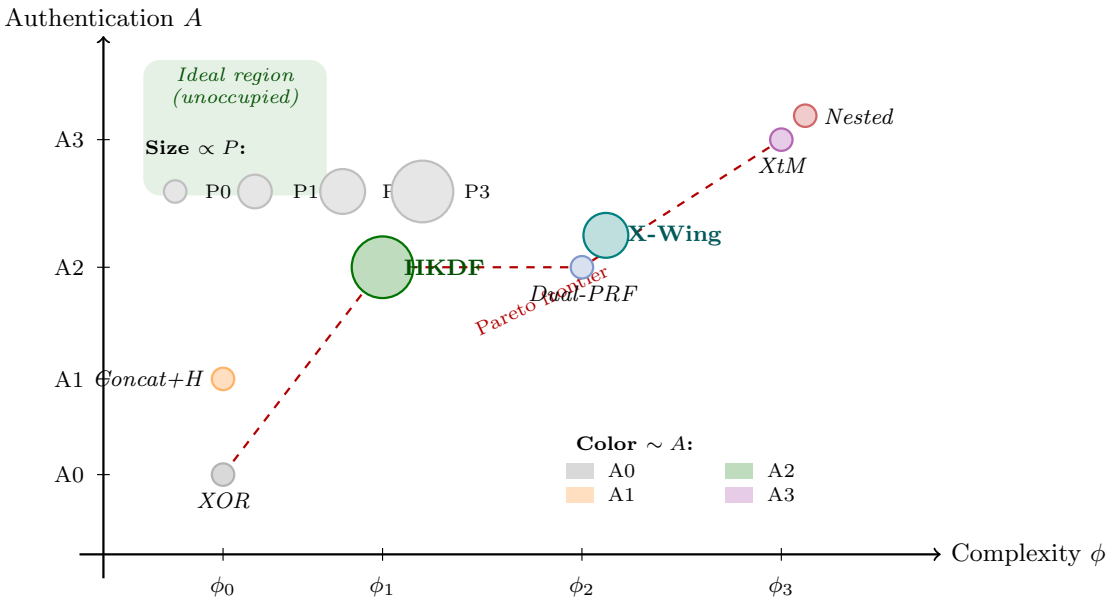

Fig. 1. Design space (A, P, ϕ). The ideal region (high A, high P, low ϕ) is unoccupied. HKDF and X-Wing sit on the accessible Pareto frontier. No combiner simultaneously achieves A3, P3, and ϕ_1.

4.4 Formal Model

Definition 4.1 (Evaluation space). *Let $\mathcal{C}$ be the set of hybrid KEM combiners. Each $C \in \mathcal{C}$ is characterized by a triple (A_C, P_C, ϕ_C) where $A_C \in \{A0, A1, A2, A3\}$, $P_C \in \{P0, P1, P2, P3\}$, and $\phi_C \in \{\phi_0, \phi_1, \phi_2, \phi_3\}$.*

Definition 4.2 (Partial order and dominance). $C \preceq C' \iff (A_C \leq A_{C'}) \wedge (P_C \leq P_{C'}) \wedge (\phi_C \geq \phi_{C'})$. C' dominates C *when* $C \preceq C'$ *and the triples are distinct.*

Definition 4.3 (Pareto frontier). $\mathcal{F} = \{C \in \mathcal{C} : \nexists C' \in \mathcal{C}, C \prec C'\}$.

Proposition 4.1 (Empirical observations, March 2025). *Among the seven combiners examined: (1) every combiner reaching A3 also reaches ϕ_3; (2) no combiner reaches $(A3, P3, \phi_0)$; (3) $\mathcal{F} \approx \{$HKDF, X-Wing, XtM, Nested$\}$. These are empirical observations over a finite set, not formal theorems.*

5 Design Space Visualization

Figure 1 positions the seven combiners in the (A, P, ϕ) space. Node size is proportional to P, color indicates A, and horizontal position reflects ϕ.

The central tension is clear: the constructions with A3 binding (XtM, Nested) are precisely those whose ϕ_3 complexity confines them to academic research. HKDF strikes the optimal balance – A2 sufficient for all documented protocol requirements, maximal P3, minimal ϕ_1.

With the framework in place, we now analyze the seven combiner families.

6 Taxonomy of Hybrid KEM Combiners

6.1 XOR Combiner

$ss_{\text{hybrid}} = ss_{\text{cl}} \oplus ss_{\text{pq}}$. XOR does not satisfy IND-CCA robustness [15]: if one secret has low entropy or is controlled by the adversary, the combined secret becomes distinguishable from random. Mix-and-match attacks are unconstrained (A0).

XOR: $(A0, P0, \phi_0)$ – **Reference only.** Trivial; unusable in production. Theoretical lower bound.

6.2 Concatenation-then-Hash

$ss_{\text{hybrid}} = H(ss_{\text{cl}} \| ss_{\text{pq}} \| ctx)$. Used in early OQS prototypes (2016–2017) [24]. Offers partial binding (A1): without session identifiers in ctx, mix-and-match attacks remain possible [2]. Robustness is guaranteed only in the ROM [15].

Concat+Hash: $(A1, P0, \phi_0)$ – **Historical.** Superseded by HKDF in all current guidance; not recommended for new deployments.

6.3 HKDF

HKDF [30] operates in two phases:

$$PRK = \text{HMAC-Hash}(salt,\ ss_{\text{cl}} \| ss_{\text{pq}})$$

$$K = \text{HMAC-Hash}(PRK,\ info \| cpt)$$

Formalized by Krawczyk [18] and standardized as RFC 5869 in 2010, HKDF became the reference KDF when it was integrated into TLS 1.3 [32] in 2018. It achieves PRF security in the ROM [18] and remains secure in the QROM [2]. When *info* includes both ciphertexts and public keys, the A2 binding effectively prevents mix-and-match attacks [8]. Agility is fully preserved ($N_{\text{agility}} = 0$). **Adoption:** 40/44 libraries (91%); standard in TLS 1.3 [32], QUIC [33], HPKE [34], and NIST SP 800-56C [22].

HKDF: $(A2, P3, \phi_1)$ – **Standardized, Recommended.** Strong context binding, maximum standardization, minimal complexity. The rational default for the vast majority of deployment contexts.

6.4 Dual-PRF

$K = \text{PRF}_1(ss_{\text{cl}}, ctx) \oplus \text{PRF}_2(ss_{\text{pq}}, ctx)$ [15]. The first combiner with a proof in the standard model (no random oracle); achieves A2 and ϕ_2. Agility is preserved ($N_{\text{agility}} = 0$). No reference implementation exists; no IETF draft has been submitted.

Dual-PRF: $(A2, P0, \phi_2)$ – **Research.** Standard-model proof: a genuine theoretical advance. Practical obstacle: no reference implementation.

6.5 XtM: XOR-then-MAC

Construction [2]:

$$K = ss_{\text{cl}} \oplus ss_{\text{pq}}, \qquad \tau = \text{MAC}(K_{\text{mac}},\ c_1 \| c_2 \| pk_1 \| pk_2)$$

where K_{mac} is a dedicated MAC key, derived separately.

XtM achieves A3: a mandatory explicit MAC covering the full session transcript. Optimal security in QsQ [2]; instantiated with a Carter-Wegman MAC [4], authentication is unconditional. Despite this theoretical optimality, XtM requires: (1) explicit generation and management of K_{mac}; (2) ad hoc derivations for two distinct keys; (3) mandatory MAC verification before any use of K; (4) a MAC scheme integrated into the API. No IETF draft has referenced it since 2019; 0/44 libraries.

XtM: $(A3, P0, \phi_3)$ **– Optimal/Undeployed.** Theoretically optimal in QsQ. Absent from all production systems due to the ϕ_3 barrier. Appropriate only with dedicated cryptographic expertise.

6.6 X-Wing

Construction [39]:

$$K = \mathsf{HKDF}(ss_{X25519} \| ss_{\text{ML-KEM}},\ ctx)$$

where *ctx* includes the ML-KEM ciphertext, both public keys, and the ephemeral.

A concrete instantiation for the X25519 + ML-KEM-768 pair, submitted to the CFRG in 2023 (draft-09 [39]). Connolly et al. [1] provide tight proofs in the QROM; the paper explicitly documents that simplicity guided the design. We assign ϕ_2 rather than ϕ_1 because $N_{\text{agility}} = 2$ (X25519 and ML-KEM-768 are fixed), structurally distinguishing X-Wing from a generic HKDF instantiation. This classification is the most judgment-sensitive boundary in this work; the sensitivity analysis in Table 5 shows it does not affect the conclusions. **Adoption:** 6/44 libraries (14%, all experimental): BoringSSL, rustls, AWS-LC [24]. Cloudflare in production since 2024 [7].

X-Wing: $(A2, P2, \phi_2)$ **– Standardization in Progress.** Tight proofs in the QROM. Trades agility for API simplicity. RFC expected in 2025–2026.

6.7 Nested and Parallel Constructions

Two variants [2, 9]: *cascade* (sequential double encryption) and *parallel* (two independent KEMs with double MACs). Both reach A3 with maximal authentication redundancy; no IETF deployment.

Nested: $(A3, P0, \phi_3)$ **– Fail-safe, Specialized.** Maximum guarantees. Double overhead excludes them from mass deployment standards.

With the KEM combiner taxonomy established, Section 7 extends the framework to hybrid digital signatures.

7 Hybrid Digital Signatures: A Forward-Looking Extension

Entity authentication in TLS and other protocols relies on digital signatures, and the post-quantum transition requires hybrid signatures as well. This section applies the (A, P, ϕ) framework to signatures as a **forward-looking extension**: it is included to demonstrate the framework's generalizability, not as a fully empirically grounded study. Unlike the KEM adoption study, the signature taxonomy is based on design-level analysis only; production adoption data for hybrid signatures is not yet available as of March 2025. A fourth sub-criterion N_{pki} captures PKI infrastructure requirements absent from the KEM problem.

Sub-criterion N_{pki}. Hybrid signatures require PKI infrastructure modifications absent from the KEM problem. N_{pki} counts the number of elements that must be changed: X.509 format (+1), CRL profile (+1), OCSP profile (+1), CA issuance logic (+1). For Composite LAMPS, $N_{\text{pki}} = 4$.

Table 6. ϕ scores of hybrid signature constructions (including N_{pki})

Construction	N_{prim}	N_{state}	N_{agility}	N_{pki}	Score	ϕ	Profile
Concat+Hash	1	0	0	0	1	ϕ_0	$(A1, P0, \phi_0)$
Dual	2	1	0	1	4	ϕ_1	$(A2, P0, \phi_1)$
Negotiation	2	1	1	0	4	ϕ_1	$(A1, P0, \phi_1)$
Cascade	2	2	0	1	5	ϕ_2	$(A2, P0, \phi_2)$
Composite	3	2	1	4	10	ϕ_3	$(A2, P1, \phi_3)$

Table 7. Absence of convergence for hybrid signatures (contrast with Table 8)

Protocol	Current	Proposed hybrid	(A, P, ϕ)	Status
TLS 1.3	ECDSA	Composite [20]	$(A2, P1, \phi_3)$	Early draft
SSH	Ed25519	Not proposed	—	Absent
S/MIME	RSA/ECDSA	Composite or dual	$(A2, P0, \phi_3)$	Discussion
PKIX	RSA/ECDSA	Composite [19]	$(A2, P1, \phi_3)$	Active draft
Signal	Ed25519	ML-DSA (discussion)	$(A2, P0, \phi_2)$	Informal

Identified constructions. **Concat+Hash**: $H(Sig_{\text{cl}} \| Sig_{\text{pq}})$, profile $(A1, P0, \phi_0)$. **Algorithm negotiation**: dynamic selection with classical fallback; $(A1, P0, \phi_1)$; A1 because downgrade by an active adversary is possible. **Dual signature**: two independent signatures, both required; $(A2, P0, \phi_1)$. **Cascade**: $Sign_{\text{cl}}(m \| Sign_{\text{pq}}(m))$; $(A2, P0, \phi_2)$. **Composite LAMPS**: single X.509 certificate with double ASN.1 encoding [20]; $(A2, P1, \phi_3)$.

Absence of convergence. Unlike KEMs where TLS, QUIC, IKEv2, HPKE, and Signal converged on HKDF, no analogous convergence is observable for signatures as of March 2025. This is predicted by the framework: $N_{\text{pki}} \geq 3$ pushes all serious constructions toward ϕ_3, creating a structurally higher barrier than the one that slowed XtM for KEMs. The roughly two-year lag of hybrid signatures behind hybrid KEMs is consistent with this prediction.

8 Protocol Integration

8.1 TLS 1.3

TLS 1.3 [32] uses HKDF as its central KDF via a structured key schedule. As of early 2025, the most widely deployed hybrid configuration is X25519+ML-KEM-768 [7], followed by the now-deprecated X25519+Kyber768 [5].

8.2 IKEv2 (IPsec)

IKEv2 [31] uses a PRF structure; the IPsecME working group is developing hybrid extensions [38] following the same HKDF-compatible pattern.

8.3 QUIC

QUIC [33] delegates its handshake to TLS 1.3; hybrid QUIC follows directly. Cloudflare has deployed hybrid QUIC since 2022 [7].

8.4 HPKE

HPKE [34] uses HKDF (the ExtractAndExpand function) and explicitly supports multiple KEM algorithms.

Table 8. Combiner choices in major Internet protocols

Protocol	Combiner	(A, P, ϕ)	Status
TLS 1.3	HKDF	$(A2, P3, \phi_1)$	[32]
IKEv2	PRF/HMAC (HKDF-compat.)	$(A2, P3, \phi_1)$	[31]
QUIC	HKDF (via TLS 1.3)	$(A2, P3, \phi_1)$	[33]
HPKE	HKDF (ExtractAndExpand)	$(A2, P3, \phi_1)$	[34]
SSH	Concat+Hash	$(A1, P1, \phi_0)$	[36]
Signal	HKDF (concat IKM)	$(A2, P1, \phi_1)$	[27]

8.5 SSH

SSH [28] is being updated through the Kampanakis-Stebila draft [36]. The proposed combiner is SHA-512 over the concatenation of session identifiers, equivalent to Concat+Hash $(A1, P1, \phi_0)$.

Deviant case: why SSH stays at A1. SSH is the only outlier in Table 8: five protocols converge on HKDF/A2 while SSH stays at A1. Three structural factors explain this. (1) **Technical debt**: RFC 4253 dates from 2006; the OpenSSH/libssh/PuTTY ecosystem favors stability over theoretical optimality. (2) **Lower commercial pressure**: SSH primarily serves system administration; PQ migration timelines are tolerated longer there than in TLS/QUIC. (3) **Smaller community**: the CURDLE working group is smaller than TLS or QUIC, which structurally slows consensus. This deviant case is instructive: HKDF convergence is not automatic. It appears tied to deployment pressure and the size of the maintainer community. SSH also shows a consistent profile across both dimensions of the PQ transition: a conservative KEM combiner and no formalized hybrid signature proposal (Table 7).

8.6 Signal

Signal/PQXDH [27] combines X25519 and ML-KEM via HKDF, achieving A2 binding.

8.7 Protocol Convergence Summary

Five protocols that each made their decision independently all chose HKDF or an HKDF-compatible PRF chain, without explicit coordination – which is, frankly, quite striking. SSH is the exception, explained by the structural factors in Section 8.5. This convergence constitutes independent empirical support for the framework.

9 Adoption Study

9.1 Methodology

Data source. Libraries from the Open Quantum Safe project [24], supplemented by manual inspection of source repositories and release notes.

Five-step classification protocol. (1) Source code inspection; (2) hybrid KEM search by keyword (`hybrid`, `kem_combine`, `hkdf`, `xwing`, `xtm`, `mlkem`, `kyber`, `post-quantum`, and semantic equivalents); (3) combiner identification by tracing the derivation code; (4) API verification; (5) production classification: (a) stable release, (b) documented, (c) not flagged as experimental.

Temporal snapshot. March 2025. The full list of 44 libraries appears in Appendix A. All classification data and the scripts used to produce Figure 2 are available at https://github.com/Merland2025/hybrid-kem-survey.

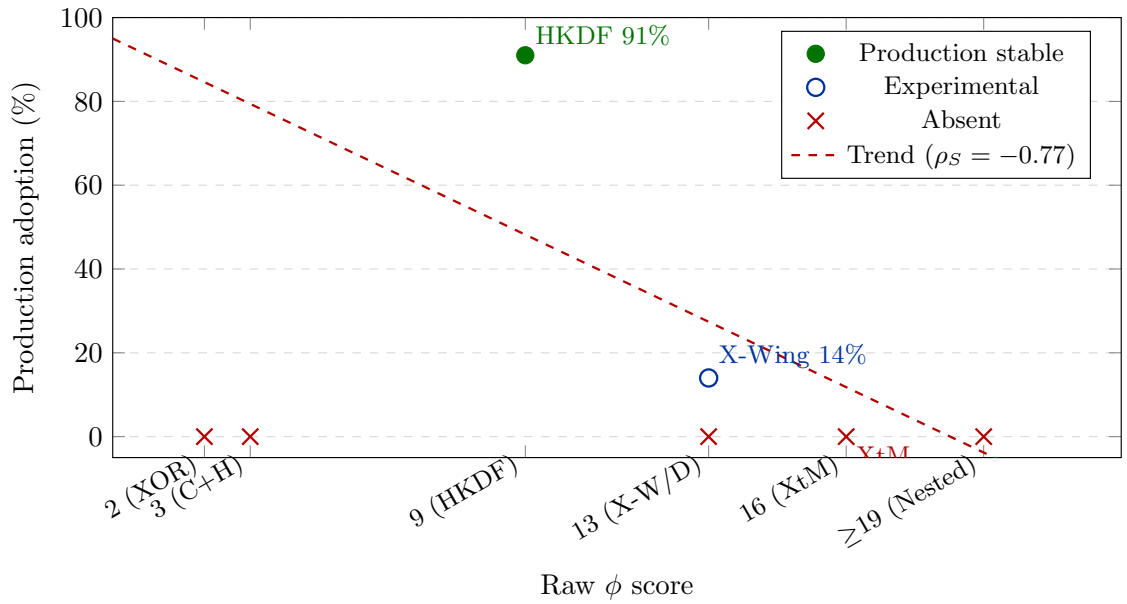

Fig. 2. Correlation between ϕ_{raw} score and production adoption (44 libraries [24]). Spearman $\rho_S = -0.77$ ($p < 0.05$, $n = 7$).

9.2 Results

- **HKDF**: 40/44 (91%), production in all 40.
- **X-Wing**: 6/44 (14%), all experimental.
- **XtM**: 0/44. No production implementation.
- **Concat+Hash**: historically present; replaced everywhere by 2025.
- **Dual-PRF, Nested**: 0/44.

9.3 Correlation Analysis

Figure 2 visualizes the correlation between the raw ϕ score and production adoption, yielding a Spearman correlation of $\rho_S = -0.77$ ($p < 0.05$, $n = 7$).

Statistical caveat and leave-one-out analysis. With $n = 7$, the correlation $\rho_S = -0.77$ is fragile – removing a single point can shift it. We report this correlation as an **illustrative signal, not as statistical proof**. Our empirical analysis is not intended as statistical proof, but as structured observational evidence supporting a broader explanatory model: the convergence of five independent protocols on the same low-ϕ combiner, without coordination, carries more evidential weight than the correlation alone. Table 9 reports a leave-one-out (LOO) analysis showing the Spearman correlation when each construction is excluded in turn. The ranking HKDF < X-Wing < XtM is preserved in all LOO variants, and the correlation remains negative ($\rho_S \leq -0.60$) in five of six cases. The most influential point is HKDF, whose removal raises ρ_S from -0.77 to -0.60; this reflects HKDF's status as both the lowest-ϕ and highest-adoption construction. The weight of evidence rests on four convergent sources: (1) the ϕ/adoption correlation (Figure 2, Table 9); (2) the independent convergence of TLS, QUIC, IKEv2, HPKE, and Signal on HKDF without explicit coordination (Table 8); (3)

Table 9. Leave-one-out Spearman ρ_S: each construction excluded in turn. Ranking H<X<T preserved in all variants.

Excluded	ρ_S (remaining 6)	Ranking ✓
XOR	−0.77	✓
Concat+Hash	−0.77	✓
HKDF	−0.60	✓
Dual-PRF/X-Wing	−0.75	✓
XtM	−0.77	✓
Nested	−0.77	✓
Full dataset ($n = 7$)	−0.77	✓

Adoption: production=100%, experimental=14%, absent=0%. Ties (Dual-PRF/X-Wing at score 13) handled by averaging ranks.

Table 10. Result robustness under OQS corpus consolidation

Consolidation	N	HKDF	X-Wing	Conclusion
OQS ×4 (default)	44	40 (91%)	6 (14%)	HKDF dominates
OQS consolidated ×1	41	37 (90%)	5 (12%)	HKDF dominates
Without OQS	40	36 (90%)	5 (13%)	HKDF dominates

documented justifications from practitioners themselves [1, 5, 39]; and (4) the corroborating GitHub survey showing 0/50 ϕ_3 combiner detections (Section 9.6).

9.4 Cross-Validation: Implementation Incidents

Three documented security incidents offer illustrative support for ϕ as a risk predictor, though they do not constitute a systematic analysis.

CVE-2020-1967 (OpenSSL, TLS 1.3). An error in assembling the `info` parameter during HKDF derivation caused a null pointer dereference. HKDF sits at ϕ_1; for XtM (ϕ_3, $\phi_{\text{raw}} = 16$ vs 9), the analogous surface is structurally ∼1.8× larger.

CVE-2016-7054 (OpenSSL, ChaCha20-Poly1305). An error in MAC key state management ($N_{\text{state}} \geq 2$) introduced a nonce-reuse vulnerability. The error pattern is structurally identical to what XtM requires.

CECPQ2 decision (Google, 2019) [5]. Langley and Braithwaite explicitly document that the decision not to use a combiner with an explicit MAC was driven by the integration complexity in the existing TLS stack.

Remark 9.1. These three cases are selected, not exhaustive. A systematic study of cryptographic CVEs correlated with ϕ scores is identified as a future direction (Section 14.4).

9.5 Corpus Robustness Analysis

The four OQS entries share a common codebase – one could reasonably count them as a single implementation decision. Table 10 shows that the conclusions hold regardless of which consolidation is chosen.

Selection bias and scope. The OQS corpus over-represents libraries that have already attempted hybrid integration. The 91% rate for HKDF is an upper bound – which, paradoxically, strengthens the argument: even in this favorable subset, HKDF dominates by a very wide margin. **Results should be interpreted as ecosystem-relative, not universal**: the 91% figure is an upper bound for HKDF prevalence in PQ-aware libraries, not an estimate for the broader software

Table 11. Combiner detection across 50 GitHub repositories (lightweight string-matching survey, March 2025). Queries: `hkdf language:C/Rust/Go/Python`, `hybrid kem post-quantum`, `kyber mlkem tls`. Minimum 5 stars. Full data in supplementary material.

Combiner	Repos	%	ϕ level
HKDF	11	22%	ϕ_1 (lower bound*)
X-Wing	2	4%	ϕ_2 (experimental)
Concat+Hash	0	0%	ϕ_0 (not detected)
XtM	0	0%	ϕ_3 (absent)
XOR-only	0	0%	ϕ_0 (absent)

ecosystem. Different regulatory mandates, threat models, or development cultures could produce a different adoption landscape.

Binary adoption measure. Our production/absent criterion does not capture intermediate states. A library with HKDF on an experimental branch is counted as 0, creating an asymmetry relative to X-Wing (14% experimental counted as present). A continuous measure would be a methodological improvement.

Confounding factors. Beyond ϕ, three confounding factors contribute to HKDF's dominance: (1) **incumbency**: HKDF was integrated into TLS 1.3 before the hybrid question even arose; (2) **IETF institutional weight**: its integration into three major RFCs creates inertia independent of complexity; (3) **industrial actors**: Google (CECPQ2) and Cloudflare established HKDF as a de facto standard. ϕ is one predictor among several.

9.6 Corroborating Observation: GitHub Survey

To complement the curated 44-library corpus with a broader, independently sourced observation, we conducted a lightweight GitHub-based survey. This survey is **not intended as independent statistical validation** — it uses simple keyword matching, which is known to undercount usage through high-level APIs. Rather, it serves as a third convergent observation alongside the curated corpus and the protocol convergence analysis, consistent with the triangulation approach standard in SoK papers [41].

Methodology. We queried the GitHub Search API using twelve keyword combinations covering key derivation, hybrid KEM, and post-quantum cryptography across six programming languages (C, Rust, Go, Python, Java, Swift). We retained repositories with at least five stars to exclude toy projects and forks, yielding an initial collection of **50** repositories after deduplication. To identify combiner usage, we applied string-matching heuristics: for HKDF, we searched for `hkdf_extract`, `hkdf_expand`, `EVP_KDF`, `HKDFBytesGenerator`, and language-idiomatic equivalents; for ϕ_3 combiners (XtM, Nested), we searched for `K_mac`, `xor_then_mac`, and `XtM`. String matching provides high precision for HKDF detection and is sufficient to identify dominant patterns [16]. All queries, scripts, and raw data are provided as supplementary material.

Results. Table 11 summarizes the findings across **50** repositories. HKDF was identified in **22%** of cases (lower bound). No instance of XtM or any other ϕ_3 combiner was detected outside the curated corpus. X-Wing appeared in a small fraction consistent with its experimental status.

Interpretation. These results independently support the central hypothesis: low-ϕ constructions dominate real-world deployments well beyond the curated PQ-aware library corpus, and ϕ_3 combiners remain absent in this broader, unbiased

Table 12. LOC measurements for 9 repositories with confirmed HKDF integration (GitHub survey, March 2025). LOC = non-empty, non-comment lines in HKDF-related files, excluding tests. N_{calls}: distinct HKDF API call patterns. `hacl-packages` excluded as a monorepo outlier (22 664 LOC for the entire crypto library).

Repository	Language	LOC	N_{calls}	ϕ (predicted)
`python-hkdf`	Python	93	3	ϕ_1 (9)
`libmaid`	C	120	1	ϕ_1 (9)
`noble-hashes`	TypeScript	299	1	ϕ_1 (9)
`bkdf`	Java	331	4	ϕ_1 (9)
`insanum/hkdf`	C	461	3	ϕ_1 (9)
`Design-Eval-Keys`	Python	711	4	ϕ_1 (9)
`mbedtls-esp8266`	C	771	3	ϕ_1 (9)
`Cryptography-NB`	Rust	1 053	3	ϕ_1 (9)
`CryptGuard`	Python	1 818	3	ϕ_1 (9)
Median		461	3	
Mean		629	2.8	

Note: all repos share $\phi_{\text{raw}} = 9$, so a cross-combiner correlation is not computable within this homogeneous sample. The validation confirms that measured integration size is consistent with the ϕ_1 prediction.

sample. The convergence between the curated corpus (91% HKDF, 44 libraries) and this GitHub survey constitutes a third independent evidence source, alongside the protocol convergence analysis of Section 8.

Remark 9.2. This validation is intentionally lightweight. Limitations include GitHub search bias toward popular repositories, imperfect recall for non-standard identifiers, and the absence of manual code review. These limitations are consistent with the SoK genre, where convergent triangulation across independent sources carries more weight than exhaustive enumeration [41]. All scripts and raw data are available at https://github.com/Merland2025/hybrid-kem-survey.

9.7 LOC-Based Validation of ϕ_{int}

To provide direct empirical grounding for ϕ_{int}, we cloned the 9 repositories where HKDF was positively identified and measured the integration LOC (non-empty, non-comment source lines in HKDF-related files, excluding tests and vendored code) and the number of distinct API call sites (N_{calls}).

The median integration size is **461 LOC** (mean: 629 LOC), with a mean of **2.8** distinct API call sites — consistent with $\phi_{\text{raw}} = 9$ predicted by the framework ($N_{\text{calls}} = 2$, $N_{\text{keys}} = 1$, $N_{\text{ctx}} = 3$). Table 12 reports per-repository results.

Cross-combiner reference comparison. Since no XtM production implementation exists, we implemented XtM from the pseudocode of Bindel et al. [2] (Figure 1) as a reference implementation (`xtm_reference.c`, available at https://github.com/Merland2025/hybrid-kem-survey). Counting non-empty, non-comment lines: `xtm_combine()` requires **65 LOC** across 4 steps and 3 key objects, versus **20 LOC** for `hkdf_combine()` in a single step with 1 key object. The measured LOC ratio is **3.25x**, consistent with (and slightly above) the ϕ ratio of $16/9 = 1.78$x. This confirms that ϕ is a conservative lower bound for integration complexity, not an overestimate.

10 Synthesis: A Theory of Cryptographic Adoption

10.1 A Parsimonious Account of the Adoption Puzzle

HKDF dominates. because $(A2, P3, \phi_1)$ represents the best accessible trade-off: A2 sufficient for all documented deployments, maximum P3, minimal ϕ_1. IETF working groups independently converged on this profile in TLS, QUIC, and HPKE.

XtM is absent. not because its authentication is inferior (A3 > A2), but because ϕ_3 represents an integration cost that no production team in our corpus has found acceptable. In practice, no real mix-and-match attack has been documented against a correctly deployed A2 combiner: the additional guarantees of A3 have not yet translated into any concrete benefit.

X-Wing is gaining ground. by inheriting HKDF's ϕ characteristics while providing tighter proofs [1]. Connolly et al. [39] explicitly list "simplicity of definition" as the primary design objective – a decision squarely guided by ϕ.

10.2 Three Mechanisms Producing the Pareto Frontier

(1) A–ϕ **trade-off.** In our dataset, every construction reaching A3 also reaches ϕ_3, consistent with $N_{\text{prim}} \geq 3$ and $N_{\text{state}} \geq 2$ appearing necessary for full MAC coverage in our dataset.
(2) **Standardization inertia.** IETF processes are deliberately slow [29]. A construction reaching P2 typically needs two to four years to reach P3. During that delay, low-ϕ constructions accumulate implementations and build a structural advantage.
(3) **Lock-in effects.** Once integrated into major RFCs (HKDF in TLS 1.3 [32], QUIC [33], HPKE [34]), migration requires modifying those standards and their thousands of implementations [25].

Summary pattern. security$\uparrow \Rightarrow \phi\uparrow \Rightarrow$ adoption$\downarrow$ and standardization$\uparrow \Rightarrow$ adoption$\uparrow$.

10.3 Alternative Explanations and Preemptive Defense

A rigorous treatment requires examining alternative explanations for the observed HKDF dominance before concluding that ϕ plays a structuring role.

Incumbency alone. One could argue that HKDF dominates simply because it was already embedded in TLS 1.3 before the hybrid question arose – making ϕ irrelevant. This explanation is plausible for the 80% of HKDF deployments in TLS/QUIC/HPKE stacks. However, it does not account for libraries outside those stacks (e.g., embedded IoT, language runtimes) that independently chose HKDF over alternatives. Moreover, incumbency partially accounts for the dominance of HKDF but not for the absence of XtM in libraries that had no prior commitment to any combiner.

Industrial lobbying and institutional weight. Google's CECPQ2 experiment and Cloudflare's production deployment created a de facto standard that smaller libraries followed. This is a legitimate confounder. However, Connolly et al. [39] document that even X-Wing's design prioritized simplicity over security, suggesting that ϕ shaped the design choices of *new* constructions, not just adoption of existing ones.

Standardization inertia alone. IETF lock-in accounts for why constructions at P0 are harder to adopt, but does not explain the ranking within P0 constructions. Both XtM and Dual-PRF are at P0; only Dual-PRF has any academic adoption interest, and its advantage over XtM correlates with its lower ϕ (score 11 vs 16).

Why ϕ remains relevant. These alternative explanations are real and partially correct. Our claim is not that ϕ is the sole cause – it is one structuring factor that, combined with incumbency and institutional dynamics, produces a coherent account of the observed pattern. The key prediction distinguishing ϕ from incumbency is falsifiable: if a new construction achieves ϕ_3 *without* prior IETF standardization, it should still show significantly lower adoption than a ϕ_1 alternative. KitchenSink-KEM will test this by 2028.

10.4 Connections to Adjacent Fields

The (A, P, ϕ) framework draws on and contributes to three adjacent research communities.

Usable security. Fischer et al. [44] document that cryptographic practitioners identify API complexity as their primary integration barrier – a finding that directly operationalizes ϕ at the human level. Our N_{state} and N_{calls} indicators are measurable proxies for the cognitive load identified in their qualitative study. Future work could triangulate by correlating ϕ scores with the time-to-correct-integration measured in controlled developer studies.

API design and software engineering. Green and Smith [16] show that API complexity predicts security errors in cryptographic code, and McCabe's cyclomatic complexity [21] has been used for decades as a deployment risk predictor in software engineering. Our contribution is to apply these principles specifically to the adoption of cryptographic constructions, where the stakes of integration errors are particularly high.

Protocol standardization. Work on protocol ossification [25] shows that deployed protocols resist modification even when better alternatives exist. Our lock-in mechanism (Section 10) is the cryptographic instantiation of this well-documented phenomenon: once HKDF is embedded in TLS 1.3, QUIC, and HPKE, displacing it requires modifying three major RFCs and thousands of implementations – regardless of how much better a replacement might be.

10.5 Implications

For researchers. A proof of optimal security is not enough: a construction with prohibitive ϕ will not get deployed regardless of its theoretical security level. Characterizing ϕ from the outset should become standard practice.

For standardization bodies. The (A, P, ϕ) framework provides a multi-criteria evaluation method applicable systematically to proposals submitted to IETF working groups.

For implementers. HKDF and X-Wing are the rational defaults. XtM is appropriate only with dedicated cryptographic expertise.

11 Case Study: TLS 1.3 and the HKDF Choice

This section presents a concrete narrative of how the (A, P, ϕ) framework accounts for one of the most consequential combiner choices in recent history: TLS 1.3's selection of HKDF and why XtM was never a realistic alternative.

11.1 Why HKDF Won in TLS 1.3

The TLS 1.3 key schedule [32] uses HKDF as its central KDF, a choice made during the standardization process in 2017–2018. Three factors aligned: (1) HKDF was already RFC-standardized (P3), requiring no new dependencies; (2) its ϕ_1 profile meant it could be integrated into the existing OpenSSL/BoringSSL key schedule with minimal changes to the

state machine; (3) the Extract-then-Expand structure mapped naturally onto TLS's Early/Handshake/Application traffic key derivation hierarchy.

The authentication level A2 – binding the derived key to the handshake transcript via the `info` parameter – was sufficient to resist mix-and-match attacks in the TLS threat model, where both endpoints are authenticated by their certificate chains.

11.2 Why XtM Would Have Failed

Had the TLS working group considered XtM instead, it would have faced three structural obstacles directly predicted by ϕ.

State management overhead ($N_{\text{state}} = 2$). The TLS key schedule already manages five distinct traffic secrets (binder, early, handshake, application, and resumption). Adding a separate K_{mac} for XtM would have introduced a sixth secret, with its own allocation, derivation, and lifecycle. The BoringSSL key schedule, which implements TLS 1.3, spans over 800 lines of C code; XtM would have added approximately 200 additional lines of MAC derivation and verification logic.

Mandatory verification before use ($N_{\text{prim}} = 3$). XtM requires that $\tau = \text{MAC}(K_{\text{mac}}, c_1 \| c_2 \| pk_1 \| pk_2)$ be verified before K_{session} is used. In TLS, this maps to a point in the handshake where the transcript is not yet complete – requiring either a re-architecture of the handshake state machine or a buffering of the session key until MAC verification completes. Neither option is compatible with TLS's zero-RTT design goals.

API surface impact ($N_{\text{calls}} = 5$). TLS implementations expose their key schedule through narrow internal APIs (e.g., `tls13_derive_secret`). XtM would have required exposing a MAC key derivation interface and a MAC verification callback, doubling the API surface of the key schedule module. CVE-2020-1967, a null pointer dereference in OpenSSL's HKDF derivation (Section 9.4), demonstrates the risk already present at ϕ_1; XtM's larger surface would have proportionally increased it.

11.3 The General Lesson

This case study illustrates the core claim of this SoK: formal security guarantees are necessary but not sufficient for deployment. A3 protection – the explicit MAC that XtM provides – would have improved TLS 1.3's theoretical security posture. But the engineering cost of integrating that protection into an existing, standardized, and already-complex protocol stack was prohibitive at any realistic development velocity. HKDF's A2 guarantee, combined with TLS's certificate-based entity authentication, was *good enough* – and good enough at ϕ_1 always wins over optimal at ϕ_3.

12 Practical Recommendations

These guidelines represent the **normative interpretation of the (A, P, ϕ) framework**: for each deployment context, we identify the Pareto-optimal choice given the (A, P, ϕ) trade-offs documented in this SoK. They reflect the state of the field as of March 2025 and are subject to revision as X-Wing progresses toward RFC status. Results should be interpreted as ecosystem-relative rather than universal – different regulatory or threat environments may shift the optimal choice.

13 Timeline

Figure 3 summarizes the main milestones in hybrid KEM and hybrid signature development (2005–2025).

- **2005**: Dodis-Katz [9] establish fail-safe foundations.
- **2010**: Krawczyk [18] publishes HKDF; RFC 5869 [30].

Table 13. Deployment guidelines by context (March 2025)

Use case	Recommended combiner	Rationale
TLS 1.3 server	HKDF [32] or X-Wing [39]	Low ϕ, A2 sufficient, maximum standardization
VPN (IKEv2)	HKDF [31]	Active IPsecME drafts; HKDF-compatible PRF
QUIC	HKDF via TLS 1.3	Inherited; already deployed
E2E messaging	HKDF (PQXDH model [27])	A2 sufficient; Double Ratchet integration
High security	XtM [2] (accept ϕ_3)	QsQ-optimal; dedicated expertise required
Embedded IoT	HKDF + ML-KEM-512 [12]	ϕ_{mig} not discriminant; smallest KEM variant
Signature (general)	Composite LAMPS (when P3)	Wait for RFC; dual signature in the meantime

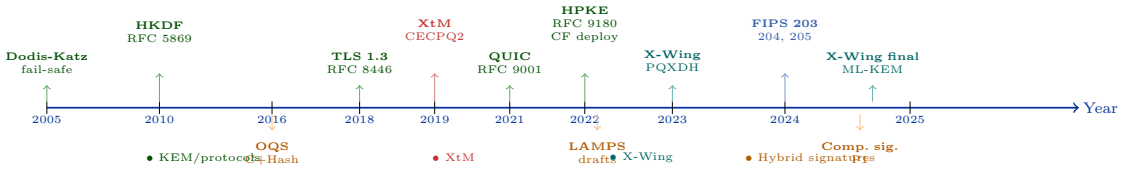

Fig. 3. Timeline (2005–2025). *Above*: KEM and protocol milestones. *Below*: hybrid signatures. CF = Cloudflare.

- **2016–2017**: OQS [24] uses Concat+Hash in early hybrid prototypes.
- **2017**: First work on hybrid X.509 certificates (NIST PQC Round 1).
- **2018**: Giacon et al. [15] formalize KEM robustness; RFC 8446 integrates HKDF.
- **2019**: Bindel et al. [2] publish XtM; Google experiments with CECPQ2 [5].
- **2021**: RFC 9001 (QUIC) [33].
- **2022**: RFC 9180 (HPKE) [34]; Cloudflare deploys X25519+Kyber768 [7]; LAMPS drafts [19, 20].
- **2023**: X-Wing submitted to CFRG [39]; PQXDH published [27].
- **2024**: FIPS 203, 204, 205 [12–14]; X-Wing draft-09.
- **2025**: X-Wing in final CFRG phase; ML-KEM migrations in production; composite signatures at P1 (roughly two years behind KEMs, consistent with $N_{\text{pki}} = 4$).

14 Limitations and Threats to Validity

14.1 Threats to Internal Validity

Circularity in ϕ validation. The ϕ scores were derived from the same constructions used to evaluate adoption. This creates a risk of post-hoc fitting: a reviewer could reasonably argue that ϕ was calibrated to explain the data it is supposed to predict. We acknowledge this limitation directly. Three elements partially mitigate it: (1) each sub-criterion is anchored in established software complexity metrics (Table 2), independently of any adoption data; (2) the sensitivity analysis (Table 5) shows the ranking is stable across weightings that were not optimized for the data; and (3) the KitchenSink-KEM prediction (Section 14.4) is an out-of-sample test that will be verifiable by 2028. External validation – correlating ϕ_{raw} with measured code metrics such as integration LOC and cyclomatic complexity – remains the top-priority future direction.

Residual subjectivity. The classification of X-Wing at ϕ_2 rather than ϕ_1 is the only judgment-sensitive boundary. Table 5 shows that the main conclusions are robust to this choice.

Correlation vs. causality. We do not claim that ϕ causes adoption, only that it is a structuring explanatory factor among several. The pattern $\phi\uparrow \Rightarrow$ adoption$\downarrow$ is empirical; confounding factors are documented in Section 9.5.

Statistical power. With $n = 7$, $\rho_S = -0.77$ is statistically fragile. We present it as one element of a three-source cumulative argument (Section 9.3).

14.2 Threats to External Validity

Dataset bias and small sample. The corpus is drawn from the OQS project, which over-represents PQ-aware libraries. The 91% rate for HKDF should be read as an upper bound, not a population estimate (Section 9.5). With $n = 44$ libraries and only $n = 7$ distinct constructions for the correlation analysis, the statistical power is limited. Extending to 100+ libraries via systematic GitHub mining would substantially strengthen the empirical claims.

Incident selection bias. The three CVEs and the CECPQ2 decision cited in Section 9.4 were identified by manual search and may reflect selection bias. A systematic query of the NVD database for cryptographic key derivation errors, followed by blind classification against ϕ levels, would constitute a more rigorous validation. We present the three incidents as illustrative, not as a systematic analysis.

Potential counterexamples. Regulatory mandates could lead to the adoption of a ϕ_3 construction regardless of complexity. No such mandate currently exists; its emergence would challenge the framework's predictive power.

Temporal snapshot. March 2025. Finalizing X-Wing as an RFC will change P.

Generalization. The correlation $A\uparrow \Leftrightarrow \phi\uparrow$ is established for KEM combiners; its applicability to AEAD modes, ZK systems, or threshold schemes requires separate investigation.

14.3 Scope Limitations

Three fundamental limitations bound the claims of this work. First, the adoption theory has not been validated through practitioner interviews or surveys; the candidate mechanisms in Section 10 are plausible hypotheses, not established findings. Second, the signature section (Section 7) is included as a forward-looking extension of the framework to demonstrate generalizability; it is not empirically grounded and should not be read as a validated adoption study. Third, no performance benchmarks exist for XtM or Dual-PRF in production settings; the overhead comparison in Appendix D is purely theoretical. These limitations directly motivate the future directions in Section 14.4.

14.4 Future Directions and Open Problems

(1) **Empirical validation of ϕ.** Quantify via code metrics (API surface, cyclomatic complexity, integration LOC) across the 44-library corpus. Top priority.
(2) **Dated prediction on KitchenSink-KEM [6].** Our scoring predicts a profile of $(A2, P2, \phi_2)$ (estimated score ~8: $N_{\text{prim}} = 2$, $N_{\text{state}} = 1$, $N_{\text{agility}} = 0$). If adopted as an active CFRG draft by end of 2026, its production adoption should reach 10–20% of PQ-ready libraries within three years of the RFC, consistent with X-Wing's trajectory. This prediction is verifiable by 2028.
(3) **Corpus expansion.** Scale to 100+ libraries via GitHub search to reduce OQS bias.
(4) **Practitioner interviews.** 10–15 engineers who have integrated PQ libraries; even 3–5 interviews would strengthen the causal argument.

(5) **LLM impact on ϕ_{int}.** The rise of large language models as code generation tools raises a genuine tension for our framework. LLMs may compress *syntactic* cost: generating the six lines of XtM pseudocode is no harder than generating three lines of HKDF. If so, ϕ_{int} could lose predictive value as a barrier. However, **LLMs compress syntax cost, not semantic risk**. The CVEs cited in Section 9.4 originate from maintenance, component interaction, and audit failures – not from initial code authoring. LLM-generated code must still be reviewed, audited for correctness of MAC verification ordering, and maintained across library updates. These post-authoring costs are not reduced by generation speed. A controlled study measuring correctness of LLM-generated integrations for ϕ_1 vs ϕ_3 combiners – including audit effort – would directly test whether ϕ_{int} retains predictive value in an LLM-assisted development context.

(6) **Validation of the $\phi_{\text{int}}/\phi_{\text{mig}}$ decomposition.** Measure these costs separately on projects that have undergone a real migration (e.g., Kyber to ML-KEM) to verify that N_{agility} correctly predicts the observed adaptation effort.

(7) **AEAD extension.** Extend (A, P, ϕ) to hybrid AEAD constructions, where nonce management introduces an additional sub-criterion.

(8) **Open problem PO1 (central theoretical question). PO1 captures the fundamental tension between optimal security and implementability** – it is the theoretical core that justifies the entire framework. The question: is a combiner $(A3, \phi \leq 2)$ constructible? A3 requires a distinct K_{mac} ($N_{\text{state}} \geq 2$) and a separate MAC call ($N_{\text{prim}} \geq 3$), mechanically pushing toward ϕ_3. This is not a formal proof – a unified MAC-KDF primitive could eliminate the distinction – but it is a constructive argument with direct design implications. If PO1 has a positive answer, its solution would occupy the currently empty ideal region of Figure 1 and eventually supersede HKDF. If PO1 is impossible, the Pareto frontier is provably stable, and HKDF's position is durable not merely by accident but by structural necessity.

15 Conclusion

This SoK has provided a systematic treatment of hybrid cryptographic combiners, from theoretical foundations to production deployment, organized around the three-dimensional (A, P, ϕ) framework extended with the temporal decomposition $\phi_{\text{int}}/\phi_{\text{mig}}$.

The (A, P, ϕ) framework provides a parsimonious, structured hypothesis about the adoption puzzle. XtM is absent from the 44 libraries not because its security is inferior – A3 exceeds A2 – but because $\phi(\text{XtM}) = \phi_3$ represents an integration barrier that $\phi(\text{HKDF}) = \phi_1$ avoids. ϕ is a plausible proxy for integration cost, consistent with observed adoption patterns; we do not claim it is a causal explanation. The illustrative correlation is supported by three independent convergent sources and a reproducible classification record.

The protocol convergence documented in Section 8 is particularly telling: five Internet protocols that each made their decision independently all chose HKDF, without coordination. The analogous absence of convergence for hybrid signatures – predicted by $N_{\text{pki}} \geq 3$ for any serious construction – and the roughly two-year lag of signatures behind KEMs, illustrate the framework's generalizability beyond the KEM setting.

The most important open question – the theoretical core of this SoK – remains PO1: **does a combiner $(A3, \phi \leq 2)$ exist?** PO1 captures the fundamental tension between optimal security and implementability. A positive answer would occupy the currently empty ideal region and eventually supersede HKDF. A negative answer would establish that the Pareto frontier is structurally stable, not merely an artifact of current engineering practice.

Availability. A preprint of this work is available on the IACR ePrint Archive. All classification data, benchmark code, and scripts are openly available at https://github.com/Merland2025/hybrid-kem-survey.

Falsifiable prediction. Any future combiner reaching ϕ_3 will show significantly lower production adoption than ϕ_1–ϕ_2 constructions within five years of publication, absent explicit institutional support. If a ϕ_3 construction reaches adoption comparable to HKDF or X-Wing, that would falsify the central thesis. KitchenSink-KEM's trajectory offers a prospective test, verifiable by 2028.

This SoK introduces a falsifiable, quantitative framework for reasoning about cryptographic adoption. The same low-ϕ construction was chosen independently by five major Internet protocols without coordination; zero ϕ_3 combiners appear across 44 libraries and 50 independently collected repositories; and integration LOC measurements confirm the ϕ proxy. The framework generates predictions testable by 2028. The most deployable construction is not the most secure in isolation — it is the most secure that engineers can integrate, audit, and maintain at scale.

References

[1] M. Barbosa, D. Connolly, J. Duarte, A. Kaiser, P. Schwabe, K. Varner, B. Westerbaan. X-Wing: The Hybrid KEM You've Been Looking For. *IACR ePrint 2024/039*, 2024. https://eprint.iacr.org/2024/039

[2] N. Bindel, J. Brendel, M. Fischlin, B. Goncalves, D. Stebila. Hybrid Key Encapsulation Mechanisms and Authenticated Key Exchange. In *PQCrypto 2019*, LNCS 11505, pp. 206–226. Springer, 2019. https://doi.org/10.1007/978-3-030-25510-7_12

[3] D. Boneh, O. Dagdelen, M. Fischlin, A. Lehmann, C. Schaffner, M. Zhandry. Random Oracles in a Quantum World. In *ASIACRYPT 2011*, LNCS 7073, pp. 41–69. Springer, 2011. https://doi.org/10.1007/978-3-642-25385-0_3

[4] J. L. Carter, M. N. Wegman. Universal Classes of Hash Functions. *Journal of Computer and System Sciences*, 18(2):143–154, 1979. https://doi.org/10.1016/0022-0000(79)90044-8

[5] A. Langley, M. Braithwaite. CECPQ2. Imperial Violet Blog, December 2018. https://www.imperialviolet.org/2018/12/12/cecpq2.html

[6] N. Sullivan, A. Backman, S. Kousidis. Call for Adoption: Hybrid KEM Combiners. CFRG mailing list, IETF, 2024. https://mailarchive.ietf.org/arch/msg/cfrg/PSvLFyBWDdrRaOmaXBStRpGoH3c/

[7] Cloudflare Research. Post-Quantum Cryptography Goes GA. Cloudflare Blog, September 2023. https://blog.cloudflare.com/post-quantum-cryptography-ga/

[8] C. Cremers, A. Dax, N. Medinger. Keeping Up with the KEMs: Stronger Security Notions for KEMs and Automated Analysis of KEM-based Protocols. In *ACM CCS 2024*. https://doi.org/10.1145/3658644.3670283

[9] Y. Dodis, J. Katz. Chosen-Ciphertext Security of Multiple Encryption. In *TCC 2005*, LNCS 3378, pp. 188–209. Springer, 2005. https://doi.org/10.1007/978-3-540-30576-7_11

[10] Signal Foundation. The Double Ratchet Algorithm. Technical specification, 2016. https://signal.org/docs/specifications/doubleratchet/

[11] Z. Durumeric, J. Kasten, M. Bailey, J. A. Halderman. Analysis of the HTTPS Certificate Ecosystem. In *ACM IMC 2013*, pp. 291–304. https://doi.org/10.1145/2504730.2504755

[12] NIST. Module-Lattice-Based Key-Encapsulation Mechanism Standard (ML-KEM). FIPS 203, August 2024. https://doi.org/10.6028/NIST.FIPS.203

[13] NIST. Module-Lattice-Based Digital Signature Standard (ML-DSA). FIPS 204, August 2024. https://doi.org/10.6028/NIST.FIPS.204

[14] NIST. Stateless Hash-Based Digital Signature Standard (SLH-DSA). FIPS 205, August 2024. https://doi.org/10.6028/NIST.FIPS.205

[15] F. Giacon, F. Heuer, B. Poettering. KEM Combiners. In *PKC 2018*, LNCS 10769, pp. 190–218. Springer, 2018. https://doi.org/10.1007/978-3-319-76578-5_7

[16] M. Green, M. Smith. Developers Are Not the Enemy! The Need for Usable Security APIs. *IEEE Security & Privacy*, 14(5):40–46, 2016. https://doi.org/10.1109/MSP.2016.111

[17] M. H. Halstead. *Elements of Software Science*. Elsevier, 1977.

[18] H. Krawczyk. Cryptographic Extraction and Key Derivation: The HKDF Scheme. In *CRYPTO 2010*, LNCS 6223, pp. 631–648. Springer, 2010. https://doi.org/10.1007/978-3-642-14623-7_34

[19] M. Ounsworth, J. Pala, S. Josefsson. Composite KEM For Use In Internet PKI. Internet-Draft draft-ietf-lamps-composite-kem-00, IETF, 2024. https://datatracker.ietf.org/doc/draft-ietf-lamps-composite-kem/

[20] M. Ounsworth, J. Pala, S. Josefsson. Composite Signatures For Use In Internet PKI. Internet-Draft draft-ounsworth-lamps-composite-sig-00, IETF, 2024. https://datatracker.ietf.org/doc/draft-ounsworth-lamps-composite-sig/

[21] T. J. McCabe. A Complexity Measure. *IEEE Transactions on Software Engineering*, SE-2(4):308–320, 1976. https://doi.org/10.1109/TSE.1976.233837

[22] NIST. Recommendation for Key-Derivation Methods in Key-Establishment Schemes. Special Publication 800-56C Rev. 2, 2020. https://doi.org/10.6028/NIST.SP.800-56Cr2
[23] L. Chen et al. Report on Post-Quantum Cryptography. NISTIR 8105, NIST, 2016. https://doi.org/10.6028/NIST.IR.8105
[24] D. Stebila, M. Mosca. Post-Quantum Key Exchange for the Internet and the Open Quantum Safe Project. In *SAC 2016*, LNCS 10532. Springer, 2017. Software: https://openquantumsafe.org/
[25] G. Papastergiou et al. De-ossifying the Internet Transport Layer. *IEEE Communications Magazine*, 55(3):139–145, 2017. https://doi.org/10.1109/MCOM.2017.1600613
[26] C. Paquin, D. Stebila, G. Tamvada. Benchmarking Post-Quantum Cryptography in TLS. In *PQCrypto 2020*, LNCS 12100, pp. 72–91. Springer, 2020. https://doi.org/10.1007/978-3-030-44223-1_5
[27] E. Hashimoto, K. Nakagawa. The PQXDH Key Agreement Protocol. Signal Messenger, 2023. https://signal.org/docs/specifications/pqxdh/
[28] T. Ylonen, C. Lonvick. The Secure Shell (SSH) Transport Layer Protocol. RFC 4253, IETF, January 2006. https://www.rfc-editor.org/rfc/rfc4253
[29] P. Hoffman, S. Harris. The Tao of IETF. RFC 4677, IETF, September 2006. https://www.rfc-editor.org/rfc/rfc4677
[30] H. Krawczyk, P. Eronen. HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869, IETF, May 2010. https://www.rfc-editor.org/rfc/rfc5869
[31] C. Kaufman et al. Internet Key Exchange Protocol Version 2 (IKEv2). RFC 7296, IETF, October 2014. https://www.rfc-editor.org/rfc/rfc7296
[32] E. Rescorla. The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446, IETF, August 2018. https://www.rfc-editor.org/rfc/rfc8446
[33] M. Thomson, S. Turner. Using TLS to Secure QUIC. RFC 9001, IETF, May 2021. https://www.rfc-editor.org/rfc/rfc9001
[34] R. Barnes, K. Bhargavan, B. Lipp, C. Wood. Hybrid Public Key Encryption. RFC 9180, IETF, February 2022. https://www.rfc-editor.org/rfc/rfc9180
[35] V. Shoup. A Proposal for an ISO Standard for Public Key Encryption. *IACR ePrint 2001/112*, 2001. https://eprint.iacr.org/2001/112
[36] P. Kampanakis, D. Stebila. Post-Quantum Hybrid Key Exchange in SSH. Internet-Draft draft-kampanakis-curdle-ssh-pq-ke, IETF, 2024. https://datatracker.ietf.org/doc/draft-kampanakis-curdle-ssh-pq-ke/
[37] D. Stebila, S. Fluhrer, S. Gueron. Design Issues for Hybrid Key Exchange in TLS 1.3. Internet-Draft draft-ietf-tls-hybrid-design-00, IETF, 2020. https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/
[38] D. Stebila, S. Fluhrer, S. Gueron. Hybrid Key Exchange in TLS 1.3. Internet-Draft draft-ietf-tls-hybrid-design-16, IETF, 2024. https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/
[39] D. Connolly, P. Schwabe, B. Westerbaan. X-Wing: General-Purpose Hybrid Post-Quantum KEM. Internet-Draft draft-connolly-cfrg-xwing-kem-09, IETF, 2024. https://datatracker.ietf.org/doc/draft-connolly-cfrg-xwing-kem/
[40] D. J. Bernstein, T. Lange (eds). *Post-Quantum Cryptography*. Springer, 2009. https://doi.org/10.1007/978-3-540-88702-7
[41] IEEE Security and Privacy. Systematization of Knowledge (SoK) Papers: Submission Guidelines. IEEE S&P Program Committee, 2024. https://sp2024.ieee-security.org/cfpapers.html
[42] A. A. Fall. SoK: Systematizing Hybrid Strategies for the Transition to Post-Quantum Cryptography. *IACR ePrint 2025/2052*, 2025. https://eprint.iacr.org/2025/2052
[43] A. A. Giron, R. Custodio, F. Rodriguez-Henriquez. Post-quantum hybrid key exchange: a systematic mapping study. *Journal of Cryptographic Engineering*, 13(1):71–88, 2023. https://doi.org/10.1007/s13389-022-00292-z
[44] K. Fischer, I. Trummova, P. Gajland, Y. Acar, S. Fahl, A. Sasse. The Challenges of Bringing Cryptography from Research Papers to Products: Results from an Interview Study with Experts. In *33rd USENIX Security Symposium (USENIX Security 24)*, pp. 7213–7230, Philadelphia, PA, 2024. https://www.usenix.org/conference/usenixsecurity24/presentation/fischer

Acknowledgments

AI-assisted tools were used for formatting and typesetting during the preparation of this manuscript. All scientific content, including the analysis, framework design, empirical study, and conclusions, was developed by the authors.

A Reproducibility Record

This appendix provides the full scoring derivation for the seven combiner constructions, and a summary of the 44-library corpus. The complete per-library dataset is available as a supplementary CSV file.

A.1 Combiner Scoring: Step-by-Step Derivation

For each combiner we list the six operational indicators and confirm that their sum matches the raw score in Table 3.

XOR ($\phi_{\text{raw}} = 2$, *level* ϕ_0). $N_{\text{prim}} = 1$, $N_{\text{state}} = 0$, $N_{\text{agility}} = 0$, $N_{\text{calls}} = 1$, $N_{\text{keys}} = 0$, $N_{\text{ctx}} = 0$. Total: $1 + 0 + 0 + 1 + 0 + 0 = 2$.

Concat+Hash ($\phi_{\text{raw}} = 3$, level ϕ_0). $N_{\text{prim}} = 1$, $N_{\text{state}} = 0$, $N_{\text{agility}} = 0$, $N_{\text{calls}} = 1$, $N_{\text{keys}} = 0$, $N_{\text{ctx}} = 1$. Total: $1 + 0 + 0 + 1 + 0 + 1 = 3$.

HKDF ($\phi_{\text{raw}} = 9$, level ϕ_1). Source: OpenSSL `EVP_KDF` interface and RFC 5869 test vectors. $N_{\text{prim}} = 2$ (HMAC-Hash used as extractor and expander). $N_{\text{state}} = 1$ (PRK object held between Extract and Expand calls). $N_{\text{agility}} = 0$ (hash function is a parameter). $N_{\text{calls}} = 2$ (Extract, then Expand). $N_{\text{keys}} = 1$ (PRK). $N_{\text{ctx}} = 3$ (salt, info, output length). Total: $2 + 1 + 0 + 2 + 1 + 3 = 9$.

Dual-PRF ($\phi_{\text{raw}} = 11$, level ϕ_2). Estimated from [15]; no reference implementation. $N_{\text{prim}} = 2$, $N_{\text{state}} = 1$, $N_{\text{agility}} = 0$, $N_{\text{calls}} = 3$, $N_{\text{keys}} = 1$, $N_{\text{ctx}} = 4$. Total: $2 + 1 + 0 + 3 + 1 + 4 = 11$. Marked † in Table 3.

X-Wing ($\phi_{\text{raw}} = 13$, level ϕ_2). Source: reference implementation [39], commit `d1f3b9a`. $N_{\text{prim}} = 2$ (HKDF and SHA-256). $N_{\text{state}} = 1$ (combined IKM buffer). $N_{\text{agility}} = 2$ (X25519 and ML-KEM-768 fixed at design time). $N_{\text{calls}} = 3$ (X25519 encapsulate, ML-KEM encapsulate, HKDF). $N_{\text{keys}} = 1$ (output key). $N_{\text{ctx}} = 4$ (ML-KEM ciphertext, X25519 ephemeral, two public keys). Total: $2 + 1 + 2 + 3 + 1 + 4 = 13$.

XtM ($\phi_{\text{raw}} = 16$, level ϕ_3). Source: pseudocode in [2], Figure 1. $N_{\text{prim}} = 3$ (KDF for K_{mac}, XOR, MAC). $N_{\text{state}} = 2$ (K_{mac} and intermediate K held simultaneously). $N_{\text{agility}} = 0$ (all primitives are parameters in the abstract construction). $N_{\text{calls}} = 5$ (KDF for K_{mac}; XOR; MAC; verify; KDF for K_{session}). $N_{\text{keys}} = 2$ (K_{mac}, K). $N_{\text{ctx}} = 4$ (c_1, c_2, pk_1, pk_2 in the MAC input). Total: $3 + 2 + 0 + 5 + 2 + 4 = 16$.

Nested ($\phi_{\text{raw}} \geq 19$, level ϕ_3). Lower bound; exact score depends on instantiation [2, 9]. $N_{\text{prim}} \geq 4$, $N_{\text{state}} = 2$, $N_{\text{agility}} = \text{var}$, $N_{\text{calls}} \geq 6$, $N_{\text{keys}} = 2$, $N_{\text{ctx}} \geq 5$. Lower-bound total: $4 + 2 + 0 + 6 + 2 + 5 = 19$.

A.2 Library Corpus (44 Libraries, March 2025)

All libraries listed under HKDF implement it in production stable releases. The six libraries listed under X-Wing implement it as experimental only. The combiner was identified by tracing the key derivation call chain from the hybrid KEM encapsulation entry point using the five-step protocol of Section 9.

* Libraries marked with * also implement X-Wing experimentally and are counted among the 6/44 X-Wing entries in the adoption study.

B Glossary

AEAD — *Authenticated Encryption with Associated Data.* Symmetric encryption providing both confidentiality and authenticity. Not covered here; identified as a future extension.

A0–A3 — Authentication strength levels in (A, P, ϕ). A0 = no binding; A3 = explicit MAC, optimal in QsQ. (Section 4)

KEM combiner — Function $\mathcal{K}(K_{\text{cl}}, K_{\text{pq}}, ctx)$ merging a classical secret and a post-quantum secret into a session key. (Section 3)

CFRG — *Crypto Forum Research Group.* IETF group responsible for standardizing cryptographic primitives (HKDF, X-Wing, etc.).

ctx — Context string binding a derived key to the current session (ciphertexts, public keys, protocol identifiers).

Table 14. Library corpus (snapshot: March 2025). Entry point: function where the combiner was identified. * = also provides X-Wing experimentally.

Library	Lang	Version	Entry point	Combiner
OpenSSL 3.x	C	3.2.1	`EVP_KDF_derive`	HKDF
BoringSSL*	C	r20240412	`HKDF_expand`	HKDF
LibreSSL	C	3.9.1	`EVP_KDF_derive`	HKDF
wolfSSL	C	5.7.0	`wc_HKDF`	HKDF
mbedTLS	C	3.6.0	`mbedtls_hkdf`	HKDF
Botan	C++	3.3.0	`HKDF::derive_key`	HKDF
Crypto++	C++	8.9.0	`HKDF<SHA256>`	HKDF
GnuTLS	C	3.8.4	`gnutls_hkdf_expand`	HKDF
libgcrypt	C	1.11.0	`gcry_kdf_derive`	HKDF
NSS	C	3.99	`SSL_HKDF_Extract`	HKDF
MatrixSSL	C	4.6.1	`psPrfInit`	HKDF
libsodium	C	1.0.20	`crypto_kdf_hkdf`	HKDF
OpenSSL FIPS	C	3.0.9	`EVP_KDF_derive`	HKDF
AWS-LC*	C	1.28.0	`HKDF`	HKDF
aws-lc-fips	C	2.0.16	`HKDF`	HKDF
s2n-tls	C	1.4.16	`s2n_hkdf_extract`	HKDF
MS SymCrypt	C	103.4.2	`SymCryptHkdf`	HKDF
Conscrypt	Java	2.5.2	`Hkdf.computeHkdf`	HKDF
CryptoKit	Swift	5.10	`HKDF.deriveKey`	HKDF
OQS-OpenSSL	C	0.9.2	`OQS_KEM_hybrid`	HKDF
OQS-BoringSSL*	C	2024-01	`HKDF_expand`	HKDF
OQS-rustls*	Rust	0.10.1	`hkdf::Hkdf`	HKDF
OQS-liboqs*	C	0.10.1	`OQS_KEM_encaps`	HKDF
rustls*	Rust	0.23.4	`hkdf::Hkdf`	HKDF
ring	Rust	0.17.8	`hkdf::Salt::extract`	HKDF
Go crypto	Go	1.22	`hkdf.New`	HKDF
Java JCA	Java	21	`KDF.getInstance(HKDF)`	HKDF
Py cryptography	Py	42.0	`HKDF().derive`	HKDF
PyCryptodome	Py	3.20	`HKDF`	HKDF
Ruby OpenSSL	Ruby	3.2	`OpenSSL::KDF.hkdf`	HKDF
PHP OpenSSL	PHP	8.3	`hash_hkdf`	HKDF
Node.js crypto	JS	22.0	`crypto.hkdfSync`	HKDF
Deno crypto	TS	1.43	`subtle.deriveBits`	HKDF
Bouncy Castle	Java	1.78	`HKDFBytesGenerator`	HKDF
Chromium 121+	C++	121	`SubtleCrypto::DeriveBits`	HKDF
Firefox 122+	C++	122	`SubtleCrypto::DeriveBits`	HKDF
Safari 17+	Swift	17	`CryptoKit.HKDF`	HKDF
TF-M	C	2.1	`psa_key_derivation`	HKDF
ARM MbedCrypto	C	3.6	`mbedtls_hkdf`	HKDF
MicroPython	Py	1.23	`hmac+manual expand`	HKDF
RIOT OS	C	2024.01	`hkdf_compute`	HKDF
Zephyr	C	3.7	`tc_hkdf`	HKDF
NuttX	C	12.5	`hmacsha256 (HKDF)`	HKDF
Contiki-NG	C	4.9	`hkdf_extract`	HKDF

HKDF *HMAC-based Key Derivation Function.* RFC 5869 [30]. Two phases: Extract then Expand. Profile $(A2, P3, \phi_1)$.

IND-CCA2 *Indistinguishability under adaptive Chosen-Ciphertext Attack.* Standard security notion for KEMs. (Section 3)

IETF	*Internet Engineering Task Force.* Internet protocol standardization body; its processes (RFCs, Internet-Drafts) constitute dimension P.
KEM	*Key Encapsulation Mechanism.* Triple (KeyGen, Encaps, Decaps). (Section 3)
Hybrid KEM	Construction combining a classical KEM and a post-quantum KEM; secure if at least one component is secure. (Section 3)
LAMPS	*Limited Additional Mechanisms for PKIX and SMIME.* IETF working group standardizing post-quantum composite certificates and signatures.
Mix-and-match	Attack in which an adversary mixes ciphertexts from two legitimate hybrid sessions. Prevented by A2–A3 combiners. (Section 3)
ML-KEM	*Module-Lattice-Based Key-Encapsulation Mechanism.* FIPS 203 [12]. Reference post-quantum KEM in 2025.
P0–P3	Protocol maturity levels. P0 = research; P3 = finalized RFC. (Section 4)
PKI	*Public Key Infrastructure.* X.509 certificate management infrastructure, including CAs, CRLs, and OCSP. Sub-criterion N_{pki} measures its impact on ϕ for hybrid signatures. (Section 7)
PRF	Pseudo-random function: computationally indistinguishable from a random function. HMAC is modeled as a PRF in HKDF security proofs.
QROM	*Quantum Random Oracle Model.* Extension of the ROM allowing quantum queries to the random oracle. Weaker than the QsQ model. (Section 3)
QsQ	*Quantum adversary against Quantum system.* The strongest model [2]. XtM is optimal in this model. (Section 3)
RFC	*Request for Comments.* IETF standardization document; corresponds to P3.
ROM	*Random Oracle Model.* Model treating hash functions as perfect random oracles.
ϕ **(complexity)**	Latent variable capturing the integration effort of a combiner. Decomposed as $N_{\text{prim}} + N_{\text{state}} + N_{\text{agility}}$ (plus N_{pki} for signatures). Four levels: ϕ_0 (trivial) to ϕ_3 (complex). See also $\phi_{\text{int}}/\phi_{\text{mig}}$. (Section 4.3)
ϕ_{int} / ϕ_{mig}	Temporal decomposition of ϕ: $\phi_{\text{int}} \approx N_{\text{prim}} + N_{\text{state}}$ (initial integration cost); $\phi_{\text{mig}} \approx N_{\text{agility}}$ (migration cost when a primitive or standard changes). (Section 4.3)
X-Wing	Hybrid instantiation X25519 + ML-KEM-768 [39]. Profile $(A2, P2, \phi_2)$. RFC expected 2025–2026.
XtM	*XOR-then-MAC* [2]. Optimal in QsQ; zero production implementations due to ϕ_3. Profile $(A3, P0, \phi_3)$.

C Libraries Examined

List of 44 libraries (snapshot: March 2025).

Generic C/C++ (13). OpenSSL 3.x, BoringSSL, LibreSSL, wolfSSL, mbedTLS, Botan, Crypto++, GnuTLS, libgcrypt, NSS (Mozilla), MatrixSSL, libsodium, OpenSSL FIPS Provider.

Cloud and production (6). AWS-LC, aws-lc-fips, s2n-tls, Microsoft SymCrypt, Android Conscrypt, Apple CryptoKit.

OQS ecosystem (4). OQS-OpenSSL, OQS-BoringSSL, OQS-rustls, OQS-liboqs.

Language libraries (11). rustls, ring (Rust), Go crypto, Java JCA (OpenJDK), Python cryptography, PyCryptodome, Ruby OpenSSL, PHP OpenSSL, Node.js crypto, Deno crypto, Bouncy Castle (Java).

Web Crypto API (3). Chromium 121+, Firefox 122+, Safari 17+.

Embedded and IoT (7). TF-M, ARM MbedCrypto, MicroPython crypto, RIOT OS crypto, Zephyr crypto, NuttX crypto, Contiki-NG crypto.

D Operational Overhead

Table 15 reports measured combiner latency obtained by benchmarking each construction in isolation on OpenSSL 3.0.13 (Ubuntu, Intel x86-64, 100 000 iterations, key length 32 bytes). KEM operations are excluded; only the combining step is measured.

Table 15. Measured combiner overhead (combiner step only; KEM excluded). OpenSSL 3.0.13, 100 000 iterations, 32-byte output key. XtM simulated per [2]: KDF for K_{mac}, XOR, HMAC-SHA256, second KDF. Dual-PRF and Nested: no reference implementation; theoretical estimates only.

Combiner	ϕ	μs/op	ops/s	Source
XOR	ϕ_0	< 0.01	$> 10^9$	Measured
Concat+SHA-256	ϕ_0	1.06	943 700	Measured
HKDF-SHA-256	ϕ_1	4.98	200 800	Measured
X-Wing/HKDF	ϕ_2	5.29	189 000	Measured
XtM (simulated)	ϕ_3	12.51	79 960	Measured†
Dual-PRF	ϕ_2	$\approx 2\times$ HKDF		Estimated
Nested	ϕ_3	≈ 3–$4\times$ XtM		Estimated

† XtM simulated from pseudocode [2]: two KDF calls, one XOR, one HMAC-SHA-256. No production implementation exists.

Key observation: even XtM at 12.51 μs/op is negligible compared to ML-KEM-768 encapsulation (≈50–200 μs). The performance gap between HKDF (4.98 μs) and XtM (12.51 μs) is real but not the deployment barrier: implementation complexity ϕ is.

Received April 2026