Formula Freshness for Staged Hybrid Authenticated Key Exchange

Anis Bkakria
IRT SystemX, France

Chadrel-chrislain BAFOUETILA
IRT SystemX, France; Université de Paris 8, France

June 10, 2026

Abstract

Hybrid post-quantum migration is entering deployed handshake designs, but hybrid KEM security protects only one shared-secret input. It does not by itself say whether handshake, application, exporter, or resumption material remains pseudorandom after branch reveals, stage-key reveals, selective corruptions, or late corruptions. We characterize these staged claims through branch-formula freshness: each stage receives a monotone formula over branch exposure, authentication freshness, transcript binding, KDF ancestry, and explicit non-reveal atoms. Secrecy follows by replacing a surviving branch contribution and then using a labelled HKDF/PRF-style target-hiding argument along a fresh KDF cut; agreement follows separately from authentication binding and injective transcript representation. We also give selector-local accounting, where a fixed admissible witness selector determines which surviving branches and KDF cuts are charged. For scoped TLS 1.3 ECDHE–ML-KEM 1-RTT, we identify the branch-replacement, HKDF-path, and binding assumptions that imply concrete preservation bounds for handshake, application, exporter, and resumption targets.

Keywords: authenticated key exchange ; post-quantum cryptography ; hybrid key exchange ; multi-stage key exchange ; transcript binding

Contents

1 Introduction **4**
- 1.1 Contributions . . . 4

2 Related Work **6**

3 Staged Hybrid AKE Syntax **7**

4 Exposure Algebra and Freshness Formulas **8**

5 KDF Ancestry and HKDF-Style Refresh Separation **9**

6 Transcript Binding and Agreement **12**

7 Stage-Secrecy Preservation **13**
- 7.1 Secrecy Experiment . . . 13
- 7.2 Dominating Cuts for Tested Stages . . . 13
- 7.3 Main Secrecy Theorem . . . 13

8 Stage-Agreement Preservation **14**
- 8.1 Matching and Agreement Freshness . . . 15
- 8.2 Agreement Experiment . . . 15
- 8.3 Authentication-to-Agreement Reduction . . . 16
- 8.4 Main Agreement Theorem . . . 16

9 Separations and Applications: Main-Text Summary **16**
- 9.1 Main Application: Conditional Scoped TLS 1.3 ECDHE–ML-KEM 1-RTT Preservation . . . 17

10 Selector-Local Reduction Accounting **20**
- 10.1 Witnesses and Selectors . . . 21
- 10.2 Selector-Local Secrecy and Agreement . . . 21

11 Scope and Limitations **22**

12 Conclusion **22**

A Proof Interface and Reduction Accounting **23**
- A.1 Canonical Advantage Terms . . . 23
- A.2 Proof Obligations . . . 24
- A.3 Separating Example for Formula Freshness . . . 24
- A.4 Terminology Convention . . . 24

B Deferred Proofs for Main Preservation Theorems **24**
- B.1 Full proof of Theorem 1 . . . 24
- B.2 Full proof of Theorem 2 . . . 25
- B.3 Detailed proofs of Corollaries 7.3 and 7.4 . . . 26

C Separations from Simpler Freshness Notions **27**
- C.1 Flat Freshness Cannot Express Hybrid Survival . . . 27
- C.2 Local Hybrid-KEM Security Does Not Imply Staged KDF Safety . . . 27
- C.3 Ordinary Matching Does Not Imply Hybrid Agreement . . . 28
- C.4 Late Corruption Requires Stage-Indexed Semantics . . . 28
- C.5 Consequence . . . 29

D TLS 1.3 ECDHE–ML-KEM 1-RTT Instantiation **29**
D.1 Protocol Skeleton and Key Schedule 29
D.2 Discharging the Obligations 30
D.3 KEM-Authentication as an Extension 31
D.4 Downgrade and Branch-Substitution Diagnostics 32

1 Introduction

Authenticated key exchange is being redesigned around staged derivation and hybrid trust. A single modern handshake may derive early traffic keys, handshake keys, application keys, exporter keys, resumption secrets, and confirmation keys, while its confidentiality input may combine a classical Diffie–Hellman branch with a post-quantum KEM branch. The security question is therefore not only whether the final session key is pseudorandom. The question is which stage remains secret and agreed after an adversary interleaves many sessions, reveals selected stage keys, compromises selected branch secrets, and corrupts parties before or after acceptance.

Post-quantum migration makes this question structural. Hybrid key exchange is attractive because a classical branch gives confidence against implementation and cryptanalytic uncertainty in new post-quantum schemes, while a post-quantum branch gives resistance against future quantum attacks against classical public-key cryptography. This redundancy is meaningful only if the protocol layer preserves it. A KEM combiner can make a one-shot shared secret robust when one component remains secure, but a staged authenticated protocol additionally has identities, roles, downgrade choices, transcript hashes, authentication binders, and multiple KDF outputs.

Previous work provides two important but incomplete components. Multi-stage AKE models capture the fact that one handshake derives several keys, with properties that may differ by stage. Hybrid KEM and hybrid AKE work captures the robustness goal that one uncompromised component should suffice for confidentiality. Our contribution is not subsumed by either line. Existing TLS 1.3 multi-stage analyses tag many derived keys by security properties, while hybrid KEM combiners give local surviving-component security for one shared secret. The new element here is formula-valued branch freshness plus KDF-cut accounting under hybrid branch reveals: the formula determines whether a stage remains testable and which branch/KDF assumptions are charged.

The central question is therefore concrete: after the adversary has revealed some branch secrets, some derived keys, and possibly some long-term state, which accepted stage keys and agreement claims still survive? We answer by assigning each stage i a monotone formula Φ_i over exposure atoms such as $\mathsf{Unexp}(C)$, $\mathsf{AuthFr}(A)$, $\mathsf{KSafe}(i)$, $\mathsf{Inj}_{\mathsf{sec}}(i)$, and $\mathsf{Inj}_{\mathsf{agr}}(i)$. A test to stage i is fresh exactly when the final adversarial exposure state satisfies Φ_i. The formula records both the surviving branch witness and the KDF ancestry condition needed to protect the target.

The main technical obstacle is that confidentiality, authentication, and KDF leakage do not compose in the same way. Confidentiality may be OR-composed: an ECDH branch or an ML-KEM branch may suffice. Authentication is often AND-composed to prevent downgrade and substitution, but may be OR-composed in migration modes. KDF security is neither OR nor AND over branches: a revealed ancestor can expose a descendant unless the derivation path is refreshed by an unrevealed branch with a domain-separated injection point. Agreement adds an independent injectivity condition over identities, roles, stages, suites, branch public values, and transcript hashes.

The resulting reduction follows the security transitions that the adversary forces. A surviving branch contribution is first replaced at its injection point. A labelled KDF argument then propagates this replacement only across paths not cut by revealed ancestors. Agreement is proved by a separate authentication-to-agreement reduction: if the binder covers identities, roles, suites, stage labels, branch public values, and transcript hashes, then downgrade, branch substitution, and role confusion become either authentication failures or binding collisions.

1.1 Contributions

We make the following contributions.

1. **Branch-formula freshness for staged hybrid AKE.** We define freshness as a monotone formula over confidentiality-branch exposure, authentication freshness, KDF safety, transcript-binding injectivity, and explicit non-reveal atoms. This lets one stage express OR-hybrid confidentiality, another express threshold robustness, and another express strict authentication conjunction without changing the surrounding AKE experiment.

2. **KDF refresh and target-hiding interface.** We formalize labelled KDF graphs and dominating refresh cuts. The KDF analysis separates local branch replacement from labelled HKDF/PRF target hiding, so a protocol proof identifies exactly where a surviving branch refreshes an exposed derivation path.

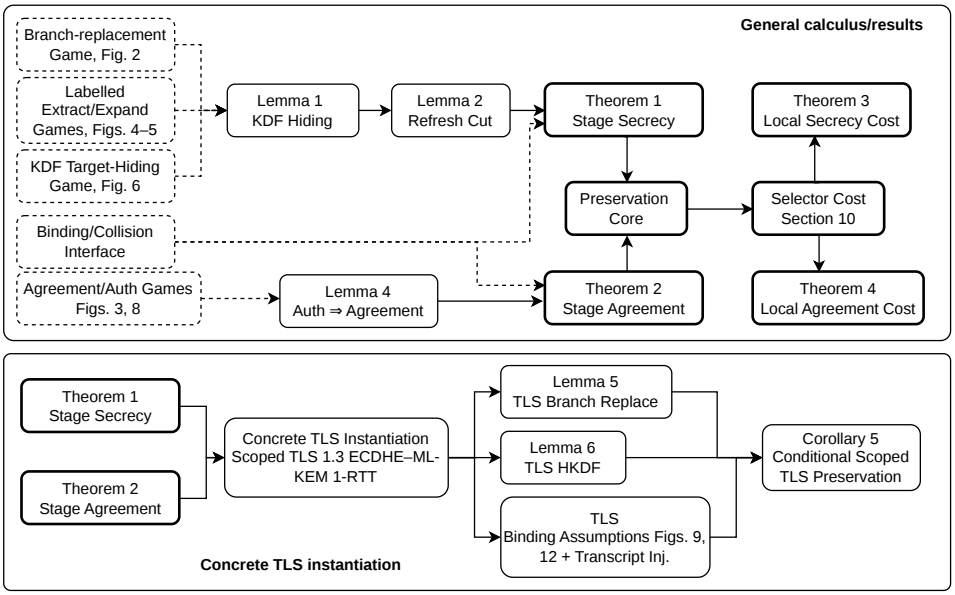
3. **Stage-secrecy and stage-agreement preservation.** We prove general preservation theorems showing that formula-fresh stages retain secrecy and agreement under branch reveals, stage-key reveals, selective corruptions, and late corruptions, provided the local branch, authentication, KDF, and binding obligations are met.

4. **Selector-local reduction accounting.** We give an accounting refinement in which a fixed admissible selector determines which witnesses and KDF cuts are charged for a fixed stage claim. This avoids charging unused branches while preventing post-hoc witness selection in the reduction.

5. **Scoped TLS 1.3 ECDHE–ML-KEM instantiation.** We instantiate the analysis for certificate-authenticated TLS 1.3 ECDHE–ML-KEM 1-RTT. The instantiation gives auditable local obligations, target-specific HKDF

paths and exposure outcomes, and concrete bounds for the scoped preservation result. KEM-based authentication is expressible, but requires a separate local authentication-binding proof for the chosen mechanism.

1.2 Technical Overview



Technical roadmap.

The roadmap summarizes how the security claims are proved. The upper block contains the general proof steps, and the lower block gives the scoped TLS instantiation. Read left to right inside each block and then downward to the concrete TLS result. The secrecy path is driven by surviving confidentiality branches and KDF ancestry; the agreement path is driven by authentication witnesses and transcript binding. The two paths meet at the preservation core, namely Theorems 7.2 and 8.4.

The secrecy path starts from the branch-replacement game of Figure 2, the labelled Extract/Expand games of Figures 4 and 5, and the KDF target-hiding game of Figure 6. The branch game is the local assumption that permits a live ECDHE, KEM, or other confidentiality contribution to be replaced at its combiner or KDF injection point. The labelled Extract/Expand games capture the HKDF/PRF-style guarantee needed by labelled key schedules. Lemma 5.3 turns these primitive games into target hiding for a labelled KDF DAG. Lemma 5.4 then adds the branch side: once a satisfying exposure formula identifies a dominating refresh cut and selected branch witnesses, target hiding survives the adversary's allowed vertex and stage-key reveals. The output is Theorem 7.2, the stage-secrecy preservation theorem. Its binding and collision terms enter at the theorem level, because the tested key must be tied to the accepted transcript, stage, suite, labels, identities, and branch public values.

The agreement path is parallel but uses different witnesses. Agreement does not follow from a hidden branch secret; it requires authenticated material to be bound to the correct agreement vector. The roadmap bundles the stage-agreement interface of Figure 3, the stage-agreement experiment of Figure 8, and the local authentication games assumed for the authentication branches. Lemma 8.3 is the main authentication-to-agreement reduction: outside session-identifier collisions and binder-injectivity failures, an accepting fresh stage without a unique matching peer yields a break of a selected authentication branch. Theorem 8.4 lifts this reduction to stage-agreement preservation.

The selector-local cost node records the refinement in Section 10. Definitions 10.1 and 10.2 fix secrecy and agreement witnesses and admissible selectors, while Figure 10 defines the charged costs. This interface introduces no new cryptographic assumption; it only refines the accounting by charging the branch witnesses, KDF cut, and binding certificate selected for a fixed stage claim. Theorem 10.3 is the selector-local refinement of Theorem 7.2, and Theorem 10.4 is the corresponding refinement of Theorem 8.4.

The lower block applies the same steps to scoped TLS 1.3 ECDHE–ML-KEM 1-RTT, excluding PSK, 0-RTT, HelloRetryRequest, and client authentication. Lemma 9.1 identifies the selected surviving branch witness at the hybrid shared-secret input. Lemma 9.2 lists the TLS HKDF paths and the Extract/Expand counts for the handshake, application, exporter, and resumption targets. CertificateVerify binding (Figure 12), Finished binding (Figure 9), and transcript injectivity discharge the agreement side. Corollary 9.3 combines these obligations with Theorems 7.2 and 8.4 to state the scoped TLS preservation bounds.

2 Related Work

Multi-stage authenticated key exchange. Multi-stage AKE models were introduced to reason about handshakes that produce several keys with different purposes and security properties. TLS 1.3 analyses are the central example: the protocol derives early, handshake, application, exporter, and resumption material, and the proof must account for authentication level, forward secrecy, replay behavior, and key usage at each stage [1, 2]. These models motivate the stage-indexed view used here. Our contribution is orthogonal: we keep stages, but replace flat freshness predicates by formulas over branch exposure and KDF ancestry. For example, if the ECDHE branch is revealed while the ML-KEM branch remains hidden, a flat predicate either rejects the stage or hides the reason it survives; the formula $C_{\mathsf{dh}} \vee C_{\mathsf{kem}}$ records the surviving witness and the corresponding KDF cut.

Hybrid KEM and key-combiner security. Hybrid KEM combiners formalize the transition setting in which a classical and a post-quantum component are combined so that the result remains secure when at least one component withstands the adversary [3, 4]. Such results are indispensable for branch secrecy. They are not sufficient for staged AKE because the combiner output is only one input to a larger authenticated transcript and KDF graph. Our theorem uses branch-combiner security as a local assumption and proves that the resulting robustness survives stage derivation, reveals, and late corruption.

Hybrid authenticated key exchange. HAKE frameworks and protocols such as Muckle+ study end-to-end hybrid authenticated key exchange with classical, post-quantum, and QKD-style components [5]. KEM-authenticated variants, including Muckle#-style designs, KEMTLS-inspired protocols, and the AuthKEM TLS draft, show that authentication itself may be built from post-quantum KEM mechanisms rather than signatures [6, 7, 8]. These works are close to the authentication side of our model. Our concrete instantiation remains certificate-authenticated TLS; a KEM-authenticated instantiation requires a local authentication-binding game for the chosen mechanism.

Transcript binding and downgrade resistance. Transcript binding prevents a proof or authentication token generated in one context from being accepted in another. In hybrid protocols, the binding surface is larger than in a single-branch protocol. The binder must include not only identities and messages, but also algorithm suites, branch modes, branch public values, KDF labels, and stage indices. This is related to predicate-based treatments of authentication in key exchange [9]. For staged hybrid handshakes, omitting one of these fields can turn downgrade or branch substitution into an apparent match; the agreement theorem rules this out by requiring injectivity of the binding input modulo the intended matching relation.

Tightness and concurrent security. Tight multi-session AKE security is an important refinement because loose reductions may lose factors in the number of parties, sessions, or stages [10, 11]. The present framework does not claim full tight AKE security in that sense; such results usually require protocol-level tightness design. Instead, Section 10 proves a local refinement tailored to the calculus: once a stage formula is satisfied, the loss can be charged to the witnesses and KDF ancestry actually used for that stage. Tight local branch assumptions can be substituted into this accounting, but the theorem remains a preservation theorem, not a new tightly secure AKE construction.

Standardization-facing hybrid designs. Recent standardization-facing work makes the question timely. The IETF ECDHE–ML-KEM draft defines concrete TLS 1.3 hybrid groups combining ECDHE with ML-KEM, including X25519MLKEM768, SecP256r1MLKEM768, and SecP384r1MLKEM1024 [12]; ML-KEM itself is standardized in FIPS 203 [14]. X-Wing gives a concrete X25519/ML-KEM-768 hybrid KEM and proves the security of the hybrid KEM construction [13]. These works fix important branch-level objects. The remaining protocol-layer question is whether the derived TLS secrets and agreement claims survive the actual transcript, KDF, Reveal, and Corrupt structure of a staged handshake.

Comparison with adjacent lines. Table 1 compares the security tasks treated by adjacent lines of work. The columns are phrased as external proof obligations rather than as internal features of our notation.

The ∘ entries mark partial or protocol-specific support. The TLS-map entry for this work is scoped to certificate-authenticated TLS 1.3 with Finished-style transcript confirmation; KEM-authenticated variants require a separate local AuthBind theorem for the chosen mechanism.

Table 1: Compact comparison with adjacent work. Columns record externally visible proof tasks: multiple keys from one handshake; surviving hybrid branch reveals; explicit stage-key reveals; selective or late corruption; downgrade or branch-substitution binding; and a concrete TLS ECDHE–ML-KEM mapping. Symbols: ✓ means central support, ∘ means partial or protocol-specific support, and − means outside the main scope.

Line	Ref.	Keys	Br. reveal	Stage reveal	Late corr.	Down. subst.	TLS map
TLS 1.3 multi-stage AKE	[1, 2]	✓	−	✓	∘	∘	∘
Hybrid KEM combiners	[3, 4]	−	✓	−	−	−	−
X-Wing	[13]	−	✓	−	−	−	−
IETF ECDHE–ML-KEM TLS	[12]	∘	✓	∘	−	∘	✓
Muckle+/Muckle#	[5, 6]	∘	✓	∘	∘	∘	−
KEMTLS/-AuthKEM	[7, 8]	∘	∘	∘	∘	∘	∘
This work	–	✓	✓	✓	✓	✓	✓

3 Staged Hybrid AKE Syntax

This section fixes the proof data needed to decide which staged claims survive a given exposure pattern. A protocol instance is represented by confidentiality branches, authentication branches, transcript binders, a labelled derivation graph, and stage formulas. The abstraction is intentionally small; every omitted field must be encoded in the transcript binder or the theorem does not apply.

Definition 3.1 (Protocol interface)**.** A staged hybrid AKE protocol is a tuple

$$\Pi = (\mathcal{P}, \mathcal{C}, \mathcal{A}, G, \mathsf{Bind}, \mathsf{KDF}, \mathsf{Acc}, \Phi)$$

with the following components.

- $\mathcal{P}$ is the set of parties. Each local session is denoted $\pi = (P, s, r)$, where $P \in \mathcal{P}$, s is a local session handle, and $r \in \{\mathsf{init}, \mathsf{resp}\}$ is the role.
- $\mathcal{C} = \{C_1, \ldots, C_m\}$ is the set of confidentiality branches. A branch instance produces public data pub_C and either a secret contribution Z_C or $\perp$.
- $\mathcal{A} = \{A_1, \ldots, A_n\}$ is the set of authentication branches. A branch instance authenticates a binding string and either accepts, rejects, or remains unused at a stage.
- $G = (V, E)$ is a finite directed acyclic KDF graph. Vertices are branch secrets, intermediate secrets, or distinguished stage keys $K_{\pi,i}$ for $i \in [t]$.
- Bind maps public context and transcript state to binding strings $B_{\pi,i}$.
- $\mathsf{Acc}(\pi, i)$ is the deterministic acceptance predicate for stage i of session π.
- $\Phi = (\Phi_{\pi,i})$ assigns to each stage a monotone exposure formula.

Definition 3.2 (KDF graph)**.** A KDF graph $G = (V, E)$ is labelled if every non-source vertex v is associated with a label ℓ_v, a public context ctx_v, and an ordered predecessor vector $\mathsf{pred}(v) = (u_1, \ldots, u_q)$ such that

$$v = \mathsf{KDF}(\ell_v, \mathsf{ctx}_v, u_1, \ldots, u_q).$$

It is stage-labelled if every stage key $K_{\pi,i}$ is a distinguished vertex and no two distinguished vertices in the same session have the same label-context pair. For vertices u, v, write $u \preceq v$ if u is an ancestor of v and write $u \prec v$ for a strict ancestor.

Definition 3.3 (Binding input)**.** For every accepting session-stage pair (π, i), where $\pi = (P, s, r)$ and the intended peer is Q, the binding input is

$$B_{\pi,i} = (\mathsf{sid}_\pi, r, P, Q, \mathsf{suite}_{\pi,i}, i, (\mathsf{pub}_C)_{C \in \mathcal{C}}, (\mathsf{pub}_A)_{A \in \mathcal{A}}, \mathsf{TH}_{\pi,i}).$$

The theorem requires that every field that can alter the key, peer identity, role, authentication decision, branch policy, or KDF path is present in $B_{\pi,i}$.

Figure 1 summarizes the generic construction interface used by the formal model.

Construction: Generic Staged Hybrid AKE

Parameters: branch sets $\mathcal{C}, \mathcal{A}$, transcript binder Bind, stage-labelled KDF graph G, acceptance predicate Acc, and formulas $\Phi_{\pi,i}$.
Input: public transcript prefix, local role role_π, peer identifiers (P, Q), and suite identifier suite.
State: branch records (pub_C, Z_C), authentication records $(\mathsf{pub}_A, \mathsf{aux}_A)$, binding strings $B_{\pi,i}$, and KDF vertices $v \in V(G)$.
Procedure:

1. Fix the branch vector, role, peer identities, suite, and stage labels for the session π.
2. For every confidentiality branch $C \in \mathcal{C}$, execute the branch algorithm. If it accepts, record (pub_C, Z_C); otherwise record $\perp$.
3. For every candidate stage i, compute the canonical binding input
$$B_{\pi,i} = (\mathsf{sid}_\pi, \mathsf{role}_\pi, P, Q, \mathsf{suite}_{\pi,i}, i, (\mathsf{pub}_C)_C, (\mathsf{pub}_A)_A, \mathsf{TH}_{\pi,i}).$$
4. Run each authentication branch required at stage i on its domain-separated binding input. Reject the stage if a required branch rejects.
5. Evaluate G in topological order. For every derived vertex v, compute
$$v \leftarrow \mathsf{KDF}(\ell_v, \mathsf{ctx}_v, \mathsf{pred}(v)).$$
6. If $\mathsf{Acc}(\pi, i) = 1$, release $K_{\pi,i}$ to the local protocol state. Otherwise set the stage output to $\perp$.

Output: accepted stage keys $K_{\pi,i}$ and public transcript data.

Figure 1: Generic staged hybrid AKE syntax. The construction records the branch vector, canonical binding inputs, and stage-labelled KDF graph used by the preservation theorems.

4 Exposure Algebra and Freshness Formulas

The adversary controls scheduling and delivery. It may issue Send queries, reveal accepted stage keys, expose selected branch state, corrupt long-term authentication capabilities, and corrupt local session state before or after acceptance. Freshness is evaluated in the final exposure state, not only at the time of the test query. This convention captures late corruptions without adding separate definitions for every protocol family.

Definition 4.1 (Exposure events)**.** An exposure event has one of the following forms. The timestamp τ is the position of the event in the adversarial execution.

- $\mathsf{expC}(C, \pi, i, \tau)$ exposes the branch secret or randomness of confidentiality branch C.
- $\mathsf{expA}(A, P, \tau)$ exposes the authentication capability or credential for branch A.
- $\mathsf{expV}(v, \tau)$ reveals KDF vertex v, including a stage key.
- $\mathsf{expS}(\pi, \tau)$ exposes the full local state of session π.
- $\mathsf{late}(P, \tau)$ corrupts party P after at least one acceptance event.

Definition 4.2 (Exposure state)**.** An exposure state is a tuple
$$E = (E_C, E_A, E_V, E_S, E_L, \tau_E)$$
where E_C, E_A, E_V, E_S, E_L are the sets of confidentiality-branch exposures, authentication exposures, KDF-vertex reveals, full-session exposures, and late corruptions, and τ_E is the current execution time. The final exposure state is the state after the adversary halts.

Definition 4.3 (Exposure atoms)**.** For a session-stage pair (π, i), the atomic predicates are evaluated with respect to the final exposure state E, the local session state, and the public transcript.

- $\mathsf{Unexp}(C, \pi, i)$ holds if no event in E_C or E_S exposes the secret contribution of C used for $K_{\pi,i}$.
- $\mathsf{AuthFr}(A, \pi, i)$ holds if the branch-A authentication capability needed for $B_{\pi,i}$ was not exposed before the relevant accepting authentication event.

- $\mathsf{KSafe}(\pi, i)$ holds if the KDF ancestry of $K_{\pi,i}$ is safe according to Definition 5.1.
- $\mathsf{NotRev}(K_{\pi,i})$ holds if the target stage key itself is not revealed in E_V and is not computable from exposed full-session state.
- $\mathsf{Inj}_{\mathsf{sec}}(\pi, i)$ holds if the public transcript fields that determine the target key, KDF labels, suite, role, stage, and branch inputs are injectively represented according to Definition 6.1.
- $\mathsf{Inj}_{\mathsf{agr}}(\pi, i)$ holds if the authenticated agreement vector is injectively represented according to Definition 6.1.

Definition 4.4 (Branch-formula freshness)**.** For every session-stage pair (π, i), the protocol specifies a monotone formula

$$\Phi_{\pi,i} = \Phi^{\mathsf{conf}}_{\pi,i} \wedge \Phi^{\mathsf{auth}}_{\pi,i} \wedge \Phi^{\mathsf{kdf}}_{\pi,i} \wedge \Phi^{\mathsf{bind}}_{\pi,i} \wedge \mathsf{NotRev}(K_{\pi,i})$$

over exposure atoms. The stage (π, i) is fresh in exposure state E, written $E \models \mathsf{Fresh}(\pi, i)$, if $E \models \Phi_{\pi,i}$.

Monotonicity is with respect to exposure: if $E \subseteq E'$ and $E' \models \Phi_{\pi,i}$, then $E \models \Phi_{\pi,i}$. Thus additional exposure can only destroy freshness. The formula itself may express disjunctions and thresholds. Hybrid OR confidentiality is

$$\Phi^{\mathsf{conf}}_{\pi,i} = \mathsf{Unexp}(C_{\mathsf{ecdh}}, \pi, i) \vee \mathsf{Unexp}(C_{\mathsf{mlkem}}, \pi, i),$$

whereas strict hybrid authentication is

$$\Phi^{\mathsf{auth}}_{\pi,i} = \mathsf{AuthFr}(A_{\mathsf{sig}}, \pi, i) \wedge \mathsf{AuthFr}(A_{\mathsf{kemauth}}, \pi, i).$$

A threshold policy is written $\mathsf{Thr}_{t,n}(X_1, \ldots, X_n)$ and is true exactly when at least t atoms are true.

Definition 4.5 (Selective and late corruption semantics)**.** A corruption before the accepting authentication event of (π, i) falsifies every authentication atom whose capability it exposes. A corruption after acceptance falsifies only those atoms whose local assumption does not provide post-acceptance security. For a forward-secure confidentiality branch, $\mathsf{late}(P, \tau)$ with τ after erasure of the branch secret does not falsify $\mathsf{Unexp}(C, \pi, i)$; for a non-forward-secure branch it does. These effects are specified by the local branch-exposure relation exported by the branch assumption.

The local branch-replacement assumption used by each confidentiality branch is the experiment in Figure 2.

Experiment: Branch-Replacement $\mathsf{BRRep}^{\beta}_{B,\rho}(1^\lambda)$

Purpose: localize the proof obligation for one confidentiality branch B under exposure relation ρ.

Experiment: The challenger runs the public branch messages and admissible protocol context honestly, and samples a bit β. If $\beta = 0$, the value injected by branch B into the combiner or KDF is the protocol value Z_B. If $\beta = 1$, that injected value is replaced by an independent string of the same length at the injection point only. The adversary receives all public transcript data and may issue exposure queries allowed by ρ; a query that exposes the protected branch contribution aborts. The adversary outputs β', and the experiment returns 1 iff $\beta' = \beta$.

Advantage:

$$\mathsf{Adv}^{\mathsf{br-rep}}_{B,\rho}(\mathcal{B}) = \left| \Pr[\beta' = \beta] - \frac{1}{2} \right|.$$

Instantiation reading. For ECDHE, ρ is discharged by the appropriate Diffie–Hellman replacement assumption together with extraction at the combiner input. For ML-KEM, ρ is discharged by the KEM shared-secret replacement property under decapsulation correctness and the allowed reveal pattern. The preservation theorem never assumes that a raw ECDH group element is itself a uniform bit string.

Figure 2: Branch-replacement experiment for one confidentiality branch. The game defines the local assumption charged when a branch contribution is replaced at its combiner or KDF injection point.

Figure 3 defines the agreement interface that is later refined by the full stage-agreement experiment.

5 KDF Ancestry and HKDF-Style Refresh Separation

A KDF graph is not secure merely because labels are distinct. If a revealed ancestor deterministically derives a descendant, then the descendant is exposed. The proof therefore separates two obligations. First, a local labelled-PRF/extractor assumption states what is required from the HKDF-style primitive. Second, a graph theorem applies that assumption to a target vertex separated from revealed ancestry by a protected cut. Branch-replacement games

Experiment: Stage-Agreement Interface $\mathsf{Exp}^{\mathsf{sa-int}}_{\Pi,\mathcal{A}}(1^\lambda)$

Input: security parameter 1^λ, protocol Π, and adversary $\mathcal{A}$.
Oracles: Send, RevealStage, RevealBranch, selective Corrupt, late Corrupt, and one Declare query.

Experiment:

1. The challenger initializes parties, long-term credentials, branch state, binders, KDF graphs, and $E \leftarrow \emptyset$.
2. The adversary adaptively schedules sessions and issues oracle queries. The challenger records every exposure event in E.
3. On $\mathsf{Declare}(\pi, i)$, the challenger requires that (π, i) is an honest accepting stage.
4. Let $B_{\pi,i}$ be the accepted binding input and let Q be the claimed peer. The experiment outputs 1 iff the final state satisfies $E \models \Phi^{\mathsf{auth}}_{\pi,i} \wedge \Phi^{\mathsf{bind}}_{\pi,i}$ and there is no unique peer session π' at Q accepting the dual role and same stage with matching binding input.

Advantage:

$$\mathsf{Adv}^{\mathsf{sa-int}}_{\Pi}(\mathcal{A}) = \Pr[\mathsf{Exp}^{\mathsf{sa-int}}_{\Pi,\mathcal{A}} = 1].$$

Figure 3: Stage-agreement interface experiment. The game isolates the event that an accepting fresh stage has no unique matching peer.

are charged outside the graph theorem, at the points where branch values enter the schedule. HKDF is standardized in RFC 5869, and TLS 1.3 specializes it through HKDF-Expand-Label and Derive-Secret in RFC 8446 [15, 16].

Figures 4 and 5 define the primitive games for labelled Extract and Expand, and Figure 6 defines the derived target-hiding game used in Lemmas 5.3 and 5.4.

Definition 5.1 (Dominating refresh cut)**.** Let $G = (V, E)$ be the KDF graph of session π, let $K_{\pi,i}$ be a target stage key, and let $X_E \subseteq V$ be the set of revealed vertices and vertices computable from exposed full-session state. A set $R \subseteq V$ is a dominating refresh cut for $K_{\pi,i}$ under E if:

1. every directed path from any $x \in X_E$ with $x \prec K_{\pi,i}$ to $K_{\pi,i}$ contains some $r \in R$;
2. each $r \in R$ is either already ideal by a branch-replacement hybrid or is derived by an HKDF-Extract call whose protected predecessor is already ideal;
3. each KDF call on a path from R to $K_{\pi,i}$ uses a label-context pair reserved for that edge of the graph; and
4. neither any $r \in R$ nor $K_{\pi,i}$ is revealed or computable from exposed full-session state.

The predicate $\mathsf{KSafe}(\pi, i)$ is true if either X_E contains no strict ancestor of $K_{\pi,i}$, or a dominating refresh cut for $K_{\pi,i}$ exists.

Game: Labelled Extract Replacement $\mathsf{Exp}^{\mathsf{lExt}}_{\mathsf{Extract},\mathcal{A}}(1^\lambda, b)$

Input: a domain-separated label-context pair (ℓ, ctx), a salt/input key s, a protected predecessor position, and bit b.

Experiment:

1. The challenger samples all public labels and contexts and gives the adversary oracle access to $\mathsf{Extract}$ on non-reserved label-context pairs.
2. The protected predecessor z is an independent ideal string. If $b = 0$, the reserved output is $s' = \mathsf{Extract}(\ell, \mathsf{ctx}, s, z)$. If $b = 1$, the reserved output is an independent key of the same length.
3. The adversary receives s', all public data, and all non-reserved oracle answers, and outputs b'.

The advantage is denoted $\mathsf{Adv}^{\mathsf{lExt}}_{\mathsf{Extract}}(\mathcal{A})$.

Figure 4: Labelled Extract replacement game. This is the local primitive game used for HKDF-Extract nodes in the KDF graph.

Definition 5.2 (Labelled HKDF/PRF interface)**.** A labelled KDF graph is implemented by domain-separated calls

$$s' = \mathsf{Extract}(\ell, \mathsf{ctx}, s, z), \qquad k = \mathsf{Expand}(\ell, \mathsf{ctx}, s, L),$$

where ℓ contains a stage-specific label and ctx contains transcript-dependent context. The local KDF assumption consists exactly of the two replacement games above. These games are local: they do not contain a graph target and do not assume KDF-DAG target hiding. For a graph G, cut R, and target K, let $N_E(G, R, K)$ and $N_X(G, R, K)$

Game: Labelled Expand Replacement $\mathsf{Exp}^{\mathsf{lExp}}_{\mathsf{Expand},\mathcal{A}}(1^\lambda, b)$

Input: a secret key s that is independent of the adversary's view, a reserved label-context family L, and bit b.

Experiment:

1. The challenger gives the adversary all public labels and contexts and oracle access to $\mathsf{Expand}(s, \ell, \mathsf{ctx}, \cdot)$ on non-reserved label-context pairs.
2. For each reserved pair $(\ell, \mathsf{ctx}) \in L$, if $b = 0$ the challenger answers with $\mathsf{Expand}(s, \ell, \mathsf{ctx}, L)$; if $b = 1$ it answers with an independent string of length L.
3. The adversary outputs b'.

The advantage is denoted $\mathsf{Adv}^{\mathsf{lExp}}_{\mathsf{Expand}}(\mathcal{A})$.

Figure 5: Labelled Expand replacement game. This is the local primitive game used for HKDF-Expand nodes in the KDF graph.

be the numbers of reserved Extract and Expand invocations, respectively, on the subgraph reachable from R and leading to K. The graph hybrid charges

$$N_E(G, R, K)\mathsf{Adv}^{\mathsf{lExt}}_{\mathsf{Extract}} + N_X(G, R, K)\mathsf{Adv}^{\mathsf{lExp}}_{\mathsf{Expand}} + \epsilon_{\mathsf{lab}}.$$

Thus target hiding is derived by replacing graph vertices one primitive call at a time.

Game: KDF Target Hiding $\mathsf{Exp}^{\mathsf{kdf-hide}}_{G,K,R,\mathcal{A}}(1^\lambda, b)$

Input: a labelled KDF graph G, target vertex K, exposed vertices X_E, a dominating cut R, and bit b.

Experiment:

1. The challenger samples all public labels, contexts, transcript hashes, and all source vertices not protected by R.
2. Every $r \in R$ is set to an independent ideal key of the correct length. The challenger then evaluates every non-target vertex of G honestly using the labelled Extract/Expand implementation.
3. The adversary receives all public data, every vertex in X_E, and oracle access on non-reserved label-context pairs.
4. If $b = 0$, the challenger gives the real target value K computed in the graph. If $b = 1$, it gives an independent uniform string of the same length.
5. The adversary outputs b'.

The advantage is $|\Pr[b' = b] - 1/2|$ and is denoted $\mathsf{Adv}^{\mathsf{kdf-hide}}_{G,K,R}$.

Figure 6: KDF target-hiding game. Protected cut vertices are idealized first, and the game distinguishes the real target node from an independent random string.

Lemma 5.3 (KDF-DAG target hiding from labelled primitive games)**.** *Let G be a labelled KDF graph, K a target vertex, and R a dominating cut for K. Assume all vertices in R are ideal and independent of the adversary's view, all reserved label-context pairs on paths from R to K are distinct, and the adversary receives only non-target vertices permitted by the exposure state. Let $N_E(G, R, K)$ and $N_X(G, R, K)$ be the numbers of reserved Extract and Expand calls on the subgraph reachable from R and leading to K. Then*

$$\mathsf{Adv}^{\mathsf{kdf-hide}}_{G,K,R}(\mathcal{A}) \le N_E(G, R, K)\Delta_E + N_X(G, R, K)\Delta_X + \epsilon_{\mathsf{lab}},$$

where $\Delta_E = \mathsf{Adv}^{\mathsf{lExt}}_{\mathsf{Extract}}(\mathcal{B}_E)$ and $\Delta_X = \mathsf{Adv}^{\mathsf{lExp}}_{\mathsf{Expand}}(\mathcal{B}_X)$.

Proof. Topologically order the reserved KDF calls on all paths from R to K. Since the cut vertices are ideal, the first reserved call on each path is keyed or refreshed by a value independent of the adversary's view. If the call is an Extract call, replace its output by the ideal output using $\mathsf{Exp}^{\mathsf{lExt}}_{\mathsf{Extract}}$; if it is an Expand call, replace the corresponding reserved output family using $\mathsf{Exp}^{\mathsf{lExp}}_{\mathsf{Expand}}$. Continue along the topological order. At each step, the current reserved call is keyed by a value that has already been made independent of the adversary's view, and its label-context pair has not appeared in a non-reserved oracle query or earlier reserved edge except with probability ϵ_{lab}. After all reserved calls leading to K have been replaced, the target vertex is independent of the adversary's view and equals the ideal target of the $b = 1$ game. Summing the $N_E(G, R, K)$ Extract hybrids, the $N_X(G, R, K)$ Expand hybrids, and the label-collision event gives the bound. □

Lemma 5.4 (Labelled KDF-DAG refresh separation)**.** *Let G be a stage-labelled KDF graph. Fix a target $K_{\pi,i}$ and final exposure state E. Suppose $\mathsf{KSafe}(\pi, i)$ holds via a dominating refresh cut R, and suppose a fixed selector identifies, for each $r \in R$ that is not already ideal, a branch contribution $C(r)$ and exposure relation $\rho(r)$ whose replacement makes r ideal. Then*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{refresh}}_{G,K_{\pi,i},R}(\mathcal{A}) &\le \sum_{r\in R} \mathsf{Adv}^{\mathsf{br-rep}}_{C(r),\rho(r)}(\mathcal{B}_r) + \mathsf{Adv}^{\mathsf{kdf-hide}}_{G,K_{\pi,i},R}(\mathcal{B}_K)\\ &\le \sum_{r\in R} \mathsf{Adv}^{\mathsf{br-rep}}_{C(r),\rho(r)}(\mathcal{B}_r) + N_E(G,R,K_{\pi,i})\mathsf{Adv}^{\mathsf{IExt}}_{\mathsf{Extract}}(\mathcal{B}_E)\\ &\quad + N_X(G,R,K_{\pi,i})\mathsf{Adv}^{\mathsf{IExp}}_{\mathsf{Expand}}(\mathcal{B}_X) + \epsilon_{\mathsf{lab}}.\end{aligned}$$

Proof. First apply the selected branch-replacement hybrids at the cut. These hybrids are local because the selector fixes the branch and exposure relation before the reduction instance is chosen; if a proof cannot determine the selected cut online, it pays the selector guessing factor stated in the main theorem. After the replacements, every cut vertex in R is ideal. Lemma 5.3 then replaces the target descendant by an independent string. The first line records the modular decomposition into branch replacement and graph hiding; the second line expands graph hiding into labelled PRF/extractor security. □

Corollary 5.5 (Stage-reveal separation)**.** *If $K_{\pi,j}$ is revealed and $K_{\pi,j} \prec K_{\pi,i}$, then $K_{\pi,i}$ can remain fresh only if there is a dominating refresh cut between $K_{\pi,j}$ and $K_{\pi,i}$ and $\mathsf{NotRev}(K_{\pi,i})$ holds. If no such cut exists, $\mathsf{KSafe}(\pi, i)$ is false and the stage-secrecy theorem gives no claim for $K_{\pi,i}$.*

Proof. If the revealed key $K_{\pi,j}$ lies on a path to $K_{\pi,i}$, Definition 5.1 requires a cut on every exposed-to-target path; if it is only a sibling or shares an ancestor, the claim depends on whether that shared ancestor or a path node is exposed. The freshness formula separately requires the target itself not to be revealed. If either condition fails, $E \not\models \Phi_{\pi,i}$. □

6 Transcript Binding and Agreement

For a session-stage pair (π, i) define

$$B_{\pi,i} = (\mathsf{sid}, \mathsf{role}, P, Q, \mathsf{suite}_i, i, (\mathsf{pub}_C)_C, (\mathsf{pub}_A)_A, \mathsf{TH}_i).$$

The binder may be implemented by signatures, MACs, or confirmation values; KEM-authenticated binders are expressible only after the corresponding local authentication-binding theorem is supplied. The abstraction requires injectivity of the accepted binding input, not a particular implementation.

Definition 6.1 (KDF and agreement injectivity)**.** We separate the injectivity needed for secrecy from the injectivity needed for agreement. The predicate $\mathsf{Inj}_{\mathsf{sec}}(\pi, i)$ holds if equality of the public key-derivation representation for two accepting stages implies equality of the fields that determine the target key and its KDF ancestry: suite, role, stage label, selected group, active branch-public values, KDF labels, transcript hash used as KDF context, and session identifier, except with the stated collision probability. The predicate $\mathsf{Inj}_{\mathsf{agr}}(\pi, i)$ holds if equality of the authenticated agreement vector additionally implies matching peer identities, dual roles, equal authentication mode and authentication public values, and equal transcript hash. Thus an omitted field that affects key derivation falsifies $\mathsf{Inj}_{\mathsf{sec}}$, while an omitted field that only affects authenticated peer interpretation falsifies $\mathsf{Inj}_{\mathsf{agr}}$.

Lemma 6.2 (Authentication-to-agreement)**.** *Assume every required authentication branch for (π, i) is fresh in E and the binder is stage-injective. If an adversary wins the stage-agreement game at (π, i), then there exists an efficient adversary that either forges one required authentication branch under its allowed corruption pattern or violates stage-injectivity of the binder.*

Proof. Let (π, i) be the first honest accepting stage without a unique matching peer. Since the binder is stage-injective, any honest peer accepting the same binding input must be unique and matching. Therefore either no honest party generated the authenticated material for $B_{\pi,i}$, or the authenticated material was generated for a different input $B' \neq B_{\pi,i}$ and accepted for $B_{\pi,i}$. In the first case, the adversary has produced accepting authentication material without access to the required fresh authentication capability, yielding a forgery or impersonation attack against one authentication branch. In the second case, acceptance of the same authentication material for distinct binding inputs violates stage-injectivity or the collision resistance of the transcript representation. The reduction guesses the first failing branch and forwards its authentication challenge; the loss is the number of required authentication branches for the stage. □

7 Stage-Secrecy Preservation

This section gives the main secrecy theorem. The theorem is stated for one distinguished test query and then lifted to the usual at-most-one-Test and many-session experiment by guessing the tested session and stage. The proof is closed: every transition is justified by a named local assumption, a KDF-refresh instance, or a collision event.

7.1 Secrecy Experiment

Let Sess be the set of session-stage pairs created during an execution. For $(\pi, i) \in \mathsf{Sess}$, let $K_{\pi,i}$ be the stage key, G_π the session KDF graph, and E the final exposure state. The experiment rejects a test query unless the tested stage accepts and remains formula-fresh in the final exposure state. Final-state freshness is essential: it gives the adversary the full power of late corruptions and post-test reveals, while preserving exactly those claims encoded in $\Phi_{\pi,i}$. Figure 7 gives the resulting two-world experiment.

Experiment: Stage Secrecy $\mathsf{Exp}^{\mathsf{ss}}_{\Pi,\mathcal{A}}(1^\lambda, b)$

Input: security parameter 1^λ, protocol Π, adversary $\mathcal{A}$, and challenge bit $b \in \{0,1\}$.
Oracles: NewSession, Send, RevealStage, RevealBranch, RevealVertex, selective Corrupt, late Corrupt, and one Test query.

Experiment:

1. The challenger samples long-term keys, initializes parties, and sets $E \leftarrow \emptyset$.
2. The adversary adaptively issues oracle queries. The challenger answers according to Π and records the corresponding exposure events in E.
3. On the unique query $\mathsf{Test}(\pi^*, i^*)$, the challenger returns $\perp$ if (π^*, i^*) has not accepted. Otherwise it stores (π^*, i^*) and returns K_{π^*,i^*} if $b = 0$, and an independent uniform string U of equal length if $b = 1$.
4. The adversary continues issuing all non-test queries. The challenger continues updating E.
5. At termination, the experiment returns 0 if the final state does not satisfy $E \models \Phi_{\pi^*,i^*}$. Otherwise it returns 1 iff the adversary outputs $b' = b$.

Advantage:

$$\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi}(\mathcal{A}) = \left|\Pr[\mathsf{Exp}^{\mathsf{ss}}_{\Pi,\mathcal{A}}(1^\lambda, 0) = 1] - \Pr[\mathsf{Exp}^{\mathsf{ss}}_{\Pi,\mathcal{A}}(1^\lambda, 1) = 1]\right|.$$

Figure 7: Two-world stage-secrecy experiment. A Test query is meaningful only when the final exposure state satisfies the tested stage formula, including target non-reveal.

The experiment permits reveals after the test. A post-test reveal invalidates the test only if the final formula is false. Thus the game does not need a separate "clean session" convention; the branch formula is the convention.

7.2 Dominating Cuts for Tested Stages

The secrecy theorem uses a witness extracted from the formula. A stage formula is called secrecy complete if, whenever $E \models \Phi_{\pi,i}$, the following objects exist:

1. a nonempty set $W_{\pi,i,E}$ of confidentiality atoms sufficient for $\Phi^{\mathsf{conf}}_{\pi,i}$;
2. a dominating refresh cut $R_{\pi,i,E}$ for $K_{\pi,i}$ in the graph G_π;
3. a map $\rho : R_{\pi,i,E} \to W_{\pi,i,E}$ such that each cut vertex r receives as protected predecessor the branch secret named by $\rho(r)$; and
4. a binding-injectivity certificate for every public value that determines the selected branch inputs and KDF labels.

This condition is syntactic for protocols that explicitly mark branch-injection vertices. It prevents a formula from claiming that a branch is unexposed while the KDF graph never injects that branch on every path to the tested key.

Definition 7.1 (Secrecy-complete stage formula)**.** A stage formula $\Phi_{\pi,i}$ is secrecy complete for a protocol instance if, for every reachable final exposure state E with $E \models \Phi_{\pi,i}$, there exists a witness (W, R, ρ) satisfying the four conditions above. The protocol is secrecy complete if every testable stage formula is secrecy complete.

7.3 Main Secrecy Theorem

Theorem 7.2 (Stage-secrecy preservation)**.** *Let Π be a secrecy-complete staged hybrid AKE protocol with branch-formula freshness Φ, fixed admissible secrecy selectors, stage-labelled KDF graphs, and a stage-injective transcript binder. Assume the following local properties.*

1. *For every confidentiality branch $C \in \mathcal{C}$, the injected branch contribution satisfies the branch-replacement game under the exposure relation used by $\mathsf{Unexp}(C, \cdot, \cdot)$.*

2. *Extract/Expand satisfies the labelled PRF/extractor interface of Definition 6, and Lemma 5.3 applies to every target-hiding instance selected by a fresh tested stage.*

3. *The transcript representation and session identifiers are collision-free except with probability ϵ_{sid}, and the binder is stage-injective except with advantage $\mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}$.*

Let q_{sess} be the number of sessions, q_{st} the maximum number of stages per session, q_{cut} the maximum size of a dominating refresh cut, and g_{sel} the selector-image guessing factor. If the selector image is online-determined, in the sense of Definition 10.2, before the relevant branch/KDF challenge is embedded, then $g_{\mathsf{sel}} = 1$; otherwise g_{sel} is the number of admissible selector images for the guessed tested pair. For every adversary $\mathcal{A}$,

$$\begin{aligned}\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi}(\mathcal{A}) \le\ & q_{\mathsf{sess}} q_{\mathsf{st}} \cdot \epsilon_{\mathsf{sid}} + q_{\mathsf{sess}} q_{\mathsf{st}} \cdot \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}(\mathcal{B}_B) \\ & + q_{\mathsf{sess}} q_{\mathsf{st}} g_{\mathsf{sel}} q_{\mathsf{cut}} \cdot \max_{C \in \mathcal{C}} \mathsf{Adv}^{\mathsf{br-rep}}_{C}(\mathcal{B}_C) \\ & + q_{\mathsf{sess}} q_{\mathsf{st}} g_{\mathsf{sel}} \cdot \mathsf{Adv}^{\mathsf{kdf-hide}}_{G,K,R}(\mathcal{B}_K).\end{aligned}$$

Equivalently, for a fixed tested pair (π^*, i^*) and fixed selector output $\mathsf{Sel}^{\mathsf{sec}}_{\pi^*,i^*}(E) = (W, R, \rho, W_T)$,

$$\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi,\pi^*,i^*}(\mathcal{A}) \le \epsilon_{\mathsf{sid}} + \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}(\mathcal{B}_B) + \sum_{r \in R} \mathsf{Adv}^{\mathsf{br-rep}}_{C(r),\rho(r)}(\mathcal{B}_r) + \mathsf{Adv}^{\mathsf{kdf-hide}}_{G_\pi, K_{\pi^*,i^*}, R}(\mathcal{B}_K).$$

The unconditioned bound is obtained by the displayed session-stage and selector-image guessing factors.

Proof. The proof proceeds by the standard hybrid sequence: guess the tested pair, remove session and binder collisions, apply the fixed selector to determine the admissible cut and branch witnesses, replace the selected branch inputs, and invoke KDF target hiding for the selected cut. The selector-image guessing factor is paid before embedding any challenge. Throughout the hybrids the reduction maintains three invariants: public transcripts and non-selected sessions are distributed as in the real execution; every revealed non-target vertex is either honestly computable in the current hybrid or makes the final freshness formula false; and a post-test reveal is answered normally but invalidates the test whenever it falsifies $\mathsf{NotRev}(K_{\pi^*,i^*})$ or $\mathsf{KSafe}(\pi^*, i^*)$. Appendix B gives the full game sequence and reductions. □

Corollary 7.3 (OR-hybrid stage secrecy)**.** *Let a stage use two confidentiality branches C_0, C_1 and let*

$$\Phi^{\mathsf{conf}}_{\pi,i} = \mathsf{Unexp}(C_0, \pi, i) \vee \mathsf{Unexp}(C_1, \pi, i).$$

If the KDF graph injects both branch secrets before every path to $K_{\pi,i}$ and at least one of the two branch atoms remains true in the final exposure state, then the tested stage key is pseudorandom up to the branch advantage of the surviving branch, the KDF target-hiding advantage, and the binding/collision terms of Theorem 7.2.

Proof. Choose any surviving branch atom $\mathsf{Unexp}(C_b, \pi, i)$ that satisfies the disjunction. The injection condition supplies a dominating refresh witness containing the C_b injection point, so Theorem 7.2 applies to that selected witness. Appendix B.3 gives the full witness-selection proof. □

Corollary 7.4 (Threshold-hybrid stage secrecy)**.** *Let*

$$\Phi^{\mathsf{conf}}_{\pi,i} = \mathsf{Thr}_{t,n}(\mathsf{Unexp}(C_1, \pi, i), \ldots, \mathsf{Unexp}(C_n, \pi, i)).$$

If every satisfying set of t branch atoms determines a dominating refresh cut whose protected predecessors are exactly those surviving branches, then stage secrecy reduces to the sum of the t selected branch advantages, the KDF target-hiding advantage, and the binding/collision terms.

Proof. A satisfying threshold formula determines at least t unexposed branch atoms. The hypothesis turns some such t-set into a dominating refresh witness, and Theorem 7.2 applies to that witness. Appendix B.3 gives the full selected-set proof. □

8 Stage-Agreement Preservation

Secrecy and agreement use different parts of the formula. A stage key may remain secret because one confidentiality branch survives, but agreement can still fail if the authenticated transcript omits a branch public value, a suite identifier, a role label, or a stage label. The agreement theorem therefore uses the authentication, binding, and late-corruption components of the formula. It gives a reusable condition for ruling out downgrade, branch substitution, authentication-branch replay, role reversal, and stage confusion. The stage-agreement experiment is shown in Figure 8.

8.1 Matching and Agreement Freshness

For an accepting session-stage pair (π, i) define the agreement vector

$$B^{\mathsf{agr}}_{\pi,i} = (\mathsf{sid}_\pi, \mathsf{role}_\pi, \mathsf{id}^{\mathsf{self}}_\pi, \mathsf{id}^{\mathsf{peer}}_\pi, \mathsf{suite}_\pi, i, P^{\mathsf{conf}}_{\pi,i}, P^{\mathsf{auth}}_{\pi,i}, \mathsf{TH}_{\pi,i}),$$

where $P^{\mathsf{conf}}_{\pi,i}$ contains the public values that select and instantiate the confidentiality branches, and $P^{\mathsf{auth}}_{\pi,i}$ contains the public values that select and instantiate the authentication branches. The vector is not a new protocol message. It is the canonical input that must be covered by the transcript binder and by the authenticated material used at stage i.

Definition 8.1 (Stage matching)**.** Let (π, i) and (π', i') be accepting stage instances. We write

$$\mathsf{Match}((\pi, i), (\pi', i')) = 1$$

if $i = i'$, the roles are dual, the peer identities are swapped, the suites and branch-public vectors are equal, the transcript hashes are equal, and the session identifiers are equal. The match is unique if there is at most one such peer instance for (π, i).

The agreement part of a stage formula is

$$\Phi^{\mathsf{agr}}_{\pi,i} = \Phi^{\mathsf{auth}}_{\pi,i} \wedge \Phi^{\mathsf{bind}}_{\pi,i} \wedge \Phi^{\mathsf{late}}_{\pi,i}.$$

The atom $\Phi^{\mathsf{auth}}_{\pi,i}$ describes which authentication branches must be uncompromised before the stage accepts. The atom $\Phi^{\mathsf{bind}}_{\pi,i}$ states that $B^{\mathsf{agr}}_{\pi,i}$ is injectively represented in the transcript binder. The atom $\Phi^{\mathsf{late}}_{\pi,i}$ states that corruptions after acceptance do not retroactively invalidate the agreement claim, except when the local authentication notion explicitly gives the adversary retroactive signing, decapsulation, or binder-production power.

Definition 8.2 (Agreement-complete formula)**.** A formula $\Phi^{\mathsf{agr}}_{\pi,i}$ is agreement complete for a stage if, whenever a reachable exposure state E satisfies $E \models \Phi^{\mathsf{agr}}_{\pi,i}$, there is a nonempty witness set $W^{\mathsf{auth}}_{\pi,i,E}$ of authentication atoms such that: (i) the Boolean authentication formula is satisfied by $W^{\mathsf{auth}}_{\pi,i,E}$; (ii) every accepting authentication object for stage i verifies over $B^{\mathsf{agr}}_{\pi,i}$; (iii) every atom in $W^{\mathsf{auth}}_{\pi,i,E}$ corresponds to a local authentication security game in which the adversary has not obtained the capability forbidden by that atom before acceptance; and (iv) the binder representation of $B^{\mathsf{agr}}_{\pi,i}$ is injective except for the explicit binding-collision event.

8.2 Agreement Experiment

Experiment: Stage Agreement $\mathsf{Exp}^{\mathsf{sa}}_{\Pi,\mathcal{A}}(1^\lambda)$

Input: security parameter 1^λ, protocol Π, and adversary $\mathcal{A}$.
Oracles: NewSession, Send, RevealBranch, RevealStage, selective Corrupt, late Corrupt, and one Declare query.

Experiment:

1. The challenger initializes all parties, long-term credentials, branch states, KDF graphs, transcript binders, and an exposure state $E \leftarrow \emptyset$.
2. $\mathcal{A}$ adaptively issues oracle queries. The challenger returns the requested protocol response or exposure material and updates E.
3. On $\mathsf{Declare}(\pi, i)$, the challenger returns 0 unless (π, i) is an honest accepting stage. Otherwise it records the declared stage.
4. Let $\mathsf{BadMatch}$ be the event that no unique accepting stage (π', i) satisfies $\mathsf{Match}((\pi, i), (\pi', i)) = 1$.
5. The experiment returns 1 iff the final exposure state satisfies $E \models \Phi^{\mathsf{agr}}_{\pi,i}$ and $\mathsf{BadMatch}$ occurs. Otherwise it returns 0.

Advantage:

$$\mathsf{Adv}^{\mathsf{stage-agr}}_{\Pi}(\mathcal{A}) = \Pr[\mathsf{Exp}^{\mathsf{sa}}_{\Pi,\mathcal{A}}(1^\lambda) = 1].$$

Figure 8: Stage-agreement experiment. The adversary wins by declaring an accepting fresh stage for which no unique matching peer exists.

The final-state check is deliberate. It lets the experiment distinguish corruptions that occur before the authenticated material is fixed from late corruptions that should not erase an already established agreement claim.

8.3 Authentication-to-Agreement Reduction

Lemma 8.3 (Authentication-to-agreement)**.** *Let (π, i) be the first declared honest accepting stage for which $E \models \Phi^{\mathsf{agr}}_{\pi,i}$ and no unique matching peer exists. If the session-identifier collision event and the binder-injectivity failure event do not occur, then some authentication branch in every agreement witness set $W^{\mathsf{auth}}_{\pi,i,E}$ is broken in its local authentication game.*

Proof. Fix the first such pair (π, i). Since the collision events are excluded, $B^{\mathsf{agr}}_{\pi,i}$ has a unique transcript representation and a unique intended dual representation. Since $E \models \Phi^{\mathsf{agr}}_{\pi,i}$ and the formula is agreement complete, there exists a witness set $W^{\mathsf{auth}}_{\pi,i,E}$ of authentication atoms whose local capabilities were not exposed before the stage accepted.

The accepting stage verifies authenticated material over $B^{\mathsf{agr}}_{\pi,i}$. If an honest dual peer generated that material for the same vector, then the definitions of dual role, swapped identities, equal suite, equal branch-public vectors, equal transcript hash, and equal stage imply $\mathsf{Match}((\pi, i), (\pi', i)) = 1$. This contradicts the winning condition unless there are two distinct such peers, which is excluded by session-identifier and binder injectivity. Hence no honest dual peer generated valid authenticated material for $B^{\mathsf{agr}}_{\pi,i}$.

Therefore the adversary caused an honest party to accept authentication material for $B^{\mathsf{agr}}_{\pi,i}$ without obtaining the required pre-acceptance capability for at least one authentication atom in the witness set. If the witness formula is conjunctive, all listed branches are required and the failure of any accepting proof gives a local forgery or impersonation. If the witness formula is disjunctive or threshold based, agreement completeness supplies a satisfying witness set; the reduction guesses one branch in this set and embeds the local authentication challenge in the authenticated material for $B^{\mathsf{agr}}_{\pi,i}$. All other branches are simulated honestly. A successful unmatched acceptance outside the excluded collision events yields a valid local authentication forgery, impersonation, or unauthorized binder production for the guessed branch. □

8.4 Main Agreement Theorem

Theorem 8.4 (Stage-agreement preservation)**.** *Let Π be a staged hybrid AKE protocol with agreement-complete formulas, canonical agreement vectors, and a stage-injective transcript binder. Assume each authentication branch $A \in \mathcal{A}$ satisfies its local authentication notion under the corruption relation used by $\mathsf{AuthFr}(A, \cdot, \cdot)$. Let q_{sess} be the number of sessions, q_{st} the maximum number of stages per session, and q_{auth} the maximum size of an agreement witness set. For every adversary $\mathcal{A}_d$,*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{stage-agr}}_{\Pi}(\mathcal{A}_d) \le\ & q_{\mathsf{sess}} q_{\mathsf{st}} \epsilon_{\mathsf{sid}} + q_{\mathsf{sess}} q_{\mathsf{st}} \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}(\mathcal{B}_B) \\ & + q_{\mathsf{sess}} q_{\mathsf{st}} q_{\mathsf{auth}} \max_{A \in \mathcal{A}} \mathsf{Adv}^{\mathsf{auth}}_{A}(\mathcal{B}_A).\end{aligned}$$

Equivalently, conditioned on a guessed winning pair (π^*, i^*) and a guessed branch in a witness set $W^{\mathsf{auth}}_{\pi^*,i^*,E}$, the loss is the corresponding local authentication advantage plus the session-identifier and binder bad-event probabilities.

Proof. The proof guesses the declared accepting stage, removes session and binder collisions, and then applies Lemma 8.3 to reduce any unmatched accepting stage to one selected local authentication-binding break. Appendix B gives the complete game sequence. □

Corollary 8.5 (Downgrade and branch-substitution exclusion)**.** *If $\Phi^{\mathsf{agr}}_{\pi,i}$ is satisfied and $B^{\mathsf{agr}}_{\pi,i}$ includes the suite identifier, stage label, all active confidentiality-branch public values, and all active authentication-branch public values, then any accepting stage that differs from its peer on one of these fields violates either binder injectivity or a local authentication assumption.*

Proof. A disagreement on any listed field makes the two agreement vectors distinct. If the same authenticated material is accepted for both, binder injectivity is violated. If only one side accepts, Theorem 8.4 reduces the unmatched acceptance to a local authentication break. □

9 Separations and Applications: Main-Text Summary

The preservation theorem is useful only if its structural predicates correspond to concrete failure modes. We isolate the main separations in the body and give the complete formal counterexamples in Appendix C. A flat freshness predicate cannot express hybrid survival: after exposing one branch of an OR-hybrid stage, the stage may remain fresh because another branch still dominates the KDF path. Local hybrid-KEM security is also insufficient, since a staged protocol may reveal a KDF ancestor or sibling key after the combiner output has been formed. Ordinary matching is insufficient for hybrid agreement unless the binder is injective over role, peer identity, suite, stage,

branch public values, authentication mode, and transcript hash. Finally, late corruption requires stage-indexed semantics: a corruption after stage i may preserve claims for i while invalidating future stages.

These separations explain the structural conditions used by the theorems: branch-formula freshness, KDF target cuts, binder injectivity, and stage-indexed corruption. The conditions are not cosmetic. If one of them is omitted, the examples show a staged hybrid execution in which a simpler abstraction either rejects a secure surviving stage or accepts a stage whose secrecy or agreement claim is false.

9.1 Main Application: Conditional Scoped TLS 1.3 ECDHE–ML-KEM 1-RTT Preservation

The scoped TLS application asks a concrete preservation question: after an adversary reveals one hybrid branch, selected KDF vertices, selected stage keys, or late party state, which TLS-derived values still remain pseudorandom and which accepted stages still agree on the same peer, role, group, branch public values, and transcript? We consider the certificate-authenticated TLS 1.3 1-RTT path with an ECDHE–ML-KEM hybrid key share, excluding PSK, 0-RTT, HelloRetryRequest, and client authentication. The targets are the handshake traffic secret K_{hs}, application traffic secret K_{app}, exporter master secret K_{exp}, and resumption master secret K_{res}.

For a negotiated hybrid group g, write $Z_g = \mathsf{HybSS}_g(Z_{\mathsf{dh}}, Z_{\mathsf{kem}})$ for the selected-group encoding of the ECDHE and ML-KEM shared secrets. This notation fixes the component order, component lengths, selected-group interpretation, and validation surface prescribed by the ECDHE–ML-KEM group; the proof does not rely on an ambiguous concatenation convention. The KDF graph is the scoped RFC 8446 HKDF schedule, and the binder vector contains the role, identities, offered and selected group context, ECDHE shares, ML-KEM public value and ciphertext, certificate chain, CertificateVerify input, Finished transcript, transcript hash, and stage label. Appendix D gives the byte-level notation and oracle-level expansion.

For each TLS target stage i we use

$$\Phi_i^{\mathsf{sec}} = (\mathsf{Unexp}(C_{\mathsf{dh}}, i) \vee \mathsf{Unexp}(C_{\mathsf{kem}}, i)) \wedge \mathsf{KSafe}(i) \wedge \mathsf{Inj}_{\mathsf{sec}}(i) \wedge \mathsf{NotRev}(K_i),$$

and

$$\Phi_i^{\mathsf{agr}} = \mathsf{AuthFr}(A_{\mathsf{sig}}, i) \wedge \mathsf{AuthFr}(A_{\mathsf{fin}}, i) \wedge \mathsf{Inj}_{\mathsf{agr}}(i).$$

Thus an ECDHE-only reveal can still leave a secrecy claim witnessed by the ML-KEM branch, and an ML-KEM-only reveal can still leave it witnessed by ECDHE; revealing both branches, the target key, or an unrefreshed KDF ancestor destroys the corresponding secrecy claim. There is no single TLS injectivity predicate in these formulas: $\mathsf{Inj}_{\mathsf{sec}}$ is the secrecy-side key-identity condition, while $\mathsf{Inj}_{\mathsf{agr}}$ is the agreement-side authentication-binding condition. Omitting a field that determines the target key, selected group, KDF path, branch input, transcript hash used as KDF context, or target identity falsifies $\mathsf{Inj}_{\mathsf{sec}}$ and hence the secrecy formula. Omitting a field that only affects peer/authentication interpretation falsifies $\mathsf{Inj}_{\mathsf{agr}}$ and hence the agreement formula.

Table 2 records the local TLS obligations used by the scoped theorem. The table is intentionally explicit about protected values and invalidating events: a preservation claim may invoke an obligation only when the final exposure state does not trigger the corresponding abort condition. Table 3 maps the scoped RFC 8446 schedule to graph nodes, and Table 5 lists representative exposure outcomes.

The branch-replacement obligations are the $\mathsf{ECDHE\text{-}Rep}_g$ and $\mathsf{MLKEM\text{-}Rep}_g$ entries of Table 2. Each replaces only the selected component of HybSS_g and is void if the corresponding target branch secret, invalid branch input, or target key is exposed. These are local obligations for the scoped TLS embedding, not first-principles proofs of the selected ECDHE group or ML-KEM parameter set; Appendix D gives the oracle-level interpretation.

Figure 12 gives the local CertificateVerify binding game used for the signature-authentication atom A_{sig}. Figure 9 gives the local Finished authentication-binding game used for the TLS agreement atom A_{fin}. It makes explicit that Finished binds the transcript hash under a key derived from the handshake traffic secret.

Lemma 9.1 (TLS selected-branch replacement)**.** *For the scoped TLS 1.3 ECDHE–ML-KEM handshake with selected group g, let $Z_g = \mathsf{HybSS}_g(Z_{\mathsf{dh}}, Z_{\mathsf{kem}})$ be the selected-group hybrid-secret input. Fix a branch witness $b \in \{\mathsf{dh}, \mathsf{kem}\}$ whose freshness atom is true in the final exposure state. Replacing only the b-component of Z_g by an independent string of the same length, at its selected-group component position, is bounded by*

$$\mathsf{Adv}^{\mathsf{TLS\text{-}br}}_{g,b} \le \mathsf{Adv}^{\mathsf{Rep}}_{g,b} + \epsilon_{\mathsf{enc}}, \qquad \mathsf{Adv}^{\mathsf{Rep}}_{g,b} = \begin{cases} \mathsf{Adv}^{\mathsf{ECDHE\text{-}Rep}}_{g}, & b = \mathsf{dh}, \\ \mathsf{Adv}^{\mathsf{MLKEM\text{-}Rep}}_{g}, & b = \mathsf{kem}. \end{cases}$$

Here ϵ_{enc} is the probability that the selected group, branch-public values, component lengths, ciphertext, or the encoding HybSS_g is malformed or non-injective. The lemma stops at the HybSS_g input; all HKDF propagation from Z_g to a TLS target is charged in Lemma 9.2.

Table 2: Local obligations for the scoped TLS 1.3 ECDHE–ML-KEM instantiation; Appendix D gives the full public inputs and oracle-level variants.

Obligation	Protected value	Invalidating condition	Charged term
$\mathsf{ECDHE\text{-}Rep}_g$	target ECDHE input Z_{dh}	reveal of the target scalar or Z_{dh}; invalid share	$\mathsf{Adv}_g^{\mathsf{ECDHE\text{-}Rep}}$
$\mathsf{MLKEM\text{-}Rep}_g$	target ML-KEM input Z_{kem}	reveal of the decapsulation key or Z_{kem}; malformed ciphertext	$\mathsf{Adv}_g^{\mathsf{MLKEM\text{-}Rep}}$
Hybrid-input Extract	first Extract into v_{hs}	no surviving ideal branch; non-injective hybrid encoding; reserved Extract collision	$\Delta^{\mathsf{hyb}}_{E,g,b}$
TLS-HKDF	post-v_{hs} path to the target	reveal of the cut, target, or unrefreshed ancestor; reserved label/context collision	$N_E\Delta_E + N_X\Delta_X + \epsilon^{\mathsf{TLS}}_{\mathsf{lab}}$
CertVerifyBind	CertificateVerify on the canonical binder	signing capability exposed before acceptance; signature on a different binder	$\mathsf{Adv}^{\mathsf{CertVerify}}_{\mathsf{sig}}$
FinishedBind	Finished under K_{hs}	reveal of K_{hs}, the Finished key, or the target Finished value	$\mathsf{Adv}^{\mathsf{FinishedBind}}_{\mathsf{fin}}$
TLS-Bind	canonical TLS binder vector	session-ID or transcript collision; omission of a key-defining field	$\mathsf{Adv}^{\mathsf{TLS\text{-}bind}}_{\mathsf{Bind}} + \epsilon^{\mathsf{TLS}}_{\mathsf{sid}}$

Game: Finished Authentication Binding $\mathsf{Exp}^{\mathsf{fin-bind}}_{\Pi,\mathcal{A}}(1^\lambda)$

Public data: a scoped TLS transcript prefix with transcript hash TH, a stage label, and a role.

Experiment:

1. The challenger runs the scoped TLS handshake until the handshake traffic secret K_{hs} is defined, records the canonical binder vector $B = (\mathsf{role}, \mathsf{sid}, g, \mathsf{TH}, \mathsf{stage})$, and computes
$$k_{\mathsf{fin}} = \mathsf{Expand}(\mathsf{finished}, \emptyset, K_{\mathsf{hs}}, L), \qquad F = \mathsf{HMAC}(k_{\mathsf{fin}}, \mathsf{TH}).$$
2. The adversary receives the public transcript and may issue allowed reveal or corrupt queries. A query revealing K_{hs}, k_{fin}, or the target Finished value before the claimed acceptance aborts the local game.
3. The adversary outputs an accepting Finished value F' for a vector B'.
4. The experiment outputs 1 iff F' verifies for B', no matching honest generation event for the same $(K_{\mathsf{hs}}, \mathsf{TH}, \mathsf{role}, \mathsf{stage})$ was recorded, and the exposure policy did not authorize production of F'.

Figure 9: Finished authentication-binding game for the scoped TLS instantiation. The accepted Finished value is bound to both the handshake traffic secret and the transcript hash; revealing either the handshake traffic secret or the derived Finished key invalidates the local claim.

Proof. The transcript fixes the selected hybrid group g and the ordered component encoding of HybSS_g. If $b = \mathsf{dh}$, the reduction embeds the $\mathsf{ECDHE\text{-}Rep}_g$ challenge as the ECDHE component and simulates the ML-KEM component honestly. If $b = \mathsf{kem}$, it embeds the $\mathsf{MLKEM\text{-}Rep}_g$ challenge as the ML-KEM component and simulates the ECDHE component honestly. The final freshness atom for b rules out the branch secret reveal that would abort the selected local game. Any ambiguity in the selected component position, component length, group interpretation, or ciphertext validation is charged to ϵ_{enc}. No HKDF call is replaced in this lemma. □

Lemma: Hybrid-Input Extract Compatibility

Fix a selected group g and a branch witness $b \in \{\mathsf{dh}, \mathsf{kem}\}$. After Lemma 9.1, the value

$$\mathsf{HybSS}_g(Z_{\mathsf{dh}}, Z_{\mathsf{kem}})$$

is not assumed to be uniform as a whole: it is a structured selected-group encoding containing one independent selected component and one honestly simulated, possibly known, component. The structured-input Extract obligation, with advantage $\Delta^{\mathsf{hyb}}_{E,g,b}$, states that the reserved call

$$v_{\mathsf{hs}} = \mathsf{Extract}(v_{d0}, \mathsf{HybSS}_g(Z_{\mathsf{dh}}, Z_{\mathsf{kem}}))$$

can nevertheless be replaced by an independent handshake secret when the selected component is independent of the adversary's view, the selected-group encoding is injective and domain separated by g, and the reserved Extract input is not revealed or queried. Equivalently, one ideal component inside the injective hybrid encoding suffices for the first TLS Extract step; the other component may be honestly generated or adversarially known subject to the exposure relation. This is the only transition that converts a single ideal hybrid component into an ideal TLS handshake secret.

Table 3: Scoped RFC 8446 key-schedule mapping. Labels use the RFC 8446 symbolic names; contexts are transcript hashes or empty context; the last column gives the paper's KDF node.

TLS value	label	context	input	node
early_secret	–	–	$(0, 0)$	v_{early}
derived0	derived	$\emptyset$	v_{early}	v_{d0}
handshake_secret	–	–	(v_{d0}, Z_g)	v_{hs}
c/s_hs_traffic	c/s hs traffic	$\mathsf{TH}_{\mathsf{SH}}$	v_{hs}	K_{hs}
derived1	derived	$\emptyset$	v_{hs}	v_{d1}
master_secret	–	–	$(v_{d1}, 0)$	v_{ms}
c/s_ap_traffic	c/s ap traffic	$\mathsf{TH}_{\mathsf{fin}}$	v_{ms}	K_{app}
exporter_master	exp master	$\mathsf{TH}_{\mathsf{fin}}$	v_{ms}	K_{exp}
resumption_master	res master	$\mathsf{TH}_{\mathsf{fin}}$	v_{ms}	K_{res}

Lemma 9.2 (TLS labelled-HKDF paths)**.** *For the scoped TLS 1.3 1-RTT schedule in Table 3, assume the Hybrid-Input Extract Compatibility lemma above, HKDF-Extract satisfies Figure 4, and HKDF-Expand-Label satisfies Figure 5 for the RFC 8446 label encoding and transcript-hash contexts. Then the schedule is a stage-labelled KDF DAG for $K \in \{K_{\mathsf{hs}}, K_{\mathsf{app}}, K_{\mathsf{exp}}, K_{\mathsf{res}}\}$. For each target, Table 4 gives the post-handshake-secret reserved path and graph-local counts $N_E(K)$ and $N_X(K)$ such that, after Lemma 9.1 idealizes one selected branch component,*

$$\mathsf{Adv}^{\mathsf{TLS\text{-}HKDF}}_K \le \Delta^{\mathsf{hyb}}_{E,g,b} + N_E(K)\Delta_E + N_X(K)\Delta_X + \epsilon^{\mathsf{TLS}}_{\mathsf{lab}}.$$

The term $\Delta^{\mathsf{hyb}}_{E,g,b}$ charges the Hybrid-Input Extract Compatibility step that turns one ideal component inside HybSS_g into the ideal v_{hs}. The counts $N_E(K), N_X(K)$ charge only the remaining reserved Extract/Expand calls from v_{hs} to the target.

Proof. The source-to-target paths are those listed in Table 4. The first transition is not treated as if HybSS_g were itself uniform: the Hybrid-Input Extract Compatibility lemma converts the injective encoding containing one ideal component into an ideal v_{hs}. From v_{hs} onward, the RFC 8446 label string and transcript-hash context form the reserved label-context pair for each Expand node, while the master-secret Extract is reserved by its graph position and input context. The remaining calls are replaced in topological order as in Lemma 5.3. A reveal of a listed cut, target, or unrefreshed ancestor falsifies KSafe or NotRev instead of being simulated. Label or context reuse outside the RFC 8446 reserved pairs is charged to $\epsilon^{\mathsf{TLS}}_{\mathsf{lab}}$. □

Table 4: Scoped TLS labelled-HKDF paths after the structured extraction into v_{hs}. Counts exclude the first Hybrid-Secret Extract, charged separately as $\Delta^{\mathsf{hyb}}_{E,g,b}$. The labels are the RFC 8446 labels listed in Table 3.

Target	Reserved path from Z_g	$N_E(K)$	$N_X(K)$	Invalidating ancestor reveals
K_{hs}	$v_{\mathsf{hs}} \to K_{\mathsf{hs}}$	0	1	v_{hs} or target
K_{app}	$v_{\mathsf{hs}} \to v_{d1} \to v_{\mathsf{ms}} \to K_{\mathsf{app}}$	1	2	v_{hs}, v_{ms}, or target
K_{exp}	$v_{\mathsf{hs}} \to v_{d1} \to v_{\mathsf{ms}} \to K_{\mathsf{exp}}$	1	2	v_{hs}, v_{ms}, or target
K_{res}	$v_{\mathsf{hs}} \to v_{d1} \to v_{\mathsf{ms}} \to K_{\mathsf{res}}$	1	2	v_{hs}, v_{ms}, or target

Table 5: Target-specific exposure outcomes for the scoped TLS 1.3 ECDHE–ML-KEM path. "Yes" means that the secrecy claim for that target remains eligible; "No" means that the secrecy formula is false. "No*" means no claim for the revealed target key. Secrecy/key-defining omissions falsify $\mathsf{Inj}_{\mathsf{sec}}$; agreement-only omissions falsify $\mathsf{Inj}_{\mathsf{agr}}$ while leaving secrecy eligible because they do not alter the target key or KDF path.

Exposure	K_{hs}	K_{app}	K_{exp}	K_{res}	Reason
none	Yes	Yes	Yes	Yes	both witnesses may survive
ECDHE branch only	Yes	Yes	Yes	Yes	ML-KEM witness
ML-KEM branch only	Yes	Yes	Yes	Yes	ECDHE witness
both branches	No	No	No	No	no branch witness for Z_g
v_{hs} revealed	No	No	No	No	revealed ancestor/cut
v_{ms} revealed	–	No	No	No	master-secret ancestor
handshake traffic key	No*	Yes	Yes	Yes	sibling for later targets
target key	No*	No*	No*	No*	NotRev false
secrecy/key-defining field omitted	No	No	No	No	$\mathsf{Inj}_{\mathsf{sec}}$ false
agreement-only field omitted	Yes	Yes	Yes	Yes	$\mathsf{Inj}_{\mathsf{agr}}$ false; secrecy unaffected

Corollary 9.3 (Concrete scoped TLS preservation)**.** *In the scoped TLS 1.3 ECDHE–ML-KEM 1-RTT handshake, assume the local obligations of Table 2. Let $K \in \{K_{\mathsf{hs}}, K_{\mathsf{app}}, K_{\mathsf{exp}}, K_{\mathsf{res}}\}$ be a target whose secrecy formula is satisfied in the final exposure state. Let $b \in \{\mathsf{dh}, \mathsf{kem}\}$ be the branch selected by the fixed TLS secrecy selector. With $N_E(K), N_X(K)$ taken from Table 4,*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{tls\text{-}sec}}_{K} \le\ & q_{\mathsf{sess}} q_{\mathsf{st}} g_{\mathsf{sel}} \big(\mathsf{Adv}^{\mathsf{Rep}}_{g,b} + \Delta^{\mathsf{hyb}}_{E,g,b} + N_E(K)\Delta_E + N_X(K)\Delta_X + \epsilon^{\mathsf{TLS}}_{\mathsf{lab}} + \epsilon_{\mathsf{enc}}\big) \\ & + q_{\mathsf{sess}} q_{\mathsf{st}} \big(\mathsf{Adv}^{\mathsf{TLS\text{-}bind}}_{\mathsf{Bind}} + \epsilon^{\mathsf{TLS}}_{\mathsf{sid}}\big).\end{aligned}$$

For every accepting scoped TLS stage whose agreement formula is satisfied,

$$\mathsf{Adv}^{\mathsf{tls\text{-}agr}}_{i} \le q_{\mathsf{sess}} q_{\mathsf{st}} \big(\mathsf{Adv}^{\mathsf{CertVerify}}_{\mathsf{sig}} + \mathsf{Adv}^{\mathsf{FinishedBind}}_{\mathsf{fin}} + \mathsf{Adv}^{\mathsf{TLS\text{-}bind}}_{\mathsf{Bind}} + \epsilon^{\mathsf{TLS}}_{\mathsf{sid}}\big).$$

Consequently, every target marked eligible by Table 5 is stage-secret under the displayed secrecy bound, and every accepting stage satisfying Φ^{agr}_i has stage agreement under the displayed agreement bound.

Proof. The fixed TLS selector chooses a surviving branch b whenever the OR branch formula is true; if both branches are exposed, Table 5 makes the secrecy formula false. Lemma 9.1 replaces only the selected component of HybSS_g, paying $\mathsf{Adv}^{\mathsf{Rep}}_{g,b} + \epsilon_{\mathsf{enc}}$. The Hybrid-Input Extract Compatibility lemma then turns the injective hybrid encoding with one ideal component into an ideal v_{hs}, paying $\Delta^{\mathsf{hyb}}_{E,g,b}$. Lemma 9.2 charges the remaining RFC 8446 labelled-HKDF path from v_{hs} to K with the counts $N_E(K), N_X(K)$. Theorem 7.2 contributes the selector, transcript-binding, and session-collision factors and rejects exactly the ancestor or target reveals shown in Table 5. For agreement, CertificateVerify and Finished give the two local authentication witnesses, while TLS-Bind fixes the selected group, branch public values, roles, identities, transcript hash, and stage label. Theorem 8.4 therefore gives the second displayed bound. □

KEM-authenticated handshakes are expressible by the same formula syntax, but invoking the agreement theorem for them requires a separate local authentication-binding proof for the chosen KEM-authentication mechanism.

10 Selector-Local Reduction Accounting

The preservation theorems are conservative when they sum over all branches in the protocol syntax. A fixed stage normally uses only a small witness. Selector-local accounting charges only that witness, provided the selector rule is fixed before the reduction; otherwise the proof guesses the selector image and pays the stated loss.

10.1 Witnesses and Selectors

Definition 10.1 (Secrecy witness)**.** For (π, i) with $E \models \Phi_{\pi,i}$, a secrecy witness is $W^{\mathsf{sec}} = (W_C, R, \rho, W_T)$, where W_C is a set of protected confidentiality branches, R is a dominating refresh cut for $K_{\pi,i}$, $\rho : R \to W_C$ selects the branch-replacement game for each cut vertex, and W_T is the transcript-binding certificate for the selected branch public values, KDF labels, suite, role, and stage.

Definition 10.2 (Admissible selector)**.** An admissible secrecy selector is a deterministic rule fixed before the reduction, valid whenever $E \models \Phi_{\pi,i}$, and returning

$$\mathsf{Sel}^{\mathsf{sec}}_{\pi,i}(E) \in \mathsf{Wit}^{\mathsf{sec}}(\Phi_{\pi,i}, E).$$

An agreement selector returns (W_A, W_T), where W_A satisfies the authentication formula and W_T is a binding certificate. A selector image is *online-determined at time* τ if the target pair, public transcript, and exposure history up to τ uniquely determine it for every final fresh extension. Otherwise the reduction guesses the image in advance.

Accounting Interface: Selector-Local Costs

For $W^{\mathsf{sec}} = (W_C, R, \rho, W_T)$ and $W^{\mathsf{agr}} = (W_A, W_T)$,

$$\mathsf{Cost}^{\mathsf{sec}}_{\pi,i,E}(W^{\mathsf{sec}}) = \sum_{r \in R} \mathsf{Adv}^{\mathsf{br-rep}}_{\rho(r)} + \mathsf{Adv}^{\mathsf{kdf-hide}}_{G_\pi, K_{\pi,i}, R} + \mathsf{Adv}^{\mathsf{bind}}_{W_T},$$

$$\mathsf{Cost}^{\mathsf{agr}}_{\pi,i,E}(W^{\mathsf{agr}}) = \sum_{A \in W_A} \mathsf{Adv}^{\mathsf{auth}}_{A} + \mathsf{Adv}^{\mathsf{bind}}_{W_T}.$$

Figure 10: Selector-local accounting interface. Costs suppress adversary, time, and query parameters and charge only the branches, KDF cut, and binder certificate selected by a fixed admissible selector.

10.2 Selector-Local Secrecy and Agreement

Theorem 10.3 (Selector-local stage secrecy)**.** *Let* Π *satisfy Theorem 7.2. For fixed* (π^*, i^*) *and admissible* $\mathsf{Sel}^{\mathsf{sec}}_{\pi^*,i^*}$*, every final* $E \models \Phi_{\pi^*,i^*}$ *satisfies*

$$\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi,\pi^*,i^*}(\mathcal{A}) \le \epsilon_{\mathsf{sid}} + \mathsf{Cost}^{\mathsf{sec}}_{\pi^*,i^*,E}(\mathsf{Sel}^{\mathsf{sec}}_{\pi^*,i^*}(E)),$$

up to the tested-pair or selector-image guessing factor when the reduction must guess it.

Proof. Run the proof of Theorem 7.2 with the selector fixed before challenge embedding. Conditioned on the selected image, only $\rho(r)$, the cut-to-target ancestry, and W_T are charged; Lemmas 5.3 and 5.4 make the unrevealed target ideal. □

Theorem 10.4 (Selector-local stage agreement)**.** *Let* Π *satisfy Theorem 8.4. For fixed accepting* (π, i) *and admissible* $\mathsf{Sel}^{\mathsf{agr}}_{\pi,i}$*, every final* $E \models \Phi^{\mathsf{agr}}_{\pi,i}$ *gives unmatched-acceptance probability at most*

$$\epsilon_{\mathsf{sid}} + \mathsf{Cost}^{\mathsf{agr}}_{\pi,i,E}(\mathsf{Sel}^{\mathsf{agr}}_{\pi,i}(E)),$$

plus the usual winning-pair guessing or summation loss.

Proof. The selector fixes the authentication witnesses. Outside session-identifier and binder failures, an unmatched acceptance yields accepted material for the selected agreement vector without a matching generation event, hence one selected local authentication-binding break. □

Non-uniform minimum accounting. For OR or threshold formulas, the quantity

$$\min_{W \in \mathsf{Wit}^{\mathsf{sec}}(\Phi_{\pi,i}, E)} \mathsf{Cost}^{\mathsf{sec}}_{\pi,i,E}(W)$$

may be reported only as a non-uniform summary: each fixed witness has its own reduction, while a uniform reduction must use a predeclared selector or pay the selector-image loss. This refinement is not full tight AKE security.

11 Scope and Limitations

The results apply to staged hybrid AKE protocols with explicit labelled KDF DAGs and branch-formula freshness. They cover concurrent Reveal, branch reveal, stage reveal, selective corruption, and late corruption when the instantiation supplies the required branch assumptions, refresh cuts, and binder-injectivity certificates. The framework is not a full tight AKE theorem; it makes no claim when all dominating branches are exposed, no refresh cut exists, the target is revealed, or the binder omits a peer/key-defining field. Cyclic derivation graphs, implicit KDF state, symbolic soundness, implementation leakage, and side channels are outside the interface.

12 Conclusion

We presented branch-formula freshness for staged hybrid AKE. The preservation theorems reduce stage secrecy and agreement to branch replacement, authentication security, labelled KDF target hiding, and binding injectivity; selector-local accounting charges only selected witnesses. The scoped TLS 1.3 ECDHE–ML-KEM instantiation makes the concrete hybrid-handshake proof obligations explicit.

AI Usage. We used ChatGPT only for text polishing. AI was not used to fully generate any part of the paper, nor was it used to generate any contribution.

References

[1] Dowling, B., Fischlin, M., Günther, F., Stebila, D.: A cryptographic analysis of the TLS 1.3 handshake protocol. Journal of Cryptology **34**, Article 37, 1–69 (2021). doi:10.1007/s00145-021-09384-1

[2] Dowling, B., Fischlin, M., Günther, F., Stebila, D.: A cryptographic analysis of the TLS 1.3 handshake protocol candidates. In: Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security (CCS 2015), pp. 1197–1210. ACM (2015). doi:10.1145/2810103.2813653

[3] Bindel, N., Brendel, J., Fischlin, M., Gonçalves, B.A., Stebila, D.: Hybrid key encapsulation mechanisms and authenticated key exchange. In: Ding, J., Steinwandt, R. (eds.) PQCrypto 2019. LNCS, vol. 11505, pp. 206–226. Springer (2019). doi:10.1007/978-3-030-25510-7_12

[4] Giacon, F., Heuer, F., Poettering, B.: KEM combiners. In: Abdalla, M., Dahab, R. (eds.) PKC 2018, Part I. LNCS, vol. 10769, pp. 190–218. Springer (2018). doi:10.1007/978-3-319-76578-5_7

[5] Bruckner, S.P., Ramacher, S., Striecks, C.: Muckle+: End-to-end hybrid authenticated key exchanges. In: Johansson, T., Smith-Tone, D. (eds.) PQCrypto 2023. LNCS, vol. 14154, pp. 601–633. Springer (2023). doi:10.1007/978-3-031-40003-2_22

[6] Battarbee, C., Striecks, C., Perret, L., Ramacher, S., Verhaeghe, K.: Quantum-safe hybrid key exchanges with KEM-based authentication. EPJ Quantum Technology **12**, Article 128 (2025). doi:10.1140/epjqt/s40507-025-00425-3

[7] Schwabe, P., Stebila, D., Wiggers, T.: Post-quantum TLS without handshake signatures. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS 2020), pp. 1461–1480. ACM (2020). doi:10.1145/3372297.3423350

[8] Wiggers, T., Celi, S., Schwabe, P., Stebila, D., Sullivan, N.: KEM-based authentication for TLS 1.3. IETF Internet-Draft draft-celi-wiggers-tls-authkem-07 (2026)

[9] Delpech de Saint Guilhem, C., Fischlin, M., Warinschi, B.: Authentication in key-exchange: Definitions, relations and composition. In: 2020 IEEE 33rd Computer Security Foundations Symposium (CSF), pp. 288–303. IEEE (2020). doi:10.1109/CSF49147.2020.00028

[10] Bader, C., Hofheinz, D., Jager, T., Kiltz, E., Li, Y.: Tightly-secure authenticated key exchange. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015, Part I. LNCS, vol. 9014, pp. 629–658. Springer (2015). doi:10.1007/978-3-662-46494-6_26

[11] Jager, T., Kiltz, E., Riepel, D., Schäge, S.: Tightly-secure authenticated key exchange, revisited. In: Canteaut, A., Standaert, F.X. (eds.) EUROCRYPT 2021, Part I. LNCS, vol. 12696, pp. 117–146. Springer (2021). doi:10.1007/978-3-030-77870-5_5

[12] Kwiatkowski, K., Kampanakis, P., Westerbaan, B.E., Stebila, D.: Post-quantum hybrid ECDHE–MLKEM key agreement for TLSv1.3. IETF Internet-Draft draft-ietf-tls-ecdhe-mlkem-04 (2026)

[13] Barbosa, M., Connolly, D., Duarte, J.D., Kaiser, A., Schwabe, P., Varner, K., Westerbaan, B.E.: X-Wing: The hybrid KEM you've been looking for. IACR Cryptology ePrint Archive, Report 2024/039 (2024)

[14] National Institute of Standards and Technology: Module-Lattice-Based Key-Encapsulation Mechanism Standard. FIPS 203 (2024). doi:10.6028/NIST.FIPS.203

[15] Krawczyk, H., Eronen, P.: HMAC-based Extract-and-Expand Key Derivation Function (HKDF). RFC 5869 (2010). doi:10.17487/RFC5869

[16] Rescorla, E.: The Transport Layer Security (TLS) Protocol Version 1.3. RFC 8446 (2018). doi:10.17487/RFC8446

A Proof Interface and Reduction Accounting

The preceding sections give the calculus, its preservation theorems, the selector-local refinement, and two representative instantiations. This section records the proof interface in a form that is directly checkable by a protocol proof. The purpose is not to introduce a new assumption. The purpose is to make the reductions use the same accounting convention and to state precisely where each hypothesis is consumed.

A.1 Canonical Advantage Terms

For a session-stage pair (π, i) and final exposure state E, write

$$\mathsf{SecFresh}(\pi, i, E) \equiv E \models \Phi_{\pi,i} \qquad \text{and} \qquad \mathsf{AgrFresh}(\pi, i, E) \equiv E \models \Phi^{\mathsf{agr}}_{\pi,i}.$$

The conservative preservation bounds are expressed using the following global terms; Section 10 refines these terms by applying a fixed admissible selector for the witnesses of a fixed tested stage.

$$\mathsf{Adv}^{\mathsf{cut}}_{\mathcal{C}}(q_{\mathsf{cut}}) = \sum_{r \in R} \mathsf{Adv}^{\mathsf{br-rep}}_{C(r)}, \qquad \mathsf{Adv}^{\mathsf{auth}}_{\mathcal{A}}(q_{\mathsf{auth}}) = \sum_{A \in W^{\mathsf{auth}}} \mathsf{Adv}^{\mathsf{auth}}_{A}.$$

The set R is a dominating refresh cut for the tested stage, and W^{auth} is a witness set for the agreement formula. The remaining terms are

$$\mathsf{Adv}^{\mathsf{lab-prf}}_{\mathsf{Extract},\mathsf{Expand}}(q_{\mathsf{cut}}), \qquad \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}, \qquad \epsilon_{\mathsf{sid}} = \Pr[\mathsf{Coll}_{\mathsf{sid}}].$$

Thus the global secrecy theorem can be read in the normalized form

$$\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi}(\mathcal{A}) \le q_{\mathsf{sess}} q_{\mathsf{st}} \left(\mathsf{Adv}^{\mathsf{cut}}_{\mathcal{C}}(q_{\mathsf{cut}}) + \mathsf{Adv}^{\mathsf{lab-prf}}_{\mathsf{Extract},\mathsf{Expand}}(q_{\mathsf{cut}}) + \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}} + \epsilon_{\mathsf{sid}} \right),$$

and the global agreement theorem in the normalized form

$$\mathsf{Adv}^{\mathsf{stage-agr}}_{\Pi}(\mathcal{A}_d) \le q_{\mathsf{sess}} q_{\mathsf{st}} q_{\mathsf{auth}} \left(\mathsf{Adv}^{\mathsf{auth}}_{\mathcal{A}}(q_{\mathsf{auth}}) + \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}} + \epsilon_{\mathsf{sid}} \right).$$

The notation hides no additional security condition: each term is discharged by one of the obligations summarized in the technical roadmap of Section 1.2. The selector-local refinement replaces the displayed global cut and authentication terms by the witnesses returned by the fixed admissible selectors for a fixed tested or accepting stage.

The roadmap should be read as the reduction dependency graph. KDF target hiding is used only on paths from exposed ancestors to a tested key, and refresh separation turns protected cuts into target hiding. Stage secrecy additionally needs a branch witness for every cut vertex. Stage agreement does not use the KDF lemmas; it uses authentication witnesses and injective transcript binding. Selector-local accounting does not add a new reduction step; it removes unused branches from the loss by applying a predeclared witness selector. Both preservation theorems share the same session-identifier and transcript-representation bad events, and the scoped TLS corollary instantiates the dependency path.

A.2 Proof Obligations

A protocol instantiation invokes the preservation theorems by providing five objects.

1. A branch vector containing every confidentiality and authentication branch that can affect a claimed stage.
2. A labelled KDF DAG whose vertices include all branch injections, intermediate secrets, and derived stage keys.
3. A binder input for each agreement claim, including identities, roles, suite identifiers, stage labels, branch public values, authentication public values, and the transcript hash.
4. A secrecy formula $\Phi_{\pi,i}$ whose satisfying assignments determine a dominating refresh cut for $K_{\pi,i}$.
5. An agreement formula $\Phi^{\mathsf{agr}}_{\pi,i}$ whose satisfying assignments determine a nonempty authentication witness set and binder injectivity for stage i.

These obligations are closed under substitution of branch assumptions. Replacing an ECDH branch by a KEM branch changes only the local branch advantage and the public fields included in the binder. Replacing signature authentication by KEM authentication changes only the local authentication game and the corresponding witness atom. The preservation theorems do not change.

A.3 Separating Example for Formula Freshness

The formula condition is a precondition, not a proof artifact. If $E \not\models \Phi_{\pi,i}$, the secrecy theorem makes no claim. The following example shows why the condition is part of the theorem statement. For a two-branch hybrid stage with

$$\Phi^{\mathsf{conf}}_{\pi,i} = \mathsf{Unexp}(C_{\mathsf{dh}}, \pi, i) \vee \mathsf{Unexp}(C_{\mathsf{kem}}, \pi, i),$$

exposing both branch secrets gives

$$\neg\mathsf{Unexp}(C_{\mathsf{dh}}, \pi, i) \wedge \neg\mathsf{Unexp}(C_{\mathsf{kem}}, \pi, i),$$

and the adversary may compute the combiner input and every descendant key not otherwise refreshed. Similarly, if the agreement binder omits the suite or stage label, then $\mathsf{Inj}_{\mathsf{agr}}(i)$ is false and the agreement theorem makes no claim even if all cryptographic branches remain unbroken. The calculus is therefore falsifiable: failed structural obligations remove exactly the affected claim.

A.4 Terminology Convention

The paper uses stage for a protocol output point, branch for a confidentiality or authentication source, binder for the authenticated transcript representation used in agreement claims, late corruption for corruption after the relevant stage acceptance, and KDF node for a vertex of the labelled derivation graph. These terms are not interchangeable. A branch may inject a value into several KDF nodes; a stage may output several application-specific keys; a binder may contain a transcript hash but is not identical to it. The distinction is part of the theorem interface.

B Deferred Proofs for Main Preservation Theorems

B.1 Full proof of Theorem 1

Proof. Fix an adversary $\mathcal{A}$. We prove the conditioned bound for a guessed tested pair (π^*, i^*); multiplying by $q_{\mathsf{sess}}q_{\mathsf{st}}$ gives the first displayed statement. The challenger aborts if the adversary tests a different pair. Conditioned on no abort, the simulation is identical to the original experiment up to the explicit bad events below. The reduction maintains four invariants in every game. First, all public transcripts, branch public values, identities, roles, suites, and session identifiers outside the guessed challenge are sampled exactly as in the real protocol. Second, every Reveal or Corrupt answer is either produced from honestly simulated state, produced from the current hybrid value at a non-target vertex, or makes the final test formula false and therefore causes the experiment to return 0 at the final freshness check. Third, the selected witness image is fixed before a branch or KDF challenge is embedded; if this image cannot be determined online, the reduction guesses it and pays the factor g_{sel}. Fourth, no oracle answer exposes a reserved KDF label-context pair or the protected cut value in a fresh execution; such an answer is treated as a reveal that falsifies KSafe or NotRev.

Game G_0. This is $\mathsf{Exp}^{\mathsf{ss}}_{\Pi,\mathcal{A}}(0)$ with a real test key.

Game G_1. Abort if two sessions have the same session identifier while different accepted transcript data determine the same stage, or if the transcript encoding collides. The transition changes the adversary's success probability by at most ϵ_{sid}. Outside this event, every accepted tested stage has a unique public transcript tuple for the purpose of key derivation.

Game G_2. Abort if the binder is not stage-injective for the tested stage: there exist accepted pairs $(\pi, i) \neq (\pi', i')$ whose binding inputs agree while their intended peer, role, suite, branch-public vector, stage label, or transcript hash differ. A reduction $\mathcal{B}_B$ forwards all public binding inputs to the binder-injectivity game and outputs a violation when G_2 aborts. Hence

$$|\Pr[G_1 = 1] - \Pr[G_2 = 1]| \leq \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}(\mathcal{B}_B).$$

Outside G_2's abort event, the branch secrets and KDF labels used by the tested stage are determined by a single transcript and cannot be substituted from another stage or suite.

Game G_3. The admissible selector rule is fixed before the reduction starts. If its image is not determined online before the relevant branch challenges are embedded, the reduction guesses one admissible selector image in advance and aborts unless the final exposure state maps to that image; this is the factor g_{sel}. Non-fresh executions already return 0. On a fresh execution, the fixed rule yields

$$\mathsf{Sel}^{\mathsf{sec}}_{\pi^*, i^*}(E) = (W, R, \rho, W_T),$$

with no retroactive choice of a cheaper witness. For each $r \in R$ in the selected order, replace the protected branch predecessor $z_{C(r),r}$ by the branch-replacement ideal value, while leaving the rest of the session simulation unchanged.

For a fixed r, the two adjacent games differ only in the value injected at the predecessor selected by the fixed map $\rho(r)$. All exposure events recorded in E satisfy the local unexposed atom for that branch. Therefore a branch-replacement reduction $\mathcal{B}_r$ embeds its challenge as $z_{C(r),r}$, simulates all other protocol values honestly, and uses $\mathcal{A}$'s distinguishing advantage between the adjacent games to distinguish the real branch injection from its replacement value. The reduction is not post-hoc: the selector rule and, if necessary, the selector-image guess are fixed before the embedded challenge is chosen. Thus

$$|\Pr[G_2 = 1] - \Pr[G_3 = 1]| \leq \sum_{r \in R} \mathsf{Adv}^{\mathsf{br-rep}}_{C(r),\rho(r)}(\mathcal{B}_r).$$

Game G_4. Replace the target key K_{π^*,i^*} by the ideal target of the KDF target-hiding game for the selected cut R. Since R is a dominating refresh cut, every path from any revealed strict ancestor of K_{π^*,i^*} to the target crosses a vertex in R. Since the cut values are ideal in G_3, Lemma 5.3 applies. A reduction $\mathcal{B}_K$ gives the adversary the same revealed vertices, public labels, and non-reserved KDF oracle access as in the real experiment, reserves the graph labels on paths from R to the target, and uses $\mathcal{A}$ to distinguish the real target from the ideal target. Hence

$$|\Pr[G_3 = 1] - \Pr[G_4 = 1]| \leq \mathsf{Adv}^{\mathsf{kdf-hide}}_{G_\pi, K_{\pi^*,i^*}, R}(\mathcal{B}_K).$$

Game G_5. Return an independent uniform string as the test key. In G_4, the target key is already ideal and independent of the bit hidden from the adversary. Therefore G_4 and G_5 are identical. In G_5, the adversary's output is independent of whether the challenger labels the returned uniform string as real or random, and its distinguishing advantage is 0.

Combining the transitions by the triangle inequality gives the conditioned bound. The unconditioned game is obtained by guessing one of at most $q_{\mathsf{sess}} q_{\mathsf{st}}$ session-stage pairs before the execution and aborting if the adversary tests a different pair. This yields the stated factor. The theorem follows. □

B.2 Full proof of Theorem 2

Proof. Fix an adversary $\mathcal{A}_d$. The proof first conditions on a guessed declared pair (π^*, i^*) and then removes the conditioning by a factor $q_{\mathsf{sess}} q_{\mathsf{st}}$. Conditioned on this guess, the challenger aborts if the adversary declares any other pair.

Game G_0. This is the real stage-agreement experiment. The adversary wins when the declared honest accepting stage satisfies $E \models \Phi^{\mathsf{agr}}_{\pi^*,i^*}$ and has no unique matching peer.

Game G_1. Abort if two distinct sessions have the same session identifier while their agreement vectors differ, or if the transcript representation used to compute B^{agr} collides. The change in winning probability is at most ϵ_{sid}. Outside this event, a session identifier names at most one accepted agreement vector for a fixed stage and role pair.

Game G_2. Abort if the transcript binder is not injective on accepted agreement vectors: two distinct accepted contexts determine the same authenticated binder input although they differ in role, peer identity, suite, stage, branch-public vector, authentication-public vector, or transcript hash. A reduction $\mathcal{B}_B$ records all accepted agreement vectors and outputs the first injectivity violation. Hence

$$|\Pr[G_1 = 1] - \Pr[G_2 = 1]| \leq \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}}(\mathcal{B}_B).$$

Outside G_2's abort event, replaying authenticated material across a different suite, branch vector, role, or stage is impossible unless the local authentication branch itself is broken.

Game G_3. If the adversary wins, agreement completeness gives a witness set $W^{\mathsf{auth}}_{\pi^*,i^*,E}$. Guess one branch $A^* \in W^{\mathsf{auth}}_{\pi^*,i^*,E}$. The reduction $\mathcal{B}_{A^*}$ embeds its local authentication challenge into the protocol messages that define the authenticated material for $B^{\mathsf{agr}}_{\pi^*,i^*}$, simulates all non-selected authentication branches honestly, and answers corruption and reveal queries according to the exposure relation of $\mathsf{AuthFr}(A^*, \pi^*, i^*)$. Since the final formula is satisfied, the forbidden pre-acceptance capability for A^* is never queried. By Lemma 8.3, a win outside the bad events gives a successful local authentication forgery, impersonation, or unauthorized authenticated decapsulation. Therefore

$$\Pr[G_2 = 1] \le q_{\mathsf{auth}} \cdot \max_{A \in \mathcal{A}} \mathsf{Adv}^{\mathsf{auth}}_A(\mathcal{B}_A).$$

Game G_4. Replace the winning condition by false outside the bad events and outside a local authentication break. In this game an honest accepting stage with a satisfied agreement formula has a unique matching peer by the definition of Match and by binder injectivity. Hence $\Pr[G_4 = 1] = 0$.

Combining the transitions gives the conditioned bound. Guessing the declared session-stage pair before the execution gives the displayed unconditioned bound. The theorem follows. □

B.3 Detailed proofs of Corollaries 7.3 and 7.4

Detailed proof of Corollary 7.3. Fix a tested pair (π, i) and a final exposure state E such that the complete freshness formula for the stage is satisfied. In particular,

$$E \models \mathsf{Unexp}(C_0, \pi, i) \vee \mathsf{Unexp}(C_1, \pi, i).$$

Let b be the first index in a fixed public order on $\{0, 1\}$ for which $E \models \mathsf{Unexp}(C_b, \pi, i)$. This rule is fixed before the reduction and is therefore an admissible selector; if a proof implementation cannot determine b online before embedding the challenge, the selector-image loss is the two-element guessing factor already accounted for by Theorem 7.2.

Let u_b be the KDF-graph injection vertex receiving the contribution of branch C_b. By the hypothesis of the corollary, every directed path to $K_{\pi,i}$ passes through the branch-injection structure in a way that makes u_b a dominating refresh point, or places u_b in a dominating refresh cut R_b for the target. More explicitly, set R_b to the smallest cut selected by the protocol's secrecy-completeness witness that contains the first descendant of the C_b injection on every path to $K_{\pi,i}$. Since $E \models \mathsf{Unexp}(C_b, \pi, i)$, no allowed exposure query reveals the protected predecessor used for u_b; since the complete formula is satisfied, $KSafe(\pi, i)$ and $NotRev(K_{\pi,i})$ also hold. Thus (W, R, ρ) with $W = \{C_b\}$, $R = R_b$, and $\rho(r) = C_b$ for the cut vertex or vertices refreshed by the selected branch is a valid secrecy witness.

The hybrid proof is then exactly the fixed-pair proof of Theorem 7.2. First remove session-identifier and transcript-binding bad events. Next embed the branch-replacement challenge for C_b at the selected injection point, paying only $\mathsf{Adv}^{\mathsf{br-rep}}_{C_b}$ for the surviving branch witness. After that replacement, the selected cut is ideal. Lemma 5.3 hides the target through the reserved KDF path from R_b to $K_{\pi,i}$, paying the corresponding KDF target-hiding term and label-collision probability. The final game returns a uniform target key. Summing the transitions gives the advertised loss: the selected surviving branch advantage, the KDF target-hiding advantage for the selected cut, and the binding/collision terms from Theorem 7.2. No exposed non-surviving branch is charged because the selector never embeds a challenge in it. □

Detailed proof of Corollary 7.4. Fix a tested pair (π, i) and a final exposure state E satisfying the complete freshness formula. Let

$$S_E = \{j \in [n] : E \models \mathsf{Unexp}(C_j, \pi, i)\}.$$

The threshold formula gives $|S_E| \ge t$. Fix once and for all a deterministic public tie-breaking rule, for example lexicographic order on branch identifiers, and let $T_E \subseteq S_E$ be the first t indices according to this rule for which the corollary's hypothesis supplies a dominating refresh witness. If this selected image is not online-determined before the reduction embeds a challenge, the proof guesses it and pays the admissible selector-image factor from Theorem 7.2; otherwise the factor is one.

By hypothesis, the selected t surviving branch atoms determine a dominating refresh cut R_T whose protected predecessors are exactly the branch contributions $\{C_j : j \in T_E\}$. Define the witness set $W_T = \{C_j : j \in T_E\}$ and map each cut vertex $r \in R_T$ to the branch $\rho(r) \in W_T$ whose contribution refreshes that vertex. Because

every selected branch atom is true in E, the corresponding protected predecessor is not exposed under the branch-replacement relation. Because the full stage formula is satisfied, the target is not revealed and every exposed strict ancestor is separated from the target by R_T.

Apply the fixed-pair proof of Theorem 7.2 to this witness. After removing session and binder collisions, replace the selected branch contributions one at a time. The jth adjacent hybrid is bounded by the local branch-replacement advantage for the corresponding selected branch, and no non-selected branch is charged. Once all selected cut predecessors are ideal, Lemma 5.3 hides the target using the reserved KDF subgraph from R_T to $K_{\pi,i}$. Therefore

$$\mathsf{Adv}^{\mathsf{stage-sec}}_{\Pi,\pi,i}(\mathcal{A}) \le \epsilon_{\mathsf{sid}} + \mathsf{Adv}^{\mathsf{bind}}_{\mathsf{Bind}} + \sum_{j \in T_E} \mathsf{Adv}^{\mathsf{br-rep}}_{C_j} + \mathsf{Adv}^{\mathsf{kdf-hide}}_{G,K_{\pi,i},R_T},$$

with the usual session-stage and selector-image factors in the unconditioned game. This is precisely the claimed threshold loss: the sum of the t selected branch advantages, the KDF target-hiding term, and the binding/collision terms. □

C Separations from Simpler Freshness Notions

This section records limitations of three simpler proof interfaces. The separations are deliberately structural: each one preserves a familiar local property while invalidating a staged branch-formula guarantee. They justify the three nonstandard ingredients used above, namely exposure formulas, KDF refresh cuts, and agreement-vector injectivity.

C.1 Flat Freshness Cannot Express Hybrid Survival

A flat freshness predicate classifies a session-stage pair as fresh or exposed. Such a predicate is adequate when a single uncompromised secret is necessary for the tested key. It is not adequate for hybrid stages whose intended confidentiality policy is disjunctive or threshold based.

Separation A: flat freshness versus branch formulas

Consider a one-stage protocol with two confidentiality branches C_0 and C_1 and a combiner

$$S = \mathsf{Comb}(Z_0, Z_1, \mathsf{TH}), \qquad K = \mathsf{KDF}(S, \ell, \mathsf{TH}).$$

The intended policy is

$$\Phi^{\mathsf{conf}} = \mathsf{Unexp}(C_0) \vee \mathsf{Unexp}(C_1).$$

Let the adversary issue $\mathsf{RevealBranch}(C_0)$ and no query exposing C_1 or the KDF ancestors after the injection of Z_1.

1. A flat predicate that rejects after any branch reveal declares the stage non-fresh although the hybrid policy is still satisfied.
2. A flat predicate that accepts after some branch reveals cannot distinguish the above exposure from the exposure of both C_0 and C_1.

Thus no single Boolean session predicate depending only on whether a reveal occurred captures the intended OR-hybrid guarantee.

Proposition C.1 (Flat freshness does not capture branch survival)**.** *For staged hybrid protocols that allow OR or threshold confidentiality policies, a preservation theorem that uses only a flat session freshness predicate either rejects some branch-survival executions or accepts executions in which all required branch witnesses are exposed.*

Proof. The first item in Separation A gives the over-rejection case. If the flat predicate is changed to accept after the exposure of one branch, then the predicate value is unchanged when the adversary additionally exposes the remaining branch unless the predicate records branch identities and the policy relating them. In the latter case it is no longer a flat predicate but a formula over exposure atoms. Therefore a flat predicate cannot soundly and completely represent OR or threshold branch survival. □

The proposition does not claim that flat freshness is unsuitable for all AKE models. It says that it is not a reusable interface for hybrid migration policies, where different stages may intentionally use different Boolean combinations of branch assumptions.

C.2 Local Hybrid-KEM Security Does Not Imply Staged KDF Safety

A hybrid KEM theorem is a local statement about one shared secret. A staged protocol uses that secret inside a KDF graph and exposes selected descendants, siblings, or ancestors. The missing condition is not another

KEM assumption, but a graph condition: every path from an exposed ancestor to the tested stage must cross an unrevealed refresh injection or a pseudorandom KDF edge protected by an unrevealed node.

Separation B: local hybrid security without KDF-stage separation

Let $S = \mathsf{Comb}(Z_0, Z_1, \mathsf{TH})$ be a secure hybrid shared secret whenever $\mathsf{Unexp}(C_0) \vee \mathsf{Unexp}(C_1)$ holds. Consider the derivation graph

$$S \xrightarrow{\ell_0} K_0, \qquad S \xrightarrow{\ell_1} K_1,$$

where the same undomain-separated KDF call or the same output block is reused for both labels. Suppose the adversary reveals K_0 and tests K_1. The local hybrid-KEM theorem still states that S is hidden under a surviving branch, but it says nothing about whether K_0 reveals K_1. If the labels collide or the outputs are reused, K_1 is determined by the reveal of K_0.

Proposition C.2 (Local hybrid security does not capture KDF reveals)**.** *Hybrid shared-secret security alone does not imply stage-key secrecy under stage-reveal queries. The separating execution is ruled out by domain-separated KDF independence for the relevant edge or by a dominating refresh cut between every exposed ancestor and the tested stage.*

Proof. In Separation B the combined secret S may be computationally hidden from the adversary before stage-key reveals, so the local hybrid-KEM guarantee is not contradicted. Nevertheless, the tested key K_1 is exposed through the revealed key K_0 when the derivation labels collide or the output block is reused. Thus the implication from local hybrid security of S to secrecy of all descendants is false without an additional KDF-stage condition. The KDF target-hiding lemma supplies exactly the missing condition by requiring a protected path, expressed as domain-separated derivation or a dominating refresh cut. □

This separation explains why the theorem interface contains $\mathsf{KSafe}(i)$ in addition to branch atoms. Branch survival supplies entropy. KDF safety determines whether that entropy still protects the tested stage after the adversary's stage-reveal pattern.

C.3 Ordinary Matching Does Not Imply Hybrid Agreement

Classical matching predicates are often defined through a session identifier or through equality of a transcript prefix. Hybrid staged protocols need a stronger object: agreement must cover the fields that determine both the key and the authentication claim. Omitting a field from the binder can make two sessions appear matched while disagreeing on the security-relevant branch structure.

Separation C: matching without full binder injectivity

Consider two accepting sessions with equal session identifier and equal transcript hash value as seen by the authentication branch. The first negotiates a hybrid suite $\mathsf{suite} = \mathsf{cl}\|\mathsf{pq}$ and uses branch vector (X, ct). The second negotiates a classical-only suite $\mathsf{suite}' = \mathsf{cl}$ and uses branch vector (X). If the authenticated binder excludes suite and the branch vector, the same authentication value can be accepted in both executions. A matching predicate based only on sid or on the reduced transcript therefore declares agreement, although the parties disagree on the active branch policy and on the public values that should determine the KDF input.

Proposition C.3 (Ordinary matching does not capture hybrid agreement)**.** *A staged hybrid agreement theorem cannot be derived from ordinary matching alone. The separating execution is ruled out by injectivity of the authenticated agreement vector over identities, roles, suite, stage, active branch public values, authentication public values, and transcript hash.*

Proof. Separation C gives two accepted executions that agree on the reduced matching information but disagree on fields that alter the security policy and the key derivation input. If the same authentication value is valid in both executions, then no local authentication assumption is necessarily broken: the value was produced for the reduced binder input that both executions share. The failure is instead a binder failure. Requiring injectivity of B^{agr} over the full field set makes the two executions distinct for authentication; accepting the same material across them then violates binder injectivity or yields a local authentication break, as in Theorem 8.4. □

C.4 Late Corruption Requires Stage-Indexed Semantics

Late corruption is another place where flat notions lose information. A post-accept corruption may be irrelevant to an already completed stage with forward secrecy, while invalidating future stages that depend on long-term state, resumption state, or unrekeyed authentication material.

Separation D: late corruption and stage index

Let a protocol derive K_i from an ephemeral hybrid secret and later derive K_j from a resumption secret stored after stage i, with $i < j$. If the adversary corrupts the party after acceptance of stage i but before the derivation of stage j, then a sound model may keep

$$\mathsf{LateCorrupt}_{>i} \not\Rightarrow \neg\Phi_i$$

while requiring

$$\mathsf{LateCorrupt}_{>i} \Rightarrow \neg\Phi_j$$

when the corruption exposes the resumption state used by stage j. A freshness predicate that is not indexed by stage cannot express this distinction.

Proposition C.4 (Session-level corruption does not capture late compromise)**.** *For protocols with late corruptions and multiple derivation stages, freshness must be evaluated relative to the target stage. A single corruption flag for the whole session either discards valid past-stage claims or preserves invalid future-stage claims.*

Proof. In Separation D the same corruption event has different consequences for K_i and K_j. If the model marks the whole session non-fresh after late corruption, it rejects the valid past-stage claim. If it preserves the whole session as fresh because stage i had already accepted, it incorrectly preserves the future claim for K_j. Hence the exposure relation must be evaluated at the stage whose secrecy or agreement is claimed. □

C.5 Consequence

Proposition C.5 (Separating examples for the structural conditions)**.** *The preceding examples show four independent failure modes for staged hybrid AKE with branch-specific Reveal queries, stage-key Reveal queries, selective corruption, and late corruption: flat freshness misses branch survival, local hybrid shared-secret security misses KDF-stage reveals, ordinary matching misses hybrid downgrade fields, and session-level corruption misses stage-indexed compromise.*

Proof. Proposition 1 gives the branch-survival separation for OR and threshold hybrid policies. Proposition 2 gives the KDF-stage reveal separation. Proposition 3 gives the agreement-vector separation for branch and downgrade fields. Proposition 4 gives the late-corruption separation. Together they justify the structural conditions used in the preservation theorems, without claiming a general impossibility theorem beyond these separating examples. □

D TLS 1.3 ECDHE–ML-KEM 1-RTT Instantiation

This appendix instantiates the calculus for the certificate-authenticated TLS 1.3 1-RTT handshake with an ECDHE–ML-KEM hybrid key share. The scope is deliberately fixed: no PSK, no 0-RTT, no HelloRetryRequest, and no client authentication. The purpose is to discharge the preservation interface for the current hybrid key-agreement shape rather than to reprove every mode of TLS 1.3.

D.1 Protocol Skeleton and Key Schedule

The client sends a ClientHello containing the offered hybrid group, the ECDHE share X, and the ML-KEM encapsulation public material required by the selected draft group. The server responds with ServerHello containing the selected group, ECDHE share Y, and the ML-KEM ciphertext ct. The branch contributions are

$$Z_{\mathsf{dh}} = \mathsf{ECDH}(x, Y) = \mathsf{ECDH}(y, X), \qquad Z_{\mathsf{kem}} = \mathsf{MLKEM.Dec}(sk, ct).$$

For the selected group g, the hybrid shared secret is $Z_g = \mathsf{HybSS}_g(Z_{\mathsf{dh}}, Z_{\mathsf{kem}})$, the selected-group encoding required by the ECDHE–ML-KEM group [12]. This encoded value feeds the TLS 1.3 HKDF key schedule [16, 15]. In the no-PSK mode,

$$\begin{aligned}
\mathsf{early_secret} &= \mathsf{Extract}(0, 0),\\
\mathsf{derived0} &= \mathsf{Expand}(\mathsf{early_secret}, \mathsf{derived}, \emptyset),\\
\mathsf{handshake_secret} &= \mathsf{Extract}(\mathsf{derived0}, Z_g),\\
c/s_\mathsf{hs_traffic} &= \mathsf{Expand}(\mathsf{handshake_secret}, c/s\ \mathsf{hs\ traffic}, \mathsf{TH}_{\mathsf{SH}}),\\
\mathsf{derived1} &= \mathsf{Expand}(\mathsf{handshake_secret}, \mathsf{derived}, \emptyset),\\
\mathsf{master_secret} &= \mathsf{Extract}(\mathsf{derived1}, 0),\\
c/s_\mathsf{ap_traffic} &= \mathsf{Expand}(\mathsf{master_secret}, c/s\ \mathsf{ap\ traffic}, \mathsf{TH}_{\mathsf{fin}}),\\
\mathsf{exporter_master} &= \mathsf{Expand}(\mathsf{master_secret}, \mathsf{exp\ master}, \mathsf{TH}_{\mathsf{fin}}),\\
\mathsf{resumption_master} &= \mathsf{Expand}(\mathsf{master_secret}, \mathsf{res\ master}, \mathsf{TH}_{\mathsf{fin}}).
\end{aligned}$$

The notation abstracts byte-level TLS 1.3 labelled-HKDF encodings but preserves the RFC labels, inputs, contexts, and graph dependencies shown in Table 3.

Figure 11 summarizes the scoped TLS instance used by the preservation corollary.

Construction: TLS 1.3 ECDHE–ML-KEM 1-RTT Instance

Branches. C_{dh} is the ECDHE branch; C_{kem} is the ML-KEM branch. A_{sig} is the certificate-signature branch used by CertificateVerify, and Finished is modeled as a MAC-style transcript-confirmation binder under the handshake traffic secret.

Binder vector.

$$B_i = (\mathsf{role}, ID_c, ID_s, \mathsf{supported_groups}, \mathsf{selected_group}, i,\ X, Y, ct, \mathsf{Certs}, \mathsf{CertificateVerify}, \mathsf{Finished}, \mathsf{TH}_i).$$

Secrecy formulas. For handshake and application traffic stages,

$$\Phi_i^{\mathsf{sec}} = (\mathsf{Unexp}(C_{\mathsf{dh}}, i) \vee \mathsf{Unexp}(C_{\mathsf{kem}}, i)) \wedge \mathsf{KSafe}(i) \wedge \mathsf{Inj}_{\mathsf{sec}}(i) \wedge \mathsf{NotRev}(K_i).$$

Exporter and resumption master secrets use the same formula with the target vertex changed to the corresponding HKDF output.

Agreement formula.

$$\Phi_i^{\mathsf{agr}} = \mathsf{AuthFr}(A_{\mathsf{sig}}, i) \wedge \mathsf{AuthBind}(\mathsf{Finished}, B_i) \wedge \mathsf{Inj}_{\mathsf{agr}}(i) \wedge \mathsf{LateOK}(i).$$

Figure 11: Scoped TLS 1.3 ECDHE–ML-KEM 1-RTT instance. The construction fixes the transcript fields and HKDF schedule used by the preservation corollary.

D.2 Discharging the Obligations

The scoped TLS proof uses the same audit form for every local obligation: public data, protected value, allowed reveals, abort events, replacement or forgery target, and charged advantage. Table 2 gives the main-text summary; this appendix spells out the interpretation. Public data always includes the selected group g, the public transcript, identities, certificates, transcript hashes, and all branch public values. A local game is invoked only if the final exposure state does not reveal the protected value named by that game and does not violate the validation predicate for the selected group. Otherwise the corresponding freshness atom is false and the preservation theorem makes no claim for that target.

Branch replacement. The ECDHE branch is discharged by $\mathsf{ECDHE\text{-}Rep}_g$ for the selected group and the HybSS_g component position. The ML-KEM branch is discharged by $\mathsf{MLKEM\text{-}Rep}_g$ under the selected ML-KEM parameter set and the draft's decapsulation validation rules. Lemma 9.1 charges only the selected surviving branch. The selected-group encoding is part of the labelled KDF input and binder vector, so branch substitution changes B_i and is covered by $\mathsf{Inj}_{\mathsf{sec}}(i)$ and $\mathsf{Inj}_{\mathsf{agr}}(i)$ as appropriate. Operationally, the reduction first fixes the target stage and the selector's branch witness b. It then embeds the corresponding branch challenge in exactly one component of HybSS_g and simulates the other component honestly. If the adversary reveals the protected scalar, decapsulation key, branch shared secret, or target key, the final formula is false and the TLS preservation claim is not evaluated. Encoding failures, including ambiguous component lengths or selected-group interpretation, are charged to ϵ_{enc}.

The TLS authentication obligations are interpreted in the same final-state style. CertVerifyBind protects the signature capability for the canonical transcript input used by CertificateVerify; an exposure of the relevant signing capability before the accepting authentication event falsifies the corresponding authentication atom. FinishedBind protects the Finished MAC value under the derived Finished key; a reveal of the handshake traffic secret, the derived Finished key, or the target Finished value before the claimed acceptance aborts the local game. TLS-Bind is the injectivity obligation for the binder vector containing the selected group, ECDHE share, ML-KEM public value and ciphertext, roles, identities, stage label, certificate-authentication material, Finished transcript, and transcript hash.

KDF cuts. The first reserved HKDF call is the extraction of handshake_secret from the structured hybrid shared secret; this cost is charged by the structured Hybrid-Secret Extract term $\Delta^{\mathsf{hyb}}_{E,g,b}$ in Lemma 9.2. The reserved paths and counts are those of Table 4: after v_{hs} is ideal, K_{hs} uses one reserved Expand, while K_{app}, K_{exp}, and K_{res} additionally pass through v_{d1} and master_secret. The reserved label-context set consists of the RFC 8446 labels on these paths together with their transcript-hash contexts. A revealed handshake traffic secret is treated as a sibling

Game: CertificateVerify Binding $\mathsf{Exp}^{\mathsf{cv-bind}}_{\Pi,\mathcal{A}}(1^\lambda)$

Public data: a scoped TLS transcript prefix, server identity, selected group, branch public values, certificate chain, stage label, and transcript hash TH.

Experiment:

1. The challenger records the canonical CertificateVerify vector $B_{\mathsf{cv}} = (\mathsf{role}, ID_c, ID_s, g, X, Y, ct, \mathsf{stage}, \mathsf{TH})$ and gives the adversary the public transcript, verification key, and certificate chain.
2. The adversary may request honest signatures for non-target canonical vectors and may issue allowed reveal or corrupt queries. A query exposing the signing capability for the target identity before the accepting CertificateVerify event aborts the local claim.
3. The adversary outputs a signature σ' and vector B'_{cv} accepted by the verification algorithm.
4. The experiment outputs 1 iff σ' verifies for B'_{cv}, no matching honest signing event for that exact canonical vector was recorded, and the exposure policy did not authorize producing σ'.

Figure 12: CertificateVerify binding game for the scoped TLS instantiation. The accepted signature must be bound to the canonical TLS transcript vector; exposing the target signing capability before acceptance invalidates the local claim.

leaf, not as an ancestor of application traffic secrets; it invalidates only claims whose formula or local authentication atom names that value. If handshake_secret or master_secret is revealed, the descendant targets in Table 4 are not fresh unless another modeled refresh occurs.

Agreement binding. The CertificateVerify signature covers the transcript hash, and Finished confirms the same transcript under the handshake traffic secret. Since B_i includes the selected hybrid group, both branch public values, the stage label, and the transcript hash, a classical-only transcript, a different ML-KEM ciphertext, or a different stage label yields a different binder vector. Thus downgrade and branch-substitution attacks violate $\mathsf{Inj}_{\mathsf{agr}}(i)$ unless they break the transcript representation or authentication branch.

Corollary D.1 (Conditional scoped TLS 1.3 ECDHE–ML-KEM 1-RTT stage secrecy)**.** *In the scoped TLS 1.3 ECDHE–ML-KEM 1-RTT instance above, assume the ECDHE and ML-KEM branches satisfy their branch-replacement games, the TLS labelled-HKDF schedule satisfies the labelled HKDF/PRF interface and target-hiding theorem, and the transcript representation is injective except with the stated collision probability. Then every handshake, application, exporter, or resumption target whose formula Φ_i^{sec} is satisfied is stage-secret with the bound of Theorem 7.2 specialized to the selected ECDHE/ML-KEM branch witness and HKDF cut.*

Proof. A satisfying formula supplies at least one unexposed branch. The branch-replacement game replaces that branch contribution at the hybrid shared-secret input. The labelled-HKDF graph gives a dominating cut from the hybrid secret to the target stage. The KDF target-hiding lemma replaces the selected descendant by an ideal value. Transcript-injectivity and session-collision events account for the remaining bad events. The target non-reveal atom excludes the post-Test reveal attack. □

Corollary D.2 (Conditional scoped TLS 1.3 ECDHE–ML-KEM 1-RTT stage agreement)**.** *In the same scoped TLS instance, every accepting stage satisfying Φ_i^{agr} has a unique matching peer except with the signature-authentication, Finished-binding, transcript-injectivity, and session-collision losses of Theorem 8.4.*

Proof. The accepting transcript contains CertificateVerify and Finished material bound to the transcript hash containing the selected group and both hybrid branch public values. If an honest peer generated the same authenticated binder vector, the matching predicate holds. Otherwise, outside transcript-collision and binder-injectivity failures, the adversary has produced valid authentication material for the vector without the corresponding local authentication capability. Theorem 8.4 gives the claim. □

D.3 KEM-Authentication as an Extension

The formula language can express KEM-authenticated policies such as

$$\mathsf{AuthFr}(A_{\mathsf{sig}}, i) \wedge \mathsf{AuthFr}(A_{\mathsf{akem}}, i) \qquad \text{or} \qquad \mathsf{AuthFr}(A_{\mathsf{sig}}, i) \vee \mathsf{AuthFr}(A_{\mathsf{akem}}, i).$$

This paper does not claim a full KEM-authentication instantiation. To invoke the agreement theorem for KEMTLS-, AuthKEM-, or Muckle#-style designs, one must first supply a concrete local AuthBind game proving that accepted

KEM-authentication material is bound to the canonical vector B_i under the allowed corruption relation. Once such a local game is available, the preservation theorem applies without changing the KDF or secrecy side of the calculus.

D.4 Downgrade and Branch-Substitution Diagnostics

Table 6 summarizes the diagnostic omissions. If the representation used for key derivation omits the selected group, branch public values, ciphertext, transcript hash, or stage label, then $\mathsf{Inj}_{\mathsf{sec}}(i)$ is false and the secrecy theorem makes no claim. If an omitted field affects only authenticated peer interpretation, then $\mathsf{Inj}_{\mathsf{agr}}(i)$ is false and the agreement theorem makes no claim. The diagnostic rows below therefore distinguish key-defining omissions from agreement-only omissions rather than using a single injectivity predicate.

Table 6: Diagnostic binder omissions. The table records only the omitted field and the resulting confusion; the surrounding prose explains why the theorem refuses the agreement claim.

Omitted field	Confusion	Failed atom
Selected group	classical/hybrid	$\mathsf{Inj}_{\mathsf{sec}}(i)$ or $\mathsf{Inj}_{\mathsf{agr}}(i)$
KEM ciphertext	branch substitution	$\mathsf{Inj}_{\mathsf{sec}}(i)$ or $\mathsf{Inj}_{\mathsf{agr}}(i)$
Stage label	handshake/application	$\mathsf{Inj}_{\mathsf{sec}}(i)$ or $\mathsf{Inj}_{\mathsf{agr}}(i)$
Role	client/server	$\mathsf{Inj}_{\mathsf{agr}}(i)$
Finished transcript	stale transcript	$\mathsf{AuthBind}$