QuantumTrustKG: Provenance-Aware Orchestration for Quantum-Safe Microservices

Xuan-Bach Le[1] ✉⋆, Phat T. Tran-Truong[1], Duy Nguyen[2], Ngoc Khanh Tran Nguyen[3], and Xuan Son Ha[4]

[1] Ho Chi Minh City University of Technology (HCMUT), VNU-HCM, Vietnam
{lexuanbach,phatttt}@hcmut.edu.vn
[2] University of Economics Ho Chi Minh City, Vietnam
duynguyen.726202321440@st.ueh.edu.vn
[3] University of Science, VNU-HCM, Vietnam
22120378@student.hcmus.edu.vn
[4] RMIT University, Vietnam
ha.son@rmit.edu.vn

Abstract. Post-quantum migration requires a service mesh to choose cryptographic profiles as endpoint capabilities, policies, threats, and evidence freshness change. Static mesh settings and policy-as-code enforce a posture once it is chosen, but neither makes that choice for each connection. We present *QuantumTrustKG*, a provenance-aware, graph-backed control-plane framework for connection-specific selection. For each communication edge, its Quantum Trust Protocol Orchestration (QTPO) procedure removes profiles that fail a capability, policy, threat, or freshness gate, ranks the rest by cost, and blocks the edge if none is admissible. We mechanize the control-plane guarantees in Coq/Rocq and implement QTPO as a Kubernetes/Istio controller. In a controlled comparison on 200-service meshes, QTPO records no unsafe promotions and 96.5% *declared-label policy compliance*, while service-level and opportunistic baselines promote unsafely on 3.4–13.8% of edges. On sampled production service graphs of up to 2000 services, these baselines reach 37–60% and QTPO stays at zero. The compliance metric reads the same declared labels that drive the controller. It therefore scores control-plane decisions and cannot show which cryptography is negotiated on the wire.

Keywords: service orchestration · microservices · knowledge graphs · service mesh · post-quantum cryptography · crypto-agility

1 Introduction

Organizations are migrating to post-quantum cryptography (PQC) to address store-now-decrypt-later attacks [14,6]. The transition is gradual because endpoints

⋆ X.-B. Le and P. T. Tran-Truong contributed equally to this work.
✉ Corresponding author: Xuan-Bach Le, lexuanbach@hcmut.edu.vn.
Artifact and extended version: https://github.com/lexuanbach/quantumtrustkg.

gain support at different times, policies vary across trust boundaries, and threat information expires. A checkout service may need a quantum-safe profile for its regulated card-processor edge while keeping a hybrid profile for a cache whose peer is still migrating (Fig. 1, stage 4). No single service-level posture fits both.

Service meshes enforce configured profiles, and policy-as-code validates those configurations [3]. Choosing profiles as capabilities, policies, advisories, and rollout state change needs a separate control loop. We provide that loop for every edge through *QuantumTrustKG*, a control-plane framework and Kubernetes/Istio controller. A knowledge graph (KG) joins services, edges, profiles, policies, provenance, and rollout state, and the Quantum Trust Protocol Orchestration (QTPO) procedure filters candidates for admissibility, then ranks survivors by cost.

We make three contributions. First, QTPO assigns profiles per edge. Per-service governance [8], per-endpoint profiling [9], and static inventories [11] cannot represent different obligations on two inbound edges of the same service. Second, admissibility is independent of cost. Quality-of-service-aware selection ranks candidates under a fixed correctness posture [4]. In QTPO the ranker sees only candidates that passed the hard gates, and changing the score weights therefore cannot promote an inadmissible profile (Prop. 1). Third, provenance takes part in admission. Stale, contradictory, or unavailable evidence removes a candidate, and an edge left with none is blocked with the failed gate recorded.

We evaluate control-plane decisions with *declared-label policy compliance* (C_{decl}), computed from the capability claims, policies, and profile classes that the controller consumes. The metric shows whether decisions satisfy these declared inputs but cannot show which cryptography is negotiated on the wire.

2 Background and the Edge-Level Decision Problem

Endpoints adopt the post-quantum algorithms standardized by the US National Institute of Standards and Technology (NIST), among them the Module-Lattice-Based Key-Encapsulation Mechanism (ML-KEM) [6], at different times. *Hybrid* profiles, which pair a classical and a post-quantum key exchange [7], therefore coexist with classical and quantum-safe ones. An Istio `DestinationRule` or `PeerAuthentication` sets the mutual TLS (mTLS) mode on a connection [1], and policy-as-code engines such as the Open Policy Agent (OPA) check configuration against rules. Neither decides *which* profile a connection should get.

Decisions are made per logical service-communication edge when relevant configuration changes. Individual RPCs and TLS connections are not decision units. We model the application as a directed service graph $G = (\mathcal{S}, E)$ in which $e = (s_i, s_j)$ is a call from s_i to s_j. Each edge carries the trust boundary it crosses, the active policy, the current assignment, the rollout status, and the provenance (source and freshness window) of each fact. Because a stale or contradictory fact can silently weaken protection, the controller treats every fact as *evidence* to be checked. Each deployable profile $p \in \mathcal{P}$ carries a *declared* class $\mathrm{Security}(p) \in \{\mathsf{classical}, \mathsf{hybrid}, \mathsf{quantum\_safe}\}$, ordered $\mathsf{classical} \preceq \mathsf{hybrid} \preceq \mathsf{quantum\_safe}$ as operator policy classes. This is not a universal ordering of cryptographic strength.

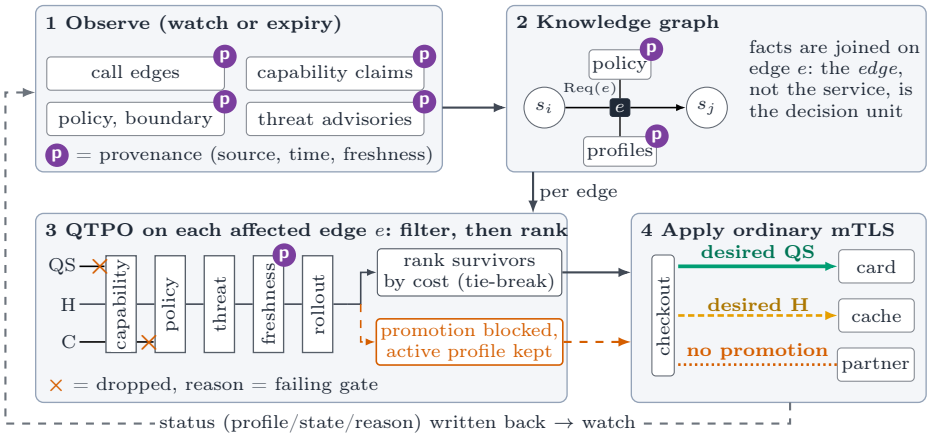

Fig. 1. The QuantumTrustKG control loop. QTPO filters each edge's classical (C), hybrid (H) and quantum-safe (QS) candidates and ranks the survivors by cost. One `checkout` service receives three per-edge outcomes.

We trust Kubernetes, Istio, and the Custom Resource Definition (CRD) store, and tampering by a principal authorized under role-based access control (RBAC) is out of scope. The problem is to compute an assignment $\pi : E \to \mathcal{P} \cup \{\mathsf{blocked}\}$ that prefers lower cost without trading it against validity. An edge is promoted only if its profile clears every gate of Sect. 3 against its required class Req(e). A failed decision sets `promotionBlocked`. It does not itself set `trafficDenied`, and the previous `activeProfile` may remain in use.

3 QuantumTrustKG

On each watch event, QuantumTrustKG runs the loop of Fig. 1. Facts enter with provenance and are joined per logical service-communication edge in the Quantum Trust Graph, QTPO decides each affected edge, the assignments become Istio configuration, and auditable status is written back. The loop is incremental, runs off the data path, and fails closed. Any path that cannot establish admissibility from fresh evidence keeps the previous assignment, never picks a weaker one, and installs no traffic-denial rule.

Graph-backed evidence. At decision time, the graph joins endpoint capabilities, inherited policy, trust-boundary context, provenance, and rollout state on its central object, `:CommunicationEdge`. We hold it as Resource Description Framework (RDF) data in Apache Jena Fuseki and query it with the SPARQL Protocol and RDF Query Language, because the inputs are open and heterogeneous, the decisions are late multi-way joins, and named graphs carry provenance per statement [16]. When several policies match an edge, we collapse the active, non-overridden matches by selector specificity to the strongest required class

Algorithm 1: QTPO: per-edge protocol selection.

Input: Quantum Trust Graph G; edge $e = (s_i, s_j)$; reconciliation time t
Output: assignment $\pi(e) \in \mathcal{P} \cup \{\texttt{blocked}\}$

$R \leftarrow \text{Req}(e)$ `// strongest applicable policy class`
$C_e \leftarrow$ `CompatibleProfiles`(s_i, s_j) `// advertised by both endpoints (fresh records)`
$A \leftarrow \{\, p \in C_e \mid$ `FilterGates`$(p, e, t)\,\}$ `// keep only admissible profiles`
if $A = \emptyset$ ***and*** `PolicyPermitsHybrid`(e) **then**
  $A \leftarrow \{\, h \in C_e \mid$ `FilterGates`(h, e, t) under fallback policy $\}$
if $A = \emptyset$ **then return** blocked `// fail closed; reason = first failed gate`
$p^\star \leftarrow \arg\min_{p \in A} \text{Score}(p, e)$; apply `PeerAuthentication/DestinationRule`
if `MeshReady`$(e, p^\star)$ **then return** $p^\star$ `// both workloads ready: commit as active`
return blocked `// not ready: last committed profile stays; retry`

Req(e), which keeps a permissive rule from weakening a stricter one. Every fact also carries a source, an observation time, a freshness window (300 s by default), and contradiction notes, and an expired or contradicted fact therefore blocks a new promotion. The NoKG/NoProv ablation (Sect. 4) removes both the graph-backed join and its provenance/freshness checks.

Selection. QTPO *filters before it ranks* (Algorithm 1) and never downgrades silently. A candidate p passes FilterGates(p, e, t) exactly when (i) Security$(p) \succeq$ Req(e), (ii) p's family is neither deprecated nor of insufficient proof status, (iii) no active advisory targets it, (iv) every $f \in \text{facts}(p, e)$ is fresh at t and non-contradictory, and (v) p is rollout-feasible. A failing candidate is dropped with its failed gate logged, and only survivors are ranked by cost. An edge with no survivor has promotion blocked and keeps its last committed assignment, unless policy explicitly permits a hybrid fallback that itself clears every gate. The score weighs overhead, residual risk, migration cost, and a switch penalty (w_o=0.25, w_r=0.40, w_m=0.20, w_h=0.15, hysteresis δ=0.05), tuned in Sect. 5.

Guarantees. Let $A(e, t)$ be the *admissible* profiles for e at reconciliation time t. A profile is admissible if both endpoints support it, it passes every hard gate active on e using facts that are fresh and non-contradictory at t, and it is rollout-feasible.

Theorem 1 (Safety: admissibility and class preservation, G1–G2). *For every edge e that QTPO promotes, $\pi(e) \in A(e, t)$ and* Security$(\pi(e)) \succeq$ Req(e)*, unless active policy explicitly permits hybrid fallback and no quantum-safe profile is admissible, in which case $\pi(e)$ is an admissible hybrid profile.*

Theorem 2 (Fail-closed promotion, G4). *If any fact required to decide admissibility for e is stale, contradictory, unavailable, or mesh-inconsistent at t, QTPO performs no new promotion on e. It preserves the last committed assignment and records* `promotionBlocked` *with the failed gate as the reason. This theorem does not imply traffic denial or continuous admissibility of the retained assignment.*

Proposition 1 (Weight-independence). *$A(e, t)$, hence both theorems, is independent of the score weights and δ, which only reorder candidates within $A(e, t)$.*

Both follow *by construction* from the filter order and promotion rule, through a case analysis over the direct and fallback cases of Algorithm 1 (full proofs

in the extended version). We mechanize them in Coq/Rocq 9.1.1 (checked as `G1`/`G2`/`G4`). Before committing a promotion, the controller re-checks it with an independent checker that mirrors the Coq predicate field-for-field and is differentially tested against it. These are *control-plane* guarantees over declared metadata. They say nothing about the bytes on the wire, since a compromised workload could advertise quantum-safe capability, negotiate classical TLS, and leave both theorems intact. We prove no primitive-level cryptographic security.
Implementation. The prototype is a Go controller (`controller-runtime`) that stores the graph in Fuseki and emits Istio objects enforcing ordinary mTLS, with the selected desired profile as metadata. Stock Envoy/Istio does not thereby negotiate that PQ group. Three operator CRDs supply the required class per edge set, the supported profiles with their source and freshness window, and a staged migration plan. The controller-owned `AssignmentStatus` records each edge's desired and active profile, state, and reason. Each promoted edge yields a `PeerAuthentication` and a `DestinationRule` whose `profile` annotation a migration changes, and a profile is recorded as active only once both objects exist and both workloads are ready. The destination rule selects a destination workload/port under Istio's ordinary selector and precedence semantics. The controller does not enforce a native per-source PQ handshake. Reconciliation is scheduled at the next evidence expiry, and a monotonic evidence-cache revision is rechecked before commit. A concurrent update blocks promotion for a fresh retry. This is not a transactional snapshot across Fuseki and the Kubernetes API server. Capability provenance is operational for now. Signed claims are not yet bound to identities from the Secure Production Identity Framework for Everyone and its runtime (`SPIFFE/SPIRE`), and RBAC separation of publishers, policy authors, and the status writer is the sole integrity boundary.

4 Evaluation

We evaluate safety against competing selectors (RQ1), behavior on sampled production service graphs (RQ2), and migration cost (RQ3). Quantities are marked [M] when measured and [P] when modelled. *H1, archived in-cluster replay* runs an earlier full controller, Fuseki, and Istio on 50/100/200-service testbeds (30 repetitions, Wilson intervals). *H2, cluster-free* isolates the selection logic so every baseline runs against an identical edge population. It is the only harness with competing systems, and its meshes include requirement divergence on about 30% of edges and corrupted provenance on 3%. Both use seed `20260509`, and H2 also runs over real Alibaba graphs [10]. The earlier release mishandled disjoint endpoint profiles and readiness-before-commit. The released controller closes both paths but has not been rerun in-cluster, and H1 does not validate its guarantees. H2 invokes the selector on endpoint-compatible candidates and is unaffected.
Metrics. *Declared-label policy compliance* C_{decl} is the share of managed edges on which the controller promotes a profile clearing every gate of Sect. 3, or a policy-permitted hybrid fallback. Because $\mathrm{Req}(e)$ and $\mathrm{Security}(p)$ come from the declared metadata the decision consumes, C_{decl} measures whether the control

plane decides correctly *given its inputs*. The *invalid-promotion rate* is evaluated independently against the full declared-input admissibility predicate: endpoint support, policy class, family/proof status, advisories, fresh noncontradictory evidence, and rollout readiness. We partition observations into valid promotion, invalid promotion, no promotion, and previous active profile retained. Neither metric verifies declared capabilities against the wire.

The two workloads are not directly comparable. At 200 services the archived H1 outcomes divide into 99.1% valid, 0.9% unpromoted, and 0% invalid under the historical predicate. H2 is 96.5%, 3.4%, and 0%, respectively. H2 intentionally contains more divergent and corrupted edges.

4.1 RQ1: Safe Edge-Level Decisions

In the archived replay, $C_{\text{decl}}^{\text{cluster}}$ falls from 99.4% [99.1, 99.6] at 50 services to 99.3% [99.0, 99.5] at 100 and 99.1% [98.9, 99.3] at 200, against 96.9% [96.5, 97.3], 96.6% [96.2, 96.9] and 96.4% [96.0, 96.7] for NoKG/NoProv. Wilson intervals separate them at every scale. The archived run recorded zero invalid promotions under its narrower historical instrumentation, and every residual edge was not promoted with a status reason. These observations do not validate the current controller. Median selection latency grows from 12 ms at 50 services to 45 ms at 200 [M]. Requests never pay it, as QTPO runs only on configuration change. As no prior post-quantum-migration system ships a runnable per-edge control plane, we distil the dominant literature patterns into reproducible selectors (Table 1). Enterprise-Level Crypto-Agility (ELCA) [8] applies per-service policy, and post-quantum TLS configuration profiling (ConfigProfile) [9] keeps a per-endpoint inventory with provenance. We add service-level crypto-agility (SL-CA), opportunistic negotiation (OPP), which trusts claims at face value, Static, a fixed mesh-wide hybrid posture, and NoKG/NoProv. A method's two rates do not sum to 100 because the remaining edges are not promoted. Because the H2 comparison starts without active assignments, the distinct retained-active count is zero, and controller tests check retention separately. The 50- and 100-service meshes agree with Table 1 within 1.2 points. Only QTPO has zero unsafe promotions. Without per-edge granularity, ELCA, SL-CA, and ConfigProfile incur 11.2–13.8% unsafe promotions, because a service whose inbound edges diverge has no single correct posture. Provenance gating lowers ConfigProfile to 11.2%, under ELCA's 13.5%.

Baseline implementation. The baselines are selector-level re-creations: ELCA and ConfigProfile lacked obtainable code, while SL-CA and OPP are patterns. They share edge populations, requirements, repetitions, and seeds but omit other system components. NoKG/NoProv shares QTPO's ranker, and its roughly 3% gap supports the combined graph-and-provenance gate without isolating RDF.

4.2 RQ2: Real Production Service Graphs

RQ2 repeats the comparison on service-dependency graphs sampled from a production trace. From one 3-minute call-graph bucket of the Alibaba trace [10] (16,247 services, 48,746 heavy-tailed edges), we take breadth-first subgraphs

Table 1. Controlled, cluster-free selection comparison on H2 (200-service meshes, 30 reps, seed 20260509) [M]. NKG/NP = NoKG/NoProv, SL-CA = service-level crypto-agility, OPP = opportunistic.

	QTPO	NKG/NP	OPP	ConfigProf.	Static	ELCA	SL-CA
Valid (%)	**96.5**	96.5	96.5	85.7	86.7	86.1	85.7
Invalid (%)	**0.0**	2.5	3.4	11.2	13.1	13.5	13.8
Invalid n	**0**	303	403	1336	1557	1614	1640
No promotion n	403	100	0	304	0	0	0
Retained-active n	0	0	0	0	0	0	0
Pooled observations	11,929 (30 repetitions)						

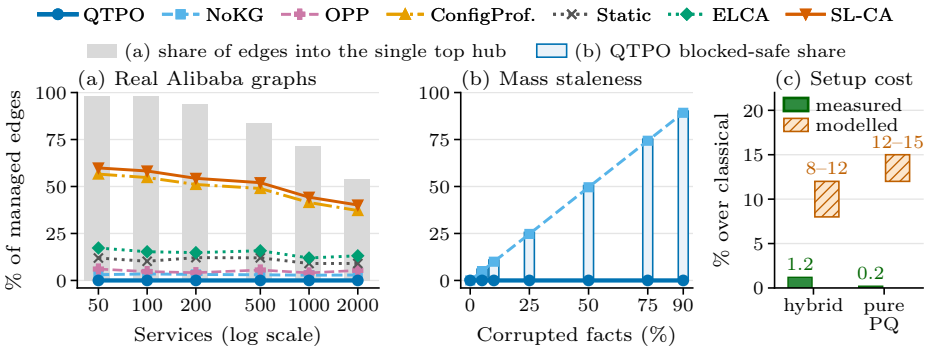

Fig. 2. (a,b) Unsafe promotions as lines, in % of managed edges over 30 reps [M], on (a) real Alibaba subgraphs and (b) a 200-service mesh under mass staleness. (c) Setup cost over classical, measured (solid, [M]) or modelled (hatched, [P]). PQ is post-quantum.

from the top hub at six scales. The trace carries no cryptographic policy. *Only its topology is real*, and the experiment compares two *structures* under one requirement model. QTPO holds 0.00% invalid promotions at all six scales [M] (Fig. 2(a)), from 50 to 3,692 unique edges per repetition (1,500 to 110,760 pooled edge–repetition observations), because a per-edge gate never has to generalize across a service's edges. SL-CA and ConfigProfile, at 11–14% unsafe on synthetic meshes, show three to four times those rates at equal size. Their rates fall monotonically with scale, from 59.87% and 56.60% at 50 services to 40.15% and 37.19% at 2000 [M], while NoKG/NoProv stays near 3%. Meanwhile, the largest hub's edge share falls from 98.0% to 54.1%, against 6.1% in the full trace.

A service-uniform method chooses one destination class compatible with every peer. One legacy peer thus lowers all inbound edges of a hub, including regulated ones. As hub share falls, its rate improves. The sampled subgraphs are more concentrated than the full trace. The band is not a production average.

The provenance gate also makes QTPO fail closed under mass staleness, as Thm. 2 predicts (Fig. 2(b)). As corrupted annotations sweep from 0 to 90% over a 200-service mesh, QTPO has no unsafe promotion at any level [M]. Its blocked-safe share tracks the injected fraction, and C_{decl} falls from 99.03% to 9.87% as QTPO refuses evidence it cannot trust. Provenance-blind NoKG/NoProv instead

promotes every corrupted edge unsafely, reaching 24.75/49.59/89.22% at 25/50/90%. On the real 200-service topology, QTPO stays at zero.

4.3 RQ3: Measured and Modelled Migration Costs

The aggregate setup-cost model combines handshake central-processing-unit (CPU)/wire, key material, rollout, and churn. Only the first is measured (Fig. 2(c), solid). On stock OpenSSL 3.6.2, which exposes ML-KEM natively, the median of 300 TLS 1.3 handshakes takes 8.746 ms with X25519, 8.850 ms with X25519MLKEM768 (+1.2%), and 8.759 ms with ML-KEM-768 (+0.2%) [M], consistent with earlier TLS benchmarks of lattice-based key exchange [15], and 30/30 end-to-end trials per profile succeed [M]. The other three terms are unmeasured model parameters. Our calibrated model puts the aggregate at 8–12% for hybrid and 12–15% for pure post-quantum [P] (hatched). Roughly 7–14 of those 8–15 points are modelled. We claim no measured aggregate migration cost. The measured handshake overhead does not dominate under the specified aggregate-cost model. It does not establish a physical total across unlike quantities. Rollout (controller logs) takes 4–7 min with 0.0%/0.0% failure/rollback for hybrid and 8–12 min with 1.1%/0.6% for pure post-quantum [M]. The extended version gives the model's ±25% envelope.

5 Discussion, Limitations, and Threats to Validity

Cost-weight guidance. Weights rank only admissible options (Prop. 1), which makes QTPO a two-level instance of priority-first selection [12]. Thus w_r trades cryptographic preference against churn, w_m and w_h control rollout pace, and δ suppresses small improvements. Keep w_o small without deployment-specific setup-cost measurements. In H2, a ±50% sweep changes the selected profile on up to 20.5% of edges while leaving $C_{\text{decl}}^{\text{harness}} = 96.5\%$ and unsafe promotions at zero [M]. A nonzero unsafe rate would expose a filter–rank mismatch.

Multi-cluster implications. We measure one cluster and separate those results from the multi-cluster implications. The edge-local gates never refer to a cluster. Deployment does, since cross-cluster edges need an owner (naturally the destination whose `PeerAuthentication` applies) and freshness becomes network-dependent. Shorter windows then turn replication delay into blocked-safe edges and never into unsafe promotions (Thm. 2). Federated partition tests are the next validation step.

Scope and threats to validity. Theorems 1–2 and C_{decl} rest on declared control-plane metadata and never see the bytes on the wire. A workload lying about its capability thus goes undetected. Topology comes from annotations alone, and setup costs are modelled apart from the roughly 1% handshake term. All four competing systems are our re-creations. Only the NoKG/NoProv ablation is a fully controlled comparison, and it shows the smallest gap. The Coq–Go correspondence rests on differential testing, and no machine-checked refinement links the two. The in-cluster runs predate the released controller, and all workloads are controlled testbeds plus one hub-centred Alibaba subset.

Table 2. Capabilities explicitly implemented in the cited configurations: yes (✓), partial (∼), no (✗), or unreported (?). ABAC = attribute-based access control.

Approach	Edge gran.	Prov. gated	Hard gate	Fail closed	Mesh cfg.	PQ wire
Policy-as-code (OPA) [3]	?	?	✓	∼	∼	✗
Service-mesh ABAC [1]	∼	?	✓	∼	✓	✗
Zero-trust mesh control plane [2]	✗	∼	✓	✓	✓	✗
Self-adaptive microservices [5]	∼	✗	∼	∼	∼	✗
Service-level crypto-agility (SL-CA)	✗	✗	∼	✗	✓	✓
Enterprise agility (ELCA) [8]	✗	✗	✓	✗	✓	✓
Config. profiling [9]	✗	✓	∼	✓	∼	✓
QuantumTrustKG (ours)	✓	✓	✓	✓	✓	✗

6 Related Work

In Table 2, a cross does not mean the underlying language or platform cannot express that feature. QoS-aware service selection [4] and microservice self-adaptation [5] rank candidates under a *fixed* correctness posture. Our decision variable is whether a profile is *admissible* on an edge at all, and refusal is a legitimate answer. Policy-as-code [3], service-mesh attribute-based access control (ABAC) [1], and zero-trust control planes [2] implement static-input postures and are not PQC-aware in the cited configurations. Runtime guards for LLM-driven service composition [13] share the fail-closed stance and block any action that their policy automaton refuses. QTPO additionally records a structured promotion-block reason while retaining any previous active profile. Crypto-agility systems are the nearest comparison. ELCA [8] drives cryptography from *per-service* policy. Configuration profiling [9] builds a *per-endpoint* inventory with provenance but cannot express per-edge divergence, and Cryptography Bills of Materials [11] standardize a *static* inventory. The NIST standards [6] and hybrid TLS guidance [7] define the profile classes QuantumTrustKG deploys per edge.

7 Conclusion

Post-quantum migration is an edge-level decision: two calls to the same service can require different profiles because their peers, policies, and evidence differ. We built QuantumTrustKG around this observation. QTPO filters profiles through capability, policy, threat, provenance, and rollout gates before ranking the survivors by cost, which keeps the control-plane guarantee independent of the cost weights. In the controlled comparison, QTPO records zero invalid promotions and 96.5% declared-label policy compliance. Service-level and opportunistic baselines record 3.4–13.8% invalid promotions on synthetic meshes and 37–60% on sampled production topologies. This evidence covers declared control-plane state and leaves negotiated cryptography unobserved. The next step is to connect the decision to the wire profile by binding attested capabilities to workload identity and Secret Discovery Service material.

Acknowledgments. We acknowledge Ho Chi Minh City University of Technology (HCMUT), VNU-HCM for supporting this study.

References

1. R. Chandramouli, Z. Butcher, and A. Chetal, "Attribute-Based Access Control for Microservices-Based Applications Using a Service Mesh," NIST Special Publication 800-204B, Aug. 2021.
2. A. Poudel et al., "Mazu: A Zero Trust Architecture for Service Mesh Control Planes," in *Proc. EuroSec*, 2025, pp. 49–55.
3. A. P. Jothimani, "Enabling Secure Cloud Governance Using Policy as Code," M.S. thesis, Chalmers Univ. of Technology and Univ. of Gothenburg, Sweden, 2022.
4. N. Li et al., "SCORE: A Resource-Efficient Microservice Orchestration Model Based on Spectral Clustering in Edge Computing," in *Proc. ICSOC 2022*, 2022, pp. 186–202.
5. M. Camilli et al., "Integrated QoS- and Vulnerability-Driven Self-Adaptation for Microservices Applications," in *Proc. ICSOC 2024*, 2025, pp. 55–71.
6. National Institute of Standards and Technology, "Module-Lattice-Based Key-Encapsulation Mechanism Standard" (FIPS 203), "Module-Lattice-Based Digital Signature Standard" (FIPS 204), and "Stateless Hash-Based Digital Signature Standard" (FIPS 205), Aug. 2024.
7. K. Kwiatkowski, P. Kampanakis, B. E. Westerbaan, and D. Stebila, "Post-Quantum Traditional (PQ/T) Hybrid Key Agreement Mechanisms for TLS 1.3," RFC 10024, Aug. 2026.
8. D. Sikeridis et al., "ELCA: Introducing Enterprise-level Cryptographic Agility for a Post-Quantum Era," *IACR Cryptology ePrint Archive*, Report 2023/1539, 2023.
9. H. Balaji et al., "Operationalising Post Quantum TLS Automated Configuration Profiling and Hybrid PQC Deployment in Financial Infrastructure," *arXiv:2605.17955*, 2026.
10. Alibaba Group, "Alibaba Cluster Trace Program: cluster-trace-microservices-v2022," 2022. https://github.com/alibaba/clusterdata.
11. OWASP CycloneDX, "Cryptography Bill of Materials (CBOM)," CycloneDX Specification v1.6, 2024. https://cyclonedx.org/capabilities/cbom/.
12. X.-B. Le et al., "Lexi: Priority-First Service Orchestration," in *Proc. ICSOC*, LNCS, Springer, 2026, to appear.
13. X.-B. Le et al., "Deon: Runtime Guards for Policy-Safe Agentic Service Composition under Attack," in *Proc. ICSOC*, LNCS, Springer, 2026, to appear.
14. M. Mosca, "Cybersecurity in an Era with Quantum Computers: Will We Be Ready?," *IEEE Security & Privacy*, vol. 16, no. 5, pp. 38–41, 2018.
15. C. Paquin, D. Stebila, and G. Tamvada, "Benchmarking Post-Quantum Cryptography in TLS," in *Proc. PQCrypto*, LNCS, Springer, 2020, pp. 72–91.
16. A. Hogan et al., "Knowledge Graphs," *ACM Computing Surveys*, vol. 54, no. 4, pp. 1–37, 2021.